J. Darbon et al. *Res Math Sci(2020)7:20*
https://doi.org/10.1007/s40687-020-00215-6

Research in the Mathematical Sciences

RESEARCH

Overcoming the curse of dimensionality for some Hamilton–Jacobi partial differential equations via neural network architectures

Jérôme Darbon*, Gabriel P. Langlois and Tingwei Meng

*Correspondence:
jerome_darbon@brown.edu
Division of Applied Mathematics, Brown University, Providence, USA
Research supported by NSF DMS 1820821. Authors' names are given in last/family name alphabetical order

Abstract

We propose new and original mathematical connections between Hamilton–Jacobi (HJ) partial differential equations (PDEs) with initial data and neural network architectures. Specifically, we prove that some classes of neural networks correspond to representation formulas of HJ PDE solutions whose Hamiltonians and initial data are obtained from the parameters of the neural networks. These results do not rely on universal approximation properties of neural networks; rather, our results show that some classes of neural network architectures naturally encode the physics contained in some HJ PDEs. Our results naturally yield efficient neural network-based methods for evaluating solutions of some HJ PDEs in high dimension without using grids or numerical approximations. We also present some numerical results for solving some inverse problems involving HJ PDEs using our proposed architectures.

1 Introduction

The Hamilton–Jacobi (HJ) equations are an important class of partial differential equation (PDE) models that arise in many scientific disciplines, e.g., physics [6,25,26,33,101], imaging science [38–40], game theory [13,24,49,82], and optimal control [9,46,55,56,110]. Exact or approximate solutions to these equations then give practical insight about the models in consideration. We consider here HJ PDEs specified by a Hamiltonian function $H\colon \mathbb{R}^n \to \mathbb{R}$ and convex initial data $J\colon \mathbb{R}^n \to \mathbb{R}$

$$\begin{cases} \dfrac{\partial S}{\partial t}(\boldsymbol{x},t) + H(\nabla_{\boldsymbol{x}} S(\boldsymbol{x},t)) = 0 & \text{in } \mathbb{R}^n \times (0,+\infty), \\ S(\boldsymbol{x},0) = J(\boldsymbol{x}) & \text{in } \mathbb{R}^n, \end{cases} \tag{1}$$

where $\frac{\partial S}{\partial t}(\boldsymbol{x},t)$ and $\nabla_{\boldsymbol{x}} S(\boldsymbol{x},t) = \left(\frac{\partial S}{\partial x_1}(\boldsymbol{x},t), \ldots, \frac{\partial S}{\partial x_n}(\boldsymbol{x},t)\right)$ denote the partial derivative with respect to t and the gradient vector with respect to $\boldsymbol{x}$ of the function $(\boldsymbol{x},t) \mapsto S(\boldsymbol{x},t)$, and the Hamiltonian H only depends on the gradient $\nabla_{\boldsymbol{x}} S(\boldsymbol{x},t)$.

Our main motivation is to compute the viscosity solution of certain HJ PDEs of the form of (1) in high dimension for a given $\boldsymbol{x} \in \mathbb{R}^n$ and $t > 0$ [9–11,34] by leveraging new efficient hardware technologies and silicon-based electric circuits dedicated to neural networks. As noted by LeCun in [102], the use of neural networks has been greatly influenced by available hardware. In addition, there have been many initiatives to create new hardware for neural networks that yield extremely efficient (in terms of speed, latency, throughput or energy) implementations: For instance, [50–52] propose efficient neural network implementations using field-programmable gate array, [8] optimizes neural network implementations for Intel's architecture, and [96] provides efficient hardware implementation of certain building blocks widely used in neural networks. It is also worth mentioning that Google created specific hardware, called "Tensor Processor Unit" [87] to implement their neural networks in data centers. Note that Xilinx announced a new set of hardware (Versal AI core) for implementing neural networks while Intel enhances their processors with specific hardware instructions for neural networks. LeCun also suggests in [102, Section 3] possible new trends for hardware dedicated to neural networks. Finally, we refer the reader to [30] (see also [69]) that describes the evolution of silicon-based electrical circuits for machine learning.

In this paper, we propose classes of neural network architectures that exactly represent viscosity solutions of certain HJ PDEs of the form of (1). Our results pave the way to leverage efficient dedicated hardware implementation of neural networks to evaluate viscosity solutions of certain HJ PDEs for initial data which takes a particular form.

Related work The viscosity solution to the HJ PDE (1) rarely admits a closed-form expression, and in general it must be computed with numerical algorithms or other methods tailored for the Hamiltonian H, initial data J, and dimension n.

The dimensionality, in particular, matters significantly because in many applications involving HJ PDE models, the dimension n is extremely large. In imaging problems, for example, the vector $\boldsymbol{x}$ typically corresponds to a noisy image whose entries are its pixel values, and the associated Hamilton–Jacobi equations describe the solution to an image denoising convex optimization problem [38,39]. Denoising a 1080 x 1920 standard full HD image on a smartphone, for example, corresponds to solving a HJ PDE in dimension $n = 1080 \times 1920 = 2{,}073{,}600$.

Unfortunately, standard grid-based numerical algorithms for PDEs are impractical when $n > 4$. Such algorithms employ grids to discretize the spatial and time domain, and the number of grid points required to evaluate accurately solutions of PDEs grows exponentially with the dimension n. It is therefore essentially impossible in practice to numerically solve PDEs in high dimension using grid-based algorithms, even with sophisticated high-order accuracy methods for HJ PDEs such as ENO [121], WENO [84], and DG [75]. This problem is known as the *curse of dimensionality* [17].

Overcoming the curse of dimensionality in general remains an open problem, but for HJ PDEs several methods have been proposed to solve it. These include, but are not limited to, max-plus algebra methods [2,3,45,54,60,110–113], dynamic programming and reinforcement learning [4,19], tensor decomposition techniques [44,73,142], sparse grids [20,59,90], model order reduction [5,97], polynomial approximation [88,89], multilevel Picard method [79–81,146], optimization methods [38–40,151] and neural networks [7,42,64,76,77,83,100,120,131,134,136,138]. Among these methods, neural networks have become increasingly popular tools to solve PDEs [7,14–16,18,29,31,41–43,53,58,

62–65,74,76–78,85,92,93,98–100,104,109,114,115,118,120,123,131,134–136,138–140,144,145,148–150] and inverse problems involving PDEs [107,108,116,117,122,126–130,143,149,152,153]. Their popularity is due to universal approximation theorems that state that neural networks can approximate broad classes of (high-dimensional, nonlinear) functions on compact sets [35,71,72,124]. These properties, in particular, have been recently leveraged to approximate solutions to high-dimensional nonlinear HJ PDEs [64,138] and for the development of physics-informed neural networks that aim to solve supervised learning problems while respecting any given laws of physics described by a set of nonlinear PDEs [128].

In this paper, we propose some neural network architectures that exactly represent viscosity solutions to HJ PDEs of the form of (1), where the Hamiltonians and initial data are obtained from the parameters of the neural network architectures. Recall our results require the initial data J to be convex and the Hamiltonian H to only depend on the gradient $\nabla_{\boldsymbol{x}} S(\boldsymbol{x}, t)$ [see Eq. (1)]. In other words, we show that some neural networks correspond to exact representation formulas of HJ PDE solutions. To our knowledge, this is the first result that shows that certain neural networks can exactly represent solutions of certain HJ PDEs.

Note that an alternative method to numerically evaluate solutions of HJ PDEs of the form of (1) with convex initial data has been proposed in [40]. This method relies on the Hopf formula and is only based on optimization. Therefore, this method is grid and approximation-free and works well in high dimension. Contrary to [40], our proposed approach does not rely on any (possibly non-convex) optimization techniques.

Contributions of this paper In this paper, we prove that some classes of shallow neural networks are, under certain conditions, viscosity solutions to Hamilton–Jacobi equations for initial data which takes a particular form. The main result of this paper is Theorem 3.1. We show in this theorem that the neural network architecture illustrated in Fig. 1 represents, under certain conditions, the viscosity solution to a set of first-order HJ PDEs of the form of (1), where the Hamiltonians and the convex initial data are obtained from the parameters of the neural network. As a corollary of this result for the one-dimensional case, we propose a second neural network architecture (illustrated in Fig. 4) that represents the spatial gradient of the viscosity solution of the HJ PDE above in 1D and show in Proposition 3.1 that under appropriate conditions, this neural network corresponds to entropy solutions of some conservation laws in 1D.

Let us emphasize that the proposed architecture in Fig. 1 for representing solutions to HJ PDEs allows us to numerically evaluate their solutions in high dimension without using grids.

We also stress that our results do not rely on universal approximation properties of neural networks. Instead, our results show that the physics contained in HJ PDEs satisfying the conditions of Theorem 3.1 can naturally be encoded by the neural network architecture depicted in Fig. 1. Our results further suggest interpretations of this neural network architecture in terms of solutions to PDEs.

We also test the proposed neural network architecture (depicted in Fig. 1) on some inverse problems. To do so, we consider the following problem. Given training data sampled from the solution S of a first-order HJ PDE (1) with unknown convex initial function J and Hamiltonian H, we aim to recover the unknown initial function. After the training process using the Adam optimizer, the trained neural network with input time variable

$t = 0$ gives an approximation to the convex initial function J. Moreover, the parameters in the trained neural network also provide partial information on the Hamiltonian H. The parameters only approximate the Hamiltonian at certain points, however, and therefore do not give complete information about the function. We show the experimental results on several examples. Our numerical results show that this problem cannot generally be solved using Adam optimizer with high accuracy. In other words, while our theoretical results (see Theorem 3.1) show that the neural network representation (depicted in Fig. 1) to some HJ PDEs is exact, the Adam optimizer for training the proposed networks in this paper sometimes gives large errors in some of our inverse problems, and as such there is no guarantee that the Adam optimizer works well for the proposed network.

Organization of this paper In Sect. 2, we briefly review shallow neural networks and concepts of convex analysis that will be used throughout this paper. In Sect. 3, we establish connections between the neural network architecture illustrated in Fig. 1 and viscosity solutions to HJ PDEs of the form of (1), and the neural network architecture illustrated in Fig. 4 and one-dimensional conservation laws. The mathematical setup for establishing these connections is described in Sect. 3.1, our main results, which concern first-order HJ PDEs, are described in Sect. 3.2, and an extension of these results to one-dimensional conservation laws is presented in Sect. 3.3. In Sect. 4, we perform numerical experiments to test the effectiveness of the Adam optimizer using our proposed architecture (depicted in Fig. 1) for solving some inverse problems. Finally, we draw some conclusions and directions for future work in Sect. 5. Several appendices contain proofs of our results.

2 Background

In this section, we introduce mathematical concepts that will be used in this paper. We review the standard structure of shallow neural networks from a mathematical point of view in Sect. 2.1 and present some fundamental definitions and results in convex analysis in Sect. 2.2. For the notation, we use $\mathbb{R}^n$ to denote the n-dimensional Euclidean space. The Euclidean scalar product and Euclidean norm on $\mathbb{R}^n$ are denoted by $\langle \cdot, \cdot \rangle$ and $\|\cdot\|_2$. The set of matrices with m rows and n columns with real entries is denoted by $\mathcal{M}_{m,n}(\mathbb{R})$.

2.1 Shallow neural networks

Neural networks provide architectures for constructing complicated nonlinear functions from simple building blocks. Common neural network architectures in applications include, for example, feedforward neural networks in statistical learning, recurrent neural networks in natural language processing, and convolutional neural networks in imaging science. In this paper, we focus on shallow neural networks, a subclass of feedforward neural networks that typically consist of one hidden layer and one output layer. We give here a brief mathematical introduction to shallow neural networks. For more details, we refer the reader to [61,103,137] and the references listed therein.

A shallow neural network with one hidden layer and one output layer is a composition of affine functions with a nonlinear function. A hidden layer with $m \in \mathbb{N}$ neurons comprises m affine functions of an *input* $\boldsymbol{x} \in \mathbb{R}^n$ with *weights* $\boldsymbol{w}_i \in \mathbb{R}^n$ and *biases* $b_i \in \mathbb{R}$:

$$\mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R} \ni (\boldsymbol{x}, \boldsymbol{w}_i, b_i) \mapsto \langle \boldsymbol{w}_i, \boldsymbol{x} \rangle + b_i.$$

These m affine functions can be succinctly written in vector form as $\boldsymbol{W}\boldsymbol{x}+\boldsymbol{b}$, where the matrix $\boldsymbol{W}\in\mathcal{M}_{m,n}(\mathbb{R})$ has for rows the weights $\boldsymbol{w}_i$ and the vector $b\in\mathbb{R}^m$ has for entries the biases b_i. The output layer comprises a nonlinear function $\sigma:\mathbb{R}^m\to\mathbb{R}$ that takes for input the vector $\boldsymbol{W}\boldsymbol{x}+\boldsymbol{b}$ of affine functions and gives the number

$$\mathbb{R}^n\times\mathbb{R}^n\times\mathbb{R}\ni(\boldsymbol{x},\boldsymbol{w}_i,b_i)\mapsto\sigma\left(\boldsymbol{W}\boldsymbol{x}+\boldsymbol{b}\right).$$

The nonlinear function σ is called the *activation function* of the output layer.

In Sect. 4, we will consider the following problem: Given data points $\{(\boldsymbol{x}_i,y_i)\}_{i=1}^N\subset\mathbb{R}^n\times\mathbb{R}$, infer the relationship between the input $\boldsymbol{x}_i$'s and the output y_i's. To infer this relation, we assume that the output takes the form (or can be approximated by) $y_i=\sigma\left(\boldsymbol{W}\boldsymbol{x}_i+\boldsymbol{b}\right)$ for some known activation function σ, unknown matrix of weights $\boldsymbol{W}\in\mathcal{M}_{m,n}(\mathbb{R})$, and unknown vector of bias $\boldsymbol{b}$. A standard approach to solve such a problem is to estimate the weights $\boldsymbol{w}_i$ and biases b_i so as to minimize the mean square error

$$\{(\bar{\boldsymbol{w}}_i,\bar{b}_i)\}_{i=1}^m\in\underset{\{(\boldsymbol{w}_i,b_i)\}_{i=1}^m\subset\mathbb{R}^n\times\mathbb{R}}{\arg\min}\left\{\frac{1}{N}\sum_{i=1}^N\left(\sigma\left(\boldsymbol{W}\boldsymbol{x}_i+\boldsymbol{b}\right)-y_i\right)^2\right\}. \tag{2}$$

In the field of machine learning, solving this minimization problem is called the *learning* or *training process*. The data $\{(\boldsymbol{x}_i,y_i)\}_{i=1}^N$ used in the training process is called *training data*. Finding a global minimizer is generally difficult due to the complexity of the minimization problem and that the objective function is not convex with respect to the weights and biases. State-of-the-art algorithms for solving these problems are stochastic gradient descent-based methods with momentum acceleration, such as the Adam optimizer for neural networks [94]. This algorithm will be used in our numerical experiments.

2.2 Convex analysis

We introduce here several definitions and results of convex analysis that will be used in this paper. We refer readers to Hiriart–Urruty and Lemaréchal [67,68] and Rockafellar [133] for comprehensive references on finite-dimensional convex analysis.

Definition 1 (*Convex sets, relative interiors, and convex hulls*) A set $C\subset\mathbb{R}^n$ is called convex if for any $\lambda\in[0,1]$ and any $\boldsymbol{x},\boldsymbol{y}\in C$, the element $\lambda\boldsymbol{x}+(1-\lambda)\boldsymbol{y}$ is in C. The relative interior of a convex set $C\subset\mathbb{R}^n$, denoted by ri C, consists of the points in the interior of the unique smallest affine set containing C. The convex hull of a set C, denoted by conv C, consists of all the convex combinations of the elements of C. An important example of a convex hull is the unit simplex in $\mathbb{R}^n$, which we denote by

$$\Lambda_n:=\left\{(\alpha_1,\ldots,\alpha_n)\in[0,1]^n:\ \sum_{i=1}^n\alpha_i=1\right\}. \tag{3}$$

Definition 2 (*Domains and proper functions*) The domain of a function $f:\mathbb{R}^n\to\mathbb{R}\cup\{+\infty\}$ is the set

$$\operatorname{dom}f=\left\{\boldsymbol{x}\in\mathbb{R}^n:f(\boldsymbol{x})<+\infty\right\}.$$

A function f is called proper if its domain is non-empty and $f(\boldsymbol{x})>-\infty$ for every $\boldsymbol{x}\in\mathbb{R}^n$.

Definition 3 (*Convex functions, lower semicontinuity, and convex envelopes*) A proper function $f\colon \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is called convex if the set $\operatorname{dom} f$ is convex and if for any $\boldsymbol{x}, \boldsymbol{y} \in \operatorname{dom} f$ and all $\lambda \in [0,1]$, there holds

$$f(\lambda \boldsymbol{x} + (1-\lambda)\boldsymbol{y}) \leqslant \lambda f(\boldsymbol{x}) + (1-\lambda) f(\boldsymbol{y}) \tag{4}$$

A proper function $f\colon \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is called lower semicontinuous if for every sequence $\{\boldsymbol{x}_k\}_{k=1}^{+\infty} \in \mathbb{R}^n$ with $\lim_{k\to+\infty} \boldsymbol{x}_k = \boldsymbol{x} \in \mathbb{R}^n$, we have $\liminf_{k\to+\infty} f(\boldsymbol{x}_k) \geqslant f(\boldsymbol{x})$.

The class of proper, lower semicontinuous convex functions is denoted by $\Gamma_0(\mathbb{R}^n)$.

Given a function $f\colon \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, we define its convex envelope $\operatorname{co} f$ as the largest convex function such that $\operatorname{co} f(\boldsymbol{x}) \leqslant f(\boldsymbol{x})$ for every $\boldsymbol{x} \in \mathbb{R}^n$. We define the convex lower semicontinuous envelope $\overline{\operatorname{co}} f$ as the largest convex and lower semicontinuous function such that $\overline{\operatorname{co}} f(\boldsymbol{x}) \leqslant f(\boldsymbol{x})$ for every $\boldsymbol{x} \in \mathbb{R}^n$.

Definition 4 (*Subdifferentials and subgradients*) The subdifferential $\partial f(\boldsymbol{x})$ of $f \in \Gamma_0(\mathbb{R}^n)$ at $\boldsymbol{x} \in \operatorname{dom} f$ is the set (possibly empty) of vectors $\boldsymbol{p} \in \mathbb{R}^n$ satisfying

$$\forall \boldsymbol{y} \in \mathbb{R}^n,\ f(\boldsymbol{y}) \geqslant f(\boldsymbol{x}) + \langle \boldsymbol{p}, \boldsymbol{y} - \boldsymbol{x} \rangle. \tag{5}$$

The subdifferential $\partial f(\boldsymbol{x})$ is a closed convex set whenever it is non-empty, and any vector $\boldsymbol{p} \in \partial f(\boldsymbol{x})$ is called a subgradient of f at $\boldsymbol{x}$. If f is a proper convex function, then $\partial f(\boldsymbol{x}) \neq \emptyset$ whenever $\boldsymbol{x} \in \operatorname{ri}(\operatorname{dom} f)$, and $\partial f(\boldsymbol{x}) = \emptyset$ whenever $\boldsymbol{x} \notin \operatorname{dom} J$ [133, Thm. 23.4]. If a convex function f is differentiable at $\boldsymbol{x}_0 \in \mathbb{R}^n$, then its gradient $\nabla_{\boldsymbol{x}} f(\boldsymbol{x}_0)$ is the unique subgradient of f at $\boldsymbol{x}_0$, and conversely if f has a unique subgradient at $\boldsymbol{x}_0$, then f is differentiable at that point [133, Thm. 21.5].

Definition 5 (*Fenchel–Legendre transforms*) Let $f \in \Gamma_0(\mathbb{R}^n)$. The Fenchel–Legendre transform $f^*\colon \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ of f is defined as

$$f^*(\boldsymbol{p}) = \sup_{\boldsymbol{x}\in\mathbb{R}^n} \left\{ \langle \boldsymbol{p}, \boldsymbol{x} \rangle - f(\boldsymbol{x}) \right\}. \tag{6}$$

For any $f \in \Gamma_0(\mathbb{R}^n)$, the mapping $f \mapsto f^*$ is one-to-one, $f^* \in \Gamma_0(\mathbb{R}^n)$, and $(f^*)^* = f$. Moreover, for any $(\boldsymbol{x}, \boldsymbol{p}) \in \mathbb{R}^n \times \mathbb{R}^n$, the so-called Fenchel's inequality holds:

$$f(\boldsymbol{x}) + f(\boldsymbol{p}) \geqslant \langle \boldsymbol{x}, \boldsymbol{p} \rangle, \tag{7}$$

with equality attained if and only if $\boldsymbol{p} \in \partial f(\boldsymbol{x})$, if and only if $\boldsymbol{x} \in \partial f^*(\boldsymbol{p})$ [68, Cor. X.1.4.4].

We summarize some notations and definitions in Table 1.

3 Connections between neural networks and Hamilton–Jacobi equations

This section establishes connections between HJ PDEs and neural network architectures. Subsection 3.1 presents the mathematical setup, subsection 3.2 describes our main results for first-order HJ PDEs, and finally subsection 3.3 presents our results for first-order one-dimensional conservation laws.

Table 1 Notation used in this paper. Here, we use C to denote a set in $\mathbb{R}^n$, f to denote a function from $\mathbb{R}^n$ to $\mathbb{R} \cup \{+\infty\}$ and x to denote a vector in $\mathbb{R}^n$

Notation	Meaning	Definition
$\langle \cdot, \cdot \rangle$	Euclidean scalar product in $\mathbb{R}^n$	$\langle \boldsymbol{x}, \boldsymbol{y} \rangle := \sum_{i=1}^n x_i y_i$
$\|\cdot\|_2$	Euclidean norm in $\mathbb{R}^n$	$\|\boldsymbol{x}\|_2 := \sqrt{\langle \boldsymbol{x}, \boldsymbol{x} \rangle}$
ri C	Relative interior of C	The interior of C with respect to the minimal hyperplane containing C in $\mathbb{R}^n$
conv C	Convex hull of C	The set containing all convex combinations of the elements of C
Λ_n	Unit simplex in $\mathbb{R}^n$	$\left\{(\alpha_1, \ldots, \alpha_n) \in [0,1]^n : \sum_{i=1}^n \alpha_i = 1\right\}$
dom f	Domain of f	$\{\boldsymbol{x} \in \mathbb{R}^n : f(\boldsymbol{x}) < +\infty\}$
$\Gamma_0(\mathbb{R}^n)$	A useful and standard class of convex functions	The set containing all proper, convex, lower semicontinuous functions from $\mathbb{R}^n$ to $\mathbb{R} \cup \{+\infty\}$
co f	Convex envelope of f	The largest convex function such that $\text{co}\, f(\boldsymbol{x}) \leqslant f(\boldsymbol{x})$ for every $\boldsymbol{x} \in \mathbb{R}^n$
$\overline{\text{co}}\, f$	Convex and lower semicontinuous envelope of f	The largest convex and lower semicontinuous function such that $\overline{\text{co}}\, f(\boldsymbol{x}) \leqslant f(\boldsymbol{x})$ for every $\boldsymbol{x} \in \mathbb{R}^n$
$\partial f(\boldsymbol{x})$	Subdifferential of f at $\boldsymbol{x}$	$\{\boldsymbol{p} \in \mathbb{R}^n : f(\boldsymbol{y}) \geqslant f(\boldsymbol{x}) + \langle \boldsymbol{p}, \boldsymbol{y} - \boldsymbol{x} \rangle \ \forall \boldsymbol{y} \in \mathbb{R}^n\}$
f^*	Fenchel–Legendre transform of f	$f^*(\boldsymbol{p}) := \sup_{\boldsymbol{x} \in \mathbb{R}^n} \{\langle \boldsymbol{p}, \boldsymbol{x} \rangle - f(\boldsymbol{x})\}$

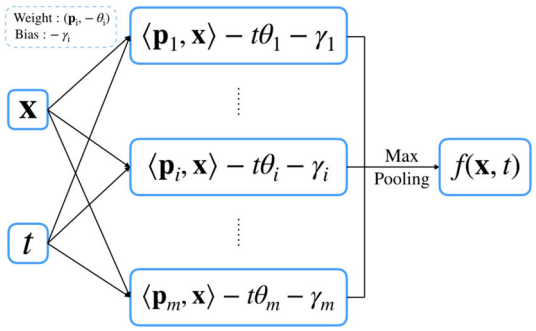

Fig. 1 Illustration of the structure of the neural network (8) that can represent the viscosity solution to first-order Hamilton–Jacobi equations for initial data which takes a particular form

3.1 Setup

In this section, we consider the function $f : \mathbb{R}^n \times [0, +\infty) \to \mathbb{R}$ given by the neural network in Fig. 1. Mathematically, the function f can be expressed using the following formula

$$f(\boldsymbol{x}, t; \{(\boldsymbol{p}_i, \theta_i, \gamma_i)\}_{i=1}^m) = \max_{i \in \{1, \ldots, m\}} \{\langle \boldsymbol{p}_i, \boldsymbol{x} \rangle - t\theta_i - \gamma_i\}. \tag{8}$$

Our goal is to show that the function f in (8) is the unique uniformly continuous viscosity solution to a suitable Hamilton–Jacobi equation. In what follows, we denote $f(\boldsymbol{x}, t; \{(\boldsymbol{p}_i, \theta_i, \gamma_i)\}_{i=1}^m)$ by $f(\boldsymbol{x}, t)$ when there is no ambiguity in the parameters.

We adopt the following assumptions on the parameters:

(A1) The parameters $\{\boldsymbol{p}_i\}_{i=1}^m$ are pairwise distinct, i.e., $\boldsymbol{p}_i \neq \boldsymbol{p}_j$ if $i \neq j$.
(A2) There exists a convex function $g\colon\ \mathbb{R}^n \to \mathbb{R}$ such that $g(\boldsymbol{p}_i) = \gamma_i$.
(A3) For any $j \in \{1, \ldots, m\}$ and any $(\alpha_1, \ldots, \alpha_m) \in \mathbb{R}^m$ that satisfy

$$\begin{cases} (\alpha_1, \ldots, \alpha_m) \in \Lambda_m \quad \text{with } \alpha_j = 0, \\ \sum_{i\neq j} \alpha_i \boldsymbol{p}_i = \boldsymbol{p}_j, \\ \sum_{i\neq j} \alpha_i \gamma_i = \gamma_j, \end{cases} \tag{9}$$

there holds $\sum_{i\neq j} \alpha_i \theta_i > \theta_j$.

Note that (A3) is not a strong assumption. Indeed, if there exist $j \in \{1, \ldots, m\}$ and $(\alpha_1, \ldots, \alpha_m) \in \mathbb{R}^m$ satisfying Eq. (9) and $\sum_{i\neq j} \alpha_i \theta_i \leqslant \theta_j$, then

$$\langle \boldsymbol{p}_j, \boldsymbol{x}\rangle - t\theta_j - \gamma_j \leqslant \sum_{i\neq j} \alpha_i (\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - t\theta_i - \gamma_i) \leqslant \max_{i\neq j} \{\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - t\theta_i - \gamma_i\}.$$

As a result, the jth neuron in the network can be removed without changing the value of $f(\boldsymbol{x}, t)$ for any $\boldsymbol{x} \in \mathbb{R}^n$ and $t \geqslant 0$. Removing all such neurons in the network, we can therefore assume (A3) holds.

Our aim is to identify the HJ equations whose viscosity solutions correspond to the neural network f defined by Eq. (8). Here, $\boldsymbol{x}$ and t play the role of the spatial and time variables, and $f(\cdot, 0)$ corresponds to the initial data. To simplify the notation, we define the function $J\colon\ \mathbb{R}^n \to \mathbb{R}$ as

$$f(\boldsymbol{x}, 0) = J(\boldsymbol{x}) := \max_{i\in\{1,\ldots,m\}} \{\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - \gamma_i\} \tag{10}$$

and the set $I_{\boldsymbol{x}}$ as the collection of maximizers in Eq. (10) at $\boldsymbol{x}$, that is,

$$I_{\boldsymbol{x}} := \arg\max_{i\in\{1,\ldots,m\}} \{\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - \gamma_i\}. \tag{11}$$

Note that the initial data J given by (10) is a convex and polyhedral function, and it satisfies several properties that we describe in the following lemma.

Lemma 3.1 *Suppose $\{(\boldsymbol{p}_i, \gamma_i)\}_{i=1}^m \subset \mathbb{R}^n \times \mathbb{R}$ satisfy assumptions (A1) and (A2). Then the following statements hold.*

(i) The Fenchel–Legendre transform of J is given by the convex and lower semicontinuous function

$$J^*(\boldsymbol{p}) = \begin{cases} \min\limits_{\substack{(\alpha_1,\ldots,\alpha_m)\in\Lambda_m \\ \sum_{i=1}^m \alpha_i \boldsymbol{p}_i = \boldsymbol{p}}} \left\{ \sum\limits_{i=1}^m \alpha_i \gamma_i \right\} & \text{if } \ \boldsymbol{p} \in \operatorname{conv}(\{\boldsymbol{p}_i\}_{i=1}^m), \\ +\infty & \text{otherwise.} \end{cases} \tag{12}$$

Moreover, its restriction to $\operatorname{dom} J^*$ *is continuous, and the subdifferential $\partial J^*(\boldsymbol{p})$ is non-empty for every $\boldsymbol{p} \in \operatorname{dom} J^*$.*

(ii) *Let* $\boldsymbol{p} \in \operatorname{dom} J^*$ *and* $\boldsymbol{x} \in \partial J^*(\boldsymbol{p})$. *Then,* $(\alpha_1, \ldots, \alpha_m) \in \mathbb{R}^m$ *is a minimizer in Eq.* (12) *if and only if it satisfies the constraints*

(a) $(\alpha_1, \ldots, \alpha_m) \in \Lambda_m$,
(b) $\sum_{i=1}^m \alpha_i \boldsymbol{p}_i = \boldsymbol{p}$,
(c) $\alpha_i = 0$ *for any* $i \notin I_{\boldsymbol{x}}$.

(iii) *For each* $i, k \in \{1, \ldots, m\}$, *let*

$$\alpha_i = \delta_{ik} := \begin{cases} 1 & \text{if } \ i = k, \\ 0 & \text{if } \ i \neq k. \end{cases}$$

Then, $(\alpha_1, \ldots, \alpha_m)$ *is a minimizer in Eq.* (12) *at the point* $\boldsymbol{p} = \boldsymbol{p}_k$. *Hence, we have* $J^*(\boldsymbol{p}_k) = \gamma_k$.

Proof See "Appendix A.1" for the proof. □

Having defined the initial condition J, the next step is to define a Hamiltonian H. To do so, first denote by $\mathcal{A}(\boldsymbol{p})$ the set of minimizers in Eq. (12) evaluated at $\boldsymbol{p} \in \operatorname{dom} J^*$, i.e.,

$$\mathcal{A}(\boldsymbol{p}) := \underset{\substack{(\alpha_1, \ldots \alpha_m) \in \Lambda_m \\ \sum_{i=1}^m \alpha_i \boldsymbol{p}_i = \boldsymbol{p}}}{\arg\min} \left\{ \sum_{i=1}^m \alpha_i \gamma_i \right\}. \tag{13}$$

Note that the set $\mathcal{A}(\boldsymbol{p})$ is non-empty for every $\boldsymbol{p} \in \operatorname{dom} J^*$ by Lemma 3.1(i). Now, we define the Hamiltonian function $H : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ by

$$H(\boldsymbol{p}) := \begin{cases} \inf\limits_{\boldsymbol{\alpha} \in \mathcal{A}(\boldsymbol{p})} \left\{ \sum\limits_{i=1}^m \alpha_i \theta_i \right\} & \text{if } \ \boldsymbol{p} \in \operatorname{dom} J^*, \\ +\infty & \text{otherwise.} \end{cases} \tag{14}$$

The function H defined in (14) is a polyhedral function whose properties are stated in the following lemma.

Lemma 3.2 *Suppose* $\{(\boldsymbol{p}_i, \theta_i, \gamma_i)\}_{i=1}^m \subset \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}$ *satisfy assumptions (A1)–(A3). Then, the following statements hold:*

(i) *For every* $\boldsymbol{p} \in \operatorname{dom} J^*$, *the set* $\mathcal{A}(\boldsymbol{p})$ *is compact and Eq.* (14) *has at least one minimizer.*
(ii) *The restriction of* H *to* $\operatorname{dom} J^*$ *is a bounded and continuous function.*
(iii) *There holds* $H(\boldsymbol{p}_i) = \theta_i$ *for each* $i \in \{1, \ldots, m\}$.

Proof See "Appendix A.2" for the proof. □

3.2 Main results: First-order Hamilton–Jacobi equations

Let f be the function represented by the neural network architecture in Fig. 1, whose mathematical definition is given in Eq. (8). In the following theorem, we identify the set of first-order HJ equations whose viscosity solutions correspond to the neural network f. Specifically, f solves a first-order HJ equation with Hamiltonian H and initial function J that were defined previously in Eqs. (14) and (10), respectively. Furthermore, we provide necessary and sufficient conditions for a first-order HJ equation of the form of (1) with initial data given in the form of (10) to have for viscosity solution the neural network f.

Theorem 3.1 *Suppose the parameters* $\{(\boldsymbol{p}_i, \theta_i, \gamma_i)\}_{i=1}^m \subset \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}$ *satisfy assumptions (A1)-(A3), and let* f *be the neural network defined by Eq. (8) with these parameters. Let* J *and* H *be the functions defined in Eqs. (10) and (14), respectively, and let* $\tilde{H}\colon \mathbb{R}^n \to \mathbb{R}$ *be a continuous function. Then the following two statements hold.*

(i) The neural network f *is the unique uniformly continuous viscosity solution to the first-order Hamilton–Jacobi equation*

$$\begin{cases} \dfrac{\partial f}{\partial t}(\boldsymbol{x}, t) + H(\nabla_{\boldsymbol{x}} f(\boldsymbol{x}, t)) = 0, & \text{in } \mathbb{R}^n \times (0, +\infty), \\ f(\boldsymbol{x}, 0) = J(\boldsymbol{x}), & \text{in } \mathbb{R}^n. \end{cases} \tag{15}$$

Moreover, f *is jointly convex in* $(\boldsymbol{x},t)$.

(ii) The neural network f *is the unique uniformly continuous viscosity solution to the first-order Hamilton–Jacobi equation*

$$\begin{cases} \dfrac{\partial f}{\partial t}(\boldsymbol{x}, t) + \tilde{H}(\nabla_{\boldsymbol{x}} f(\boldsymbol{x}, t)) = 0, & \text{in } \mathbb{R}^n \times (0, +\infty), \\ f(\boldsymbol{x}, 0) = J(\boldsymbol{x}), & \text{in } \mathbb{R}^n, \end{cases} \tag{16}$$

if and only if $\tilde{H}(\boldsymbol{p}_i) = H(\boldsymbol{p}_i)$ *for each* $i \in \{1, \ldots, m\}$ *and* $\tilde{H}(\boldsymbol{p}) \geqslant H(\boldsymbol{p})$ *for every* $\boldsymbol{p} \in \operatorname{dom} J^*$.

Proof See "Appendix B" for the proof. □

Remark 1 This theorem identifies the set of HJ equations with initial data J whose solution is given by the neural network f. To each such HJ equation, there corresponds a continuous Hamiltonian $\tilde{H}$ satisfying $\tilde{H}(\boldsymbol{p}_i) = H(\boldsymbol{p}_i)$ for every $i = \{1, \ldots, m\}$ and $\tilde{H}(\boldsymbol{p}) \geqslant H(\boldsymbol{p})$ for every $\boldsymbol{p} \in \operatorname{dom} J^*$. The smallest possible Hamiltonian satisfying these constraints is the function H defined in (14), and its corresponding HJ equation is given by (15).

Example 1 In this example, we consider the HJ PDE with initial data $J^{\text{true}}(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$ and the Hamiltonian $H^{\text{true}}(\boldsymbol{p}) = -\frac{\|\boldsymbol{p}\|_2^2}{2}$ for all $\boldsymbol{x}, \boldsymbol{p} \in \mathbb{R}^n$. The viscosity solution to this HJ PDE is given by

$$S(\boldsymbol{x}, t) = \|\boldsymbol{x}\|_1 + \frac{nt}{2} = \max_{i \in \{1, \ldots, m\}} \{\langle \boldsymbol{p}_i, \boldsymbol{x} \rangle - t\theta_i - \gamma_i\} \text{ for every } \boldsymbol{x} \in \mathbb{R}^n \text{ and } t \geqslant 0,$$

where $m = 2^n$, each entry of $\boldsymbol{p}_i$ takes value in $\{\pm 1\}$, and $\theta_i = -\frac{n}{2}$, $\gamma_i = 0$ for every $i \in \{1, \ldots, m\}$. In other words, the solution S can be represented using the proposed neural network with parameters $\{(\boldsymbol{p}_i, -\frac{n}{2}, 0)\}_{i=1}^m$. We can compute the functions J and H using definitions in Eqs. (10) and (14) and then obtain

$$J(\boldsymbol{x}) = \|\boldsymbol{x}\|_1 = J^{\text{true}}(\boldsymbol{x}) \text{ for every } \boldsymbol{x} \in \mathbb{R}^n;$$

$$H(\boldsymbol{p}) = \begin{cases} -\frac{n}{2}, & \boldsymbol{p} \in [-1, 1]^n; \\ +\infty, & \text{otherwise.} \end{cases}$$

Theorem 3.1 stipulates that S solves the HJ PDE (16) if and only if $\tilde{H}(\boldsymbol{p}_i) = -\frac{n}{2}$ for every $i \in \{1, \ldots, m\}$ and $\tilde{H}(\boldsymbol{p}) \geq -\frac{n}{2}$ for every $\boldsymbol{p} \in [-1, 1]^n \setminus \{\boldsymbol{p}_i\}_{i=1}^m$. The Hamiltonian H^{true} is one candidate satisfying these constraints.

Example 2 In this example, we consider the case when $J^{\text{true}}(\boldsymbol{x}) = \|\boldsymbol{x}\|_\infty$ and $H^{\text{true}}(\boldsymbol{p}) = -\frac{\|\boldsymbol{p}\|_2^2}{2}$ for every $\boldsymbol{x}, \boldsymbol{p} \in \mathbb{R}^n$. Denote by $\boldsymbol{e}_i$ the ith standard unit vector in $\mathbb{R}^n$. Let $m = 2n$, $\{\boldsymbol{p}_i\}_{i=1}^m = \{\pm \boldsymbol{e}_i\}_{i=1}^n$, $\theta_i = -\frac{n}{2}$, and $\gamma_i = 0$ for every $i \in \{1, \ldots, m\}$. The viscosity solution S is given by

$$S(\boldsymbol{x}, t) = \|\boldsymbol{x}\|_\infty + \frac{nt}{2} = \max_{i \in \{1,\ldots,m\}} \{\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - t\theta_i - \gamma_i\} \text{ for every } \boldsymbol{x} \in \mathbb{R}^n \text{ and } t \geqslant 0.$$

Hence, S can be represented using the proposed neural network with parameters $\{(\boldsymbol{p}_i, -\frac{n}{2}, 0)\}_{i=1}^m$. Similarly, as in the first example, we compute J and H and obtain the following results

$$J(\boldsymbol{x}) = \|\boldsymbol{x}\|_\infty \text{ for every } \boldsymbol{x} \in \mathbb{R}^n;$$
$$H(\boldsymbol{p}) = \begin{cases} -\frac{n}{2} & \boldsymbol{p} \in B_n; \\ +\infty & \text{otherwise,} \end{cases}$$

where B_n denotes the unit ball with respect to the l^1 norm in $\mathbb{R}^n$, i.e., $B_n = \text{conv}\,\{\pm \boldsymbol{e}_i : i \in \{1, \ldots, n\}\}$. By Theorem 3.1, S is a viscosity solution to the HJ PDE (16) if and only if $\tilde{H}(\boldsymbol{p}_i) = -\frac{n}{2}$ for every $i \in \{1, \ldots, m\}$ and $\tilde{H}(\boldsymbol{p}) \geq -\frac{n}{2}$ for every $\boldsymbol{p} \in B_n \backslash \{\boldsymbol{p}_i\}_{i=1}^m$. The Hamiltonian H^{true} is one candidate satisfying these constraints.

Example 3 In this example, we consider the HJ PDE with Hamiltonian $H^{\text{true}}(\boldsymbol{p}) = \|\boldsymbol{p}\|_1$ and initial data $J^{\text{true}}(\boldsymbol{x}) = \max\left\{\|\boldsymbol{x}\|_\infty, \frac{1}{\sqrt{2}}(|x_1| + |x_2|)\right\}$, for all $\boldsymbol{p} \in \mathbb{R}^n$ and $\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n$. The corresponding neural network has $m = 2n + 5$ neurons, where the parameters are given by

$$\{(\boldsymbol{p}_i, \theta_i, \gamma_i)\}_{i=1}^{2n} = \{(\boldsymbol{e}_i, 1, 0)\}_{i=1}^n \cup \{(-\boldsymbol{e}_i, 1, 0)\}_{i=1}^n,$$
$$(\boldsymbol{p}_{2n+1}, \theta_{2n+1}, \gamma_{2n+1}) = (\boldsymbol{0}, 0, 0),$$
$$\{(\boldsymbol{p}_i, \theta_i, \gamma_i)\}_{i=2n+2}^{2n+5} = \left\{\frac{1}{\sqrt{2}}(\alpha \boldsymbol{e}_1 + \beta \boldsymbol{e}_2, 2, 0) \colon \alpha, \beta \in \{\pm 1\}\right\},$$

where $\boldsymbol{e}_i$ is the i^{th} standard unit vector in $\mathbb{R}^n$ and $\boldsymbol{0}$ denotes the zero vector in $\mathbb{R}^n$. The functions J and H defined by (10) and (14) coincide with the underlying true initial data J^{true} and Hamiltonian H^{true}. Therefore, by Theorem 3.1, the proposed neural network represents the viscosity solution to the HJ PDE. In other words, given the true parameters $\{(\boldsymbol{p}_i, \theta_i, \gamma_i)\}_{i=1}^m$, the proposed neural network solves this HJ PDE without the curse of dimensionality. We illustrate the solution with dimension $n = 16$ in Fig. 2, which shows several slices of the solution evaluated at $\boldsymbol{x} = (x_1, x_2, 0, \ldots, 0) \in \mathbb{R}^{16}$ and $t = 0, 1, 2, 3$ in figures 2(A), 2(B), 2(C), 2(D), respectively. In each figure, the x and y axes correspond to the first two components x_1 and x_2 in $\boldsymbol{x}$, while the color represents the function value $S(\boldsymbol{x}, t)$.

Remark 2 Let $\epsilon > 0$ and consider the neural network $f_\epsilon \colon \mathbb{R}^n \times [0, +\infty) \to \mathbb{R}$ defined by

$$f_\epsilon(\boldsymbol{x}, t) := \epsilon \log\left(\sum_{i=1}^m e^{(\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - t\theta_i - \gamma_i)/\epsilon}\right) \tag{17}$$

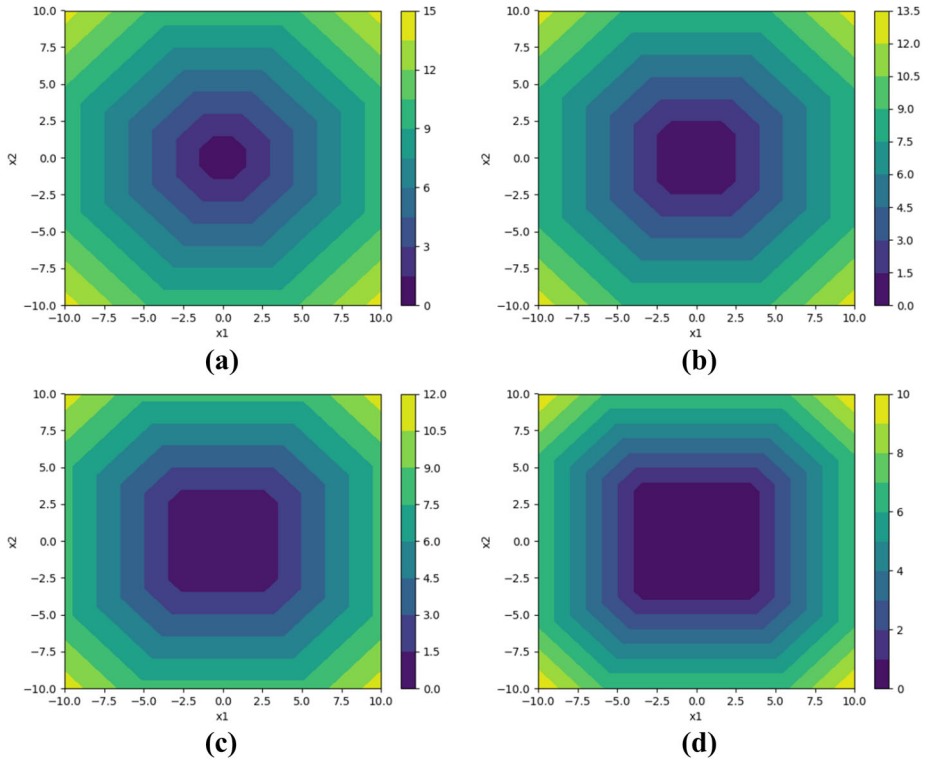

Fig. 2 Solution $S\colon \mathbb{R}^{16} \times [0, +\infty) \to \mathbb{R}$ to the HJ PDE in Example 3 is solved using the proposed neural network. Several slices of the solution S evaluated at $\boldsymbol{x} = (x_1, x_2, 0, \ldots, 0)$ and $t = 0, 1, 2, 3$ are shown in figures 2(A), 2(B), 2(C), 2(D), respectively. In each figure, the x and y axes correspond to the first two components x_1 and x_2 in the variable $\boldsymbol{x}$, while the color represents the function value $S(\boldsymbol{x}, t)$

and illustrated in Fig. 3. This neural network substitutes the non-smooth maximum activation function in the neural network f defined by Eq. (8) (and depicted in Fig. 1) with a smooth log-exponential activation function. When the parameter $\theta_i = -\frac{1}{2}\left\|\boldsymbol{p}_i\right\|_2^2$, then the neural network f_ϵ is the unique, jointly convex and smooth solution to the following viscous HJ PDE

$$\begin{cases} \dfrac{\partial f_\epsilon(\boldsymbol{x}, t)}{\partial t} - \dfrac{1}{2}\left\|\nabla_{\boldsymbol{x}} f_\epsilon(\boldsymbol{x}, t)\right\|_2^2 = \dfrac{\epsilon}{2}\Delta_{\boldsymbol{x}} f_\epsilon(\boldsymbol{x}, t) & \text{in } \mathbb{R}^n \times (0, +\infty), \\ f_\epsilon(\boldsymbol{x}, 0) = \epsilon \log\left(\displaystyle\sum_{i=1}^m e^{(\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - \gamma_i)/\epsilon}\right) & \text{in } \mathbb{R}^n. \end{cases} \tag{18}$$

This result relies on the Cole–Hopf transformation ([47], Sect. 4.4.1); see Appendix C for the proof. While this neural network architecture represents, under certain conditions, the solution to the viscous HJ PDE (18), we note that the particular form of the convex initial data in the HJ PDE (18), which effectively corresponds to a soft Legendre transform in that $\lim_{\substack{\epsilon \to 0 \\ \epsilon > 0}} \epsilon \log\left(\sum_{i=1}^m e^{(\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - \gamma_i)/\epsilon}\right) = \max_{i \in \{1,\ldots,m\}}\{\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - \gamma_i\}$, severely restricts the practicality of this result.

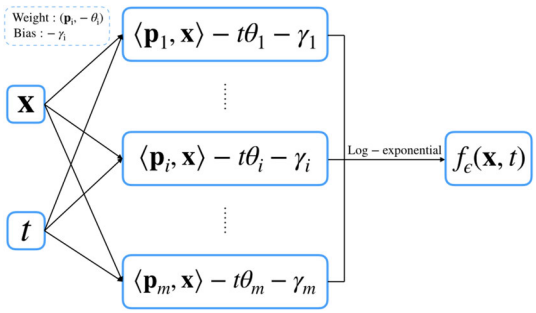

Fig. 3 Illustration of the structure of the neural network (17) that represents the solution to a subclass of second-order HJ equations when $\theta_i = -\frac{1}{2}\|\boldsymbol{p}_i\|_2^2$ for $i \in \{1, \ldots, m\}$

3.3 First-order one-dimensional conservation laws

It is well known that one-dimensional conservation laws are related to HJ equations (see, e.g., [1,22,23,28,32,86,91,95,106], and also [37] for a comprehensive introduction to conservation laws and entropy solutions). Formally, by taking spatial gradient of the HJ equation (1) and identifying the gradient $\nabla_x f \equiv u$, we obtain the conservation law

$$\begin{cases} \dfrac{\partial u}{\partial t}(x,t) + \nabla_x H(u(x,t)) = 0 & \text{in } \mathbb{R} \times (0, +\infty), \\ u(x,0) = u_0(x) := \nabla J(x) & \text{in } \mathbb{R}, \end{cases} \tag{19}$$

where the flux function corresponds to the Hamiltonian H in the HJ equation. Here, we assume that the initial data J is convex and globally Lipschitz continuous, and the symbols ∇ and ∇_x in this section correspond to derivatives in the sense of distribution if the classical derivatives do not exist.

In this section, we show that the conservation law derived from the HJ equation (1) can be represented by a neural network architecture. Specifically, the corresponding entropy solution $u(x,t) \equiv \nabla_x f(x,t)$ to the one-dimensional conservation law (19) can be represented using a neural network architecture with an argmax based activation function, i.e.,

$$\nabla_x f(x,t) = p_j, \text{ where } j \in \underset{i \in \{1,\ldots,m\}}{\arg\max} \{\langle p_i, x \rangle - t\theta_i - \gamma_i\}. \tag{20}$$

The structure of this network is shown in Fig. 4. When more than one maximizer exist in the optimization problem above, one can choose any maximizer j and define the value to be p_j. We now prove that the function $\nabla_x f$ given by the neural network (20) is indeed the entropy solution to the one-dimensional conservation law (19) with flux function H and initial data ∇J, where H and J are defined by Eqs. (14) and (10), respectively.

Proposition 3.1 *Consider the one-dimensional case, i.e., $n = 1$. Suppose the parameters $\{(p_i, \theta_i, \gamma_i)\}_{i=1}^m \subset \mathbb{R} \times \mathbb{R} \times \mathbb{R}$ satisfy assumptions (A1)–(A3), and let $u := \nabla_x f$ be the neural network defined in Eq. (20) with these parameters. Let J and H be the functions defined in Eqs. (10) and (14), respectively, and let $\tilde{H} : \mathbb{R} \to \mathbb{R}$ be a locally Lipschitz continuous function. Then, the following two statements hold.*

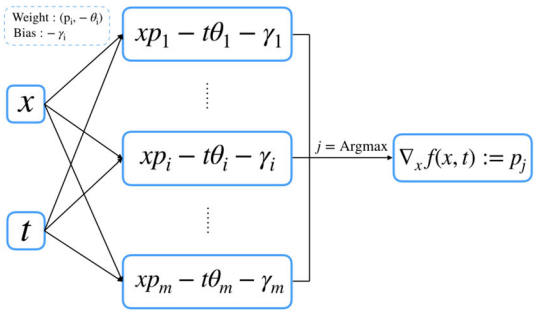

Fig. 4 Illustration of the structure of the neural network (20) that can represent the entropy solution to one-dimensional conservation laws

(i) The neural network u is the entropy solution to the conservation law

$$\begin{cases} \dfrac{\partial u}{\partial t}(x,t) + \nabla_x H(u(x,t)) = 0 & \text{in } \mathbb{R} \times (0,+\infty), \\ u(x,0) = \nabla J(x) & \text{in } \mathbb{R}. \end{cases} \tag{21}$$

(ii) The neural network u is the entropy solution to the conservation law

$$\begin{cases} \dfrac{\partial u}{\partial t}(x,t) + \nabla_x \tilde{H}(u(x,t)) = 0 & \text{in } \mathbb{R} \times (0,+\infty), \\ u(x,0) = \nabla J(x) & \text{in } \mathbb{R}, \end{cases} \tag{22}$$

if and only if there exists a constant $C \in \mathbb{R}$ *such that* $\tilde{H}(p_i) = H(p_i) + C$ *for every* $i \in \{1,\dots,m\}$ *and* $\tilde{H}(p) \geqslant H(p) + C$ *for any* $p \in \operatorname{conv}\{p_i\}_{i=1}^m$.

Proof See "Appendix D" for the proof. □

Example 4 Here, we give one example related to Example 1. Consider $J^{\text{true}}(x) = |x|$ and $H^{\text{true}}(p) = -\frac{p^2}{2}$ for every $x, p \in \mathbb{R}$. The entropy solution u to the corresponding one dimensional conservation law is given by

$$u(x,t) = \begin{cases} 1 & \text{if } \ x > 0, \\ -1 & \text{if } \ x < 0. \end{cases}$$

This solution u can be represented using the neural network in Fig. 4 with $m = 2$, $p_1 = 1$, $p_2 = -1$, $\theta_1 = \theta_2 = -\frac{1}{2}$ and $\gamma_1 = \gamma_2 = 0$. To be specific, we have

$$u(x) = p_j, \text{ where } j \in \operatorname*{arg\,max}_{i \in \{1,\dots,m\}} \{xp_i - t\theta_i - \gamma_i\}.$$

The initial data J and Hamiltonian H defined in Eqs. (10) and (14) are given by

$$J(x) = |x| \text{ for every } x \in \mathbb{R};$$

$$H(p) = \begin{cases} -\frac{1}{2} & p \in [-1,1], \\ +\infty & \text{otherwise.} \end{cases}$$

By Proposition 3.1, u solves the one-dimensional conservation law (22) if and only if there exists some constant $C \in \mathbb{R}$ such that $\tilde{H}(\pm 1) = -\frac{1}{2} + C$ and $\tilde{H}(p) \geqslant -\frac{1}{2} + C$ for every $p \in (-1, 1)$. Note that H^{true} is one candidate satisfying these constraints.

4 Numerical experiments

4.1 First-order Hamilton–Jacobi equations

In this subsection, we present several numerical experiments to test the effectiveness of the Adam optimizer using our proposed architecture (depicted in Fig. 1) for solving some inverse problems. We focus on the following inverse problem: We are given data samples from a function $S\colon \mathbb{R}^n \times [0, +\infty) \to \mathbb{R}$ that is the viscosity solution to an HJ equation (1) with unknown convex initial data J and Hamiltonian H, which only depends on $\nabla_{\boldsymbol{x}} S(\boldsymbol{x}, t)$. Our aim is to recover the convex initial data J. We propose to learn the neural network using machine learning techniques to recover the convex initial data J. We shall see that this approach also provides partial information on the Hamiltonian H.

Specifically, given data samples $\{(\boldsymbol{x}_j, t_j, S(\boldsymbol{x}_j, t_j))\}_{j=1}^N$, where $\{(\boldsymbol{x}_j, t_j)\}_{j=1}^N \subset \mathbb{R}^n \times [0, +\infty)$, we train the neural network f with structure in Fig. 1 using the mean square loss function defined by

$$l(\{(\boldsymbol{p}_i, \theta_i, \gamma_i)\}_{i=1}^m) = \frac{1}{N} \sum_{j=1}^{N} |f(\boldsymbol{x}_j, t_j; \{(\boldsymbol{p}_i, \theta_i, \gamma_i)\}_{i=1}^m) - S(\boldsymbol{x}_j, t_j)|^2.$$

The training problem is formulated as

$$\underset{\{(\boldsymbol{p}_i, \theta_i, \gamma_i)\}_{i=1}^m \subset \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}}{\arg\min} \quad l(\{(\boldsymbol{p}_i, \theta_i, \gamma_i)\}_{i=1}^m). \tag{23}$$

After training, we approximate the initial condition in the HJ equation, denoted by $\tilde{J}$, by evaluating the trained neural network at $t = 0$. That is, we approximate the initial condition by

$$\tilde{J} := f(\cdot, 0). \tag{24}$$

In addition, we obtain partial information of the Hamiltonian H using the parameters in the trained neural network via the following procedure. We first detect the effective neurons of the network, which we define to be the affine functions $\{\langle \boldsymbol{p}_i, \boldsymbol{x} \rangle - t\theta_i - \gamma_i\}$ that contribute to the pointwise maximum in the neural network f (see Eq. (8)). We then denote by L the set of indices that correspond to the parameters of the effective neurons, i.e.,

$$L := \bigcup_{\boldsymbol{x} \in \mathbb{R}^n,\, t \geq 0} \underset{i \in \{1, \ldots, m\}}{\arg\max} \{\langle \boldsymbol{p}_i, \boldsymbol{x} \rangle - t\theta_i - \gamma_i\},$$

and we finally use each effective parameter $(\boldsymbol{p}_l, \theta_l)$ for $l \in L$ to approximate the point $(\boldsymbol{p}_l, H(\boldsymbol{p}_l))$ on the graph of the Hamiltonian. In practice, we approximate the set L using a large number of points $(\boldsymbol{x}, t)$ sampled in the domain $\mathbb{R}^n \times [0, +\infty)$.

4.1.1 Randomly generalized piecewise affine H and J

In this subsection, we randomly select m parameters $\boldsymbol{p}_i^{true}$ in $[-1,1)^n$ for $i \in \{1,\dots,m\}$, and define θ_i^{true} and γ_i^{true} as follows

Case 1. $\theta_i^{true} = -\|\boldsymbol{p}_i^{true}\|_2$ and $\gamma_i^{true} = 0$ for $i \in \{1,\dots,m\}$.
Case 2. $\theta_i^{true} = -\|\boldsymbol{p}_i^{true}\|_2$ and $\gamma_i^{true} = \frac{1}{2}\|\boldsymbol{p}_i^{true}\|_2^2$ for $i \in \{1,\dots,m\}$.
Case 3. $\theta_i^{true} = -\frac{1}{2}\|\boldsymbol{p}_i^{true}\|_2^2$ and $\gamma_i^{true} = 0$ for $i \in \{1,\dots,m\}$.
Case 4. $\theta_i^{true} = -\frac{1}{2}\|\boldsymbol{p}_i^{true}\|_2^2$ and $\gamma_i^{true} = \frac{1}{2}\|\boldsymbol{p}_i^{true}\|_2^2$ for $i \in \{1,\dots,m\}$.

Define the function S as

$$S(\boldsymbol{x},t) := \max_{i\in\{1,\dots,m\}} \{\langle \boldsymbol{p}_i^{true}, \boldsymbol{x}\rangle - t\theta_i^{true} - \gamma_i^{true}\}.$$

By Theorem 3.1, this function S is a viscosity solution to the HJ equations whose Hamiltonian and initial function are the piecewise affine functions defined in Eqs. (14) and (10), respectively. In other words, S solves the HJ equation with initial data J satisfying

$$\begin{aligned} J(\boldsymbol{x}) &:= \max_{i\in\{1,\dots,m\}} \left\langle \boldsymbol{p}_i^{true}, \boldsymbol{x}\right\rangle, \quad \text{for Case 1 and 3;} \\ J(\boldsymbol{x}) &:= \max_{i\in\{1,\dots,m\}} \left\{\left\langle \boldsymbol{p}_i^{true}, \boldsymbol{x}\right\rangle - \frac{1}{2}\|\boldsymbol{p}_i^{true}\|^2\right\}, \quad \text{for Case 2 and 4,} \end{aligned} \tag{25}$$

and Hamiltonian H satisfying

$$H(\boldsymbol{p}) := \begin{cases} -\max\limits_{\boldsymbol{\alpha}\in\mathcal{A}(\boldsymbol{p})} \left\{\sum\limits_{i=1}^m \alpha_i \|\boldsymbol{p}_i^{true}\|_2\right\}, & \text{if } \boldsymbol{p} \in \operatorname{dom} J^*, \\ +\infty, & \text{otherwise,} \end{cases} \quad \text{for Case 1 and 2;}$$

$$H(\boldsymbol{p}) := \begin{cases} -\frac{1}{2}\max\limits_{\boldsymbol{\alpha}\in\mathcal{A}(\boldsymbol{p})} \left\{\sum\limits_{i=1}^m \alpha_i \|\boldsymbol{p}_i^{true}\|_2^2\right\} & \text{if } \boldsymbol{p} \in \operatorname{dom} J^*, \\ +\infty & \text{otherwise,} \end{cases} \quad \text{for Case 3 and 4,}$$

where $\mathcal{A}(\boldsymbol{p})$ is the set of maximizers of the corresponding maximization problem in Eq. (25). Specifically, if we construct a neural network f as shown in Fig. 1 with the underlying parameters $\{(\boldsymbol{p}_i^{true}, \theta_i^{true}, \gamma_i^{true})\}_{i=1}^m$, then the function given by the neural network is exactly the same as the function S. In other words, $\{(\boldsymbol{p}_i^{true}, \theta_i^{true}, \gamma_i^{true})\}_{i=1}^m$ is a global minimizer for the training problem (23) with the global minimal loss value equal to zero.

Now, we train the neural network f with training data $\{(\boldsymbol{x}_j, t_j, S(\boldsymbol{x}_j, t_j))\}_{j=1}^N$, where the points $\{(\boldsymbol{x}_j, t_j)\}_{j=1}^N$ are randomly sampled in $\mathbb{R}^n \times [0,+\infty)$ with respect to the standard normal distribution for each $j \in \{1,\dots,N\}$. (We take the absolute value for t to make sure it is nonnegative.) Here and after, the number of training data points is $N = 20{,}000$. We run 60,000 descent steps using the Adam optimizer to train the neural network f. The parameters for the Adam optimizer are chosen to be $\beta_1 = 0.5$, $\beta_2 = 0.9$, the learning rate is 10^{-4} and the batch size is 500.

To measure the performance of the training process, we compute the relative mean square errors of the sorted parameters in the trained neural network, denoted by $\{(\boldsymbol{p}_i, \theta_i, \gamma_i)\}_{i=1}^m$, and the sorted underlying true parameters $\{(\boldsymbol{p}_i^{true}, \theta_i^{true}, \gamma_i^{true})\}_{i=1}^m$. To be specific, the errors are computed as follows

Table 2 Relative mean square errors of the parameters in the neural network f with 2 neurons in different cases and different dimensions averaged over 100 repeated experiments

# Case		Case 1	Case 2	Case 3	Case 4
Averaged Relative Errors of $\{p_i\}$	2D	4.10E−03	2.10E−03	3.84E−03	2.82E−03
	4D	1.41E−09	1.20E−09	1.38E−09	1.29E−09
	8D	1.14E−09	1.03E−09	1.09E−09	1.20E−09
	16D	1.14E−09	6.68E−03	1.23E−09	7.74E−03
	32D	1.49E−09	3.73E−01	1.46E−03	4.00E−01
Averaged Relative Errors of $\{\theta_i\}$	2D	4.82E−02	7.31E−02	1.17E−01	1.79E−01
	4D	3.47E−10	2.82E−10	1.15E−09	1.15E−09
	8D	1.47E−10	1.08E−10	2.10E−10	2.25E−10
	16D	5.44E−11	1.69E−03	4.75E−11	4.12E−03
	32D	3.61E−11	3.27E−01	6.42E−03	2.39E−01
Averaged Relative Errors of $\{\gamma_i\}$	2D	1.35E−02	1.01E−01	1.33E−02	9.24E−02
	4D	3.71E−10	1.24E−09	3.67E−10	1.10E−09
	8D	2.91E−10	1.74E−10	2.82E−10	2.01E−10
	16D	2.80E−10	2.08E−04	3.10E−10	3.20E−04
	32D	3.56E−10	1.88E−02	1.56E−01	3.62E−02

$$\text{relative mean square error of } \{\boldsymbol{p}_i\} = \frac{\sum_{i=1}^m \|\boldsymbol{p}_i - \boldsymbol{p}_i^{true}\|_2^2}{\sum_{i=1}^m \|\boldsymbol{p}_i^{true}\|_2^2},$$
$$\text{relative mean square error of } \{\theta_i\} = \frac{\sum_{i=1}^m |\theta_i - \theta_i^{true}|^2}{\sum_{i=1}^m |\theta_i^{true}|^2},$$
$$\text{relative mean square error of } \{\gamma_i\} = \frac{\sum_{i=1}^m |\gamma_i - \gamma_i^{true}|^2}{\sum_{i=1}^m |\gamma_i^{true}|^2}.$$

For the cases when the denominator $\sum_{i=1}^m |\gamma_i^{true}|^2$ is zero, such as Case 1 and Case 3, we measure the absolute mean square error $\frac{1}{m}\sum_{i=1}^m |\gamma_i - \gamma_i^{true}|^2$ instead.

We test Cases 1–4 on the neural networks with 2 and 4 neurons, i.e., we set $m = 2, 4$ and repeat the experiments 100 times. We then compute the relative mean square errors in each experiment and take the average. The averaged relative mean square errors are shown in Tables 2 and 3, respectively. From the error tables, we observe that the training process performs pretty well and gives errors below 10^{-8} in some cases when $m = 2$. However, for the case when $m = 4$, we do not obtain the global minimizers and the error is above 10^{-3}. Therefore, there is no guarantee for the performance of the Adam optimizer in this training problem and it may be related to the complexity of the solution S to the underlying HJ equation.

4.1.2 Quadratic Hamiltonians

In this subsection, we consider two inverse problems of first-order HJ equations whose Hamiltonians and initial data are defined as follows:

1. $H(\boldsymbol{p}) = -\frac{1}{2}\|\boldsymbol{p}\|_2^2$ and $J(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$ for $\boldsymbol{p}, \boldsymbol{x} \in \mathbb{R}^n$.
2. $H(\boldsymbol{p}) = \frac{1}{2}\|\boldsymbol{p}\|_2^2$ and $J(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$ for $\boldsymbol{p}, \boldsymbol{x} \in \mathbb{R}^n$.

The solution to each of the two corresponding HJ equations can be represented using the Hopf formula [70] and reads

Table 3 Relative mean square errors of the parameters in the neural network f with 4 neurons in different cases and different dimensions averaged over 100 repeated experiments

# Case		Case 1	Case 2	Case 3	Case 4
Averaged Relative Errors of $\{p_i\}$	2D	3.12E−01	2.21E−01	2.85E−01	2.14E−01
	4D	7.82E−02	6.12E−02	7.92E−02	4.30E−02
	8D	2.62E−02	4.31E−03	4.02E−02	7.82E−03
	16D	2.88E−02	3.64E−02	4.35E−02	1.73E−02
	32D	1.42E−02	3.72E−01	1.42E−01	5.04E−01
Averaged Relative Errors of $\{\theta_i\}$	2D	2.59E−01	3.68E−01	4.82E−01	1.34E+00
	4D	6.07E−02	8.37E−02	9.47E−02	1.23E−01
	8D	1.04E−02	8.48E−03	1.41E−02	1.31E−02
	16D	2.66E−03	2.53E−02	7.80E−03	1.90E−02
	32D	8.09E−04	4.41E−01	1.81E−02	3.66E−01
Averaged Relative Errors of $\{\gamma_i\}$	2D	1.01E−02	3.19E−01	1.51E−02	2.65E−01
	4D	6.72E−03	1.79E−02	1.03E−02	1.30E−02
	8D	3.22E−03	2.34E−03	3.93E−03	2.65E−03
	16D	9.48E−03	3.70E−03	1.92E−02	1.94E−03
	32D	1.33E−02	5.35E−02	4.73E−01	1.17E−01

1. $S(\boldsymbol{x}, t) = \|\boldsymbol{x}\|_1 + \frac{nt}{2}$ for $\boldsymbol{x} \in \mathbb{R}^n$ and $t \geqslant 0$.
2. $S(\boldsymbol{x}, t) = \sum_{i:|x_i| \geqslant t} \left(|x_i| - \frac{t}{2}\right) + \sum_{i:|x_i|<t} \frac{x_i^2}{2t}$, where $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$ and $t \geqslant 0$.

We train the neural network f using the same procedure as in the previous subsection and obtain the function $\tilde{J}$ (see Eq. (24)) and the parameters $\{(\boldsymbol{p}_l, \theta_l)\}_{l \in L}$ associated with the effective neurons. We compute the relative mean square error of $\tilde{J}$ and $\{(\boldsymbol{p}_l, \theta_l)\}_{l \in L}$ as follows:

$$\text{relative error of } \tilde{J} := \frac{\sum_{j=1}^{N^{test}} |\tilde{J}(\boldsymbol{x}_i^{test}) - J(\boldsymbol{x}_i^{test})|^2}{\sum_{j=1}^{N^{test}} |J(\boldsymbol{x}_i^{test})|^2},$$

$$\text{relative error of } \{(\boldsymbol{p}_l, \theta_l)\}_l := \frac{\sum_{l \in L} |\theta_l - H(\boldsymbol{p}_l)|^2}{\sum_{l \in L} |H(\boldsymbol{p}_l)|^2},$$

where $\{\boldsymbol{x}_i^{test}\}$ are randomly sampled with respect to the standard normal distribution in $\mathbb{R}^n$ and there are in total $N^{test} = 2{,}000$ testing data points. We repeat the experiments 100 times. The corresponding averaged errors in the two examples are listed in Tables 4 and 5, respectively.

In the first example, we have $H(\boldsymbol{p}) = -\frac{1}{2}\|\boldsymbol{p}\|_2^2$ and $J(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$. According to Theorem 3.1, the solution S can be represented without error by the neural network in Fig. 1 with parameters

$$\left\{(\boldsymbol{p}, \theta, \gamma) \in \mathbb{R}^n \times \mathbb{R} \times \mathbb{R} : \ \boldsymbol{p}(i) \in \{\pm 1\}, \ \text{for } i \in \{1, \ldots, n\}, \ \theta = \frac{n}{2}, \ \gamma = 0\right\}, \tag{26}$$

where $\boldsymbol{p}(i)$ denotes the i^{th} entry of the vector $\boldsymbol{p}$. In other words, the global minimal loss value in the training problem is theoretically guaranteed to be zero. From the numerical errors in Table 4, we observe that in low dimension such as 1D and 2D, the errors of the initial function are small. However, in most cases, the errors of the parameters are pretty large. In the case of n dimension, the viscosity solution can be represented using the 2^n parameters in Eq. (26). However, the number of effective neurons are larger than 2^n in all

Table 4 Relative mean square errors of $\tilde{J}$ and $\{(p_l, \theta_l)\}$ for the inverse problems of the first-order HJ equations in different dimensions with $J = \|\cdot\|_1$ and $H = -\frac{1}{2}\|\cdot\|_2^2$, averaged over 100 repeated experiments

# Neurons		64	128	256	512	1024
Averaged Relative Errors of $\tilde{J}$	1D	2.29E−07	2.20E−07	2.12E−07	2.14E−07	1.82E−07
	2D	1.49E−06	1.27E−06	1.16E−06	1.01E−06	9.25E−07
	4D	6.27E−04	1.81E−04	5.93E−05	1.69E−06	3.44E−07
	8D	1.27E−02	1.10E−02	1.03E−02	9.92E−03	9.73E−03
	16D	5.69E−02	5.83E−02	5.96E−02	5.99E−02	6.01E−02
Averaged Relative Errors of $\{(p_l, \theta_l)\}$	1D	2.58E−01	1.29E−01	7.05E−02	3.56E−02	1.72E−02
	2D	4.77E−02	3.28E−02	2.03E−02	1.03E−02	6.53E−03
	4D	9.36E−03	4.09E−03	1.58E−03	5.31E−04	1.73E−04
	8D	3.75E−02	3.39E−02	3.25E−02	2.78E−02	2.60E−02
	16D	5.30E−01	5.40E−01	5.43E−01	5.43E−01	5.42E−01
Averaged Number of Effective Neurons	1D	4.45	4.37	4.18	3.92	3.55
	2D	8.84	8.59	7.87	7.1	6.3
	4D	20.04	20.62	19.52	18.3	17.06
	8D	36.97	43.91	47.84	49.19	50.03
	16D	48.2	59.53	64.85	65.79	64.84

Table 5 Relative mean square errors of $\tilde{J}$ and $\{(p_l, \theta_l)\}$ for the inverse problems of the first-order HJ equations in different dimensions with $J = \|\cdot\|_1$ and $H = \|\cdot\|_2^2/2$, averaged over 100 repeated experiments

# Neurons		64	128	256	512	1024
Averaged Relative Errors of $\tilde{J}$	1D	5.23E−08	2.45E−08	1.96E−08	1.77E−08	1.77E−08
	2D	1.75E−05	1.67E−05	1.77E−05	1.85E−05	1.91E−05
	4D	5.82E−04	4.94E−04	5.28E−04	5.76E−04	6.16E−04
	8D	1.54E−02	1.40E−02	1.35E−02	1.33E−02	1.32E−02
	16D	4.19E−02	4.33E−02	4.43E−02	4.46E−02	4.49E−02
Averaged Relative Errors of $\{(p_l, \theta_l)\}$	1D	3.25E−02	1.93E−02	1.24E−02	5.62E−03	2.92E−03
	2D	8.30E−03	7.08E−03	5.78E−03	4.25E−03	3.47E−03
	4D	2.41E−02	2.41E−02	2.51E−02	2.65E−02	2.82E−02
	8D	7.33E−02	7.32E−02	7.25E−02	7.15E−02	7.08E−02
	16D	3.85E−01	3.90E−01	3.92E−01	3.92E−01	3.91E−01
Averaged Number of Effective Neurons	1D	20.26	26.94	32.26	36.02	38.61
	2D	32.74	48.05	65.7	84.87	99.83
	4D	46.69	72.3	103.71	147.41	198.27
	8D	55.55	82.04	95.46	90.82	82.5
	16D	61.51	99.63	119.95	118.89	109.1

cases, which also implies that the Adam optimizer does not find the global minimizers in this example.

In the second example, the solution S cannot be represented using our proposed neural network without error. Hence, the results describe the approximation of the solution S by the neural network. From Table 5, we observe that the errors become larger when the dimension increases. For this example, the number of effective neurons should be m where m is the number of neurons used in the architecture. Table 5 shows that the average number of effective neurons is below this optimal number. Therefore, this implies that the Adam optimizer does not find the global minimizers in this example either.

In conclusion, these numerical experiments suggest that recovering initial data from data samples using our proposed neural network architecture with the Adam optimizer is unsatisfactory for solving these inverse problems. In particular, Adam optimizer is not always able to find a global minimizer when the solution can be represented without error using our network architecture.

4.2 One-dimensional conservation laws

In this part, we show the representability of the neural network $\nabla_x f$ given in Fig. 4 and Eq. (20). Since the number of neurons is finite, the function $\nabla_x f$ only takes values in the finite set $\{p_i\}_{i=1}^m$. In other words, it can represent the entropy solution u to the PDE (19) without error only if u takes values in a finite set.

Here, we consider the following two examples

1. $H(p) = -\frac{1}{2}p^2$ and $J(x) = |x|$ for $p, x \in \mathbb{R}$. The initial condition u_0 is then given by

$$u_0(x) = \begin{cases} 1, & x > 0, \\ -1, & x < 0. \end{cases}$$

2. $H(p) = \frac{1}{2}p^2$ and $J(x) = |x|$ for $p, x \in \mathbb{R}$. Hence, the initial function u_0 is the same as in example 1.

In the first example, the entropy solution u only takes values in the finite set $\{\pm 1\}$, and it can be represented by the neural network $\nabla_x f$ without error by Prop. 3.1. However, in the second example, the solution u takes values in the infinite set $[-1, 1]$; hence, the neural network $\nabla_x f$ is only an approximation of the corresponding solution u.

To show the representability of the neural network, in each example, we choose the parameters $\{p_i\}_{i=1}^m$ to be the uniform grid points in $[-1, 1]$, i.e.,

$$p_i = -1 + \frac{2(i-1)}{m-1} \quad \text{for} \quad i \in \{1, \ldots, m\}.$$

We set $\theta_i = H(p_i)$ and $\gamma_i = J^*(p_i)$ for each $i \in \{1, \ldots, m\}$, where J^* is the Fenchel–Legendre transform of the antiderivative of the initial function u_0. Hence, in these two examples, γ_i equals for each i. Figures 5 and 6 show the neural network $\nabla_x f$ and the true entropy solution u in these two examples at time $t = 1$. As expected, the error in Fig. 5 for example 1 is negligible. For example 2, we consider neural networks with 32 and 128 neurons whose graphs are plotted in Figs. 6a and 6b, respectively. We observe in these figures that the error of the neural networks with the specific parameters decreases as the number of neurons increases. In conclusion, the neural network $\nabla_x f$ with the architecture in Fig. 4 can represent the solution to the one-dimensional conservation laws given in Eq. (19) pretty well. In fact, because of the discontinuity of the activation function, the proposed neural network $\nabla_x f$ has advantages in representing the discontinuity in solution such as shocks, but it requires more neurons when approximating non-constant smooth parts of the solution.

5 Conclusion

Summary of the proposed work In this paper, we have established novel mathematical connections between some classes of HJ PDEs with convex initial data and neural net-

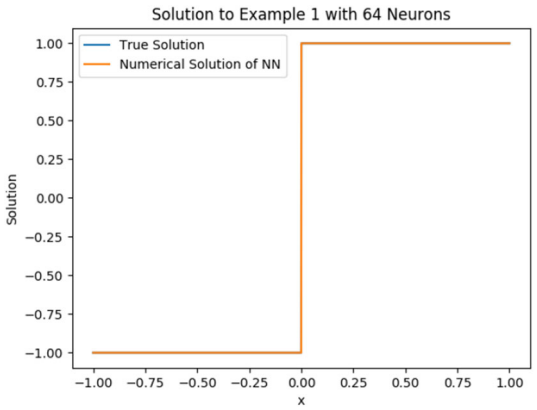

Fig. 5 Plot of the function represented by the neural network $\nabla_x f$ at time $t = 1$ with 64 neurons whose parameters are defined using H and J^* in example 1. The function given by the neural network is plotted in orange and the true solution is plotted in blue

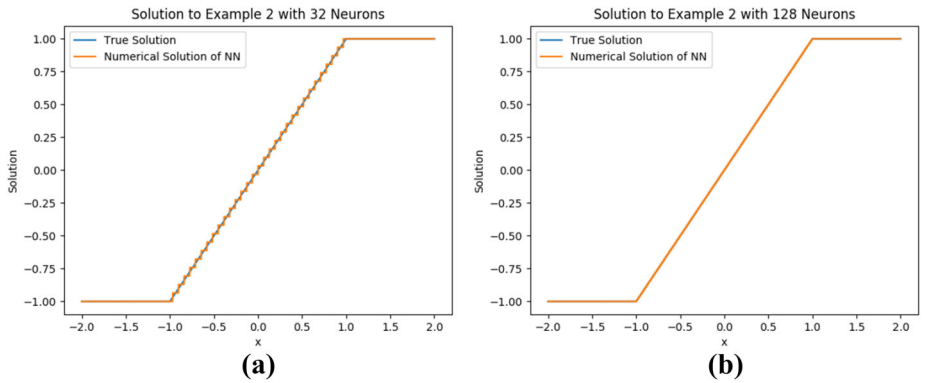

Fig. 6 Plot of the function represented by the neural network $\nabla_x f$ at time $t = 1$ with 32 and 128 neurons whose parameters are defined using H and J^* in example 2. The function given by the neural network is plotted in orange and the true solution is plotted in blue. The neural network with 32 neurons is shown on the left, while the neural network with 128 neurons is shown on the right

work architectures. Our main results give conditions under which for initial data which takes a particular form. These results do not rely on universal approximation properties of neural networks; rather, our results show that some neural networks correspond to representation formulas of solutions to HJ PDEs whose Hamiltonians and convex initial data are obtained from the parameters of the neural network. This means that some neural network architectures naturally encode the physics contained in some HJ PDEs satisfying the conditions in Theorem 3.1.

The first neural network architecture that we have proposed is depicted in Fig. 1. We have shown in Theorem 3.1 that under certain conditions on the parameters, this neural network architecture represents the viscosity solution of the HJ PDEs (16) for initial data which takes a particular form. The corresponding Hamiltonian and convex initial data can be recovered from the parameters of this neural network. As a corollary of this result for the one-dimensional case, we have proposed a second neural network architecture (depicted in Fig. 4) that represents the spatial gradient of the viscosity solution of the

HJ PDEs (1) (in one dimension), and we have shown in Prop. 3.1 that under appropriate conditions on the parameters, this neural network corresponds to entropy solutions of the conservation laws (22).

Let us emphasize that the neural network architecture depicted in Fig. 1 that represents solutions to the HJ PDEs (16) allows us to numerically evaluate these solutions in high dimension without using grids or numerical approximations. Our work also paves the way to leverage efficient technologies and hardware developed for neural networks to compute efficiently solutions to certain HJ PDEs.

We have also tested the performance of the state-of-the-art Adam optimizer using our proposed neural network architecture (depicted in Fig. 1) on some inverse problems. Our numerical experiments in Sect. 4 show that these problems cannot generally be solved with the Adam optimizer with high accuracy. These numerical results suggest further developments of efficient neural network training algorithms for solving inverse problems with our proposed neural network architectures.

Perspectives on other neural network architectures and HJ PDEs We now present extensions of the proposed architectures that are viable candidates for representing solutions of HJ PDEs.

First consider the following multi-time HJ PDE [12,27,39,105,119,125,132,141] which reads

$$\begin{cases} \dfrac{\partial S}{\partial t_j}(\boldsymbol{x}, t_1, \ldots, t_N) + H_j(\nabla_{\boldsymbol{x}} S(\boldsymbol{x}, t_1, \ldots, t_N)) \\ \quad = 0 \quad \text{for each } j \in \{1, \ldots, N\} & \text{in } \mathbb{R}^n \times (0, +\infty)^N, \\ S(\boldsymbol{x}, 0, \ldots, 0) = J(\boldsymbol{x}) & \text{in } \mathbb{R}^n. \end{cases} \tag{27}$$

A generalized Hopf formula [39,105,132] for this multi-time HJ equation is given by

$$S(\boldsymbol{x}, t_1, \ldots, t_N) = \left(\sum_{i=1}^{N} t_i H_i + J^* \right)^* (\boldsymbol{x}) = \sup_{\boldsymbol{p} \in \mathbb{R}^n} \left\{ \langle \boldsymbol{p}, \boldsymbol{x} \rangle - \sum_{j=1}^{N} t_j H_j(\boldsymbol{p}) - J^*(\boldsymbol{p}) \right\}, \tag{28}$$

for any $\boldsymbol{x} \in \mathbb{R}^n$ and $t_1, \ldots, t_N \geqslant 0$. Based on this formula, we propose a neural network architecture, depicted in Fig. 7, whose mathematical definition is given by

$$f(\boldsymbol{x}, t_1, \ldots, t_N; \{(\boldsymbol{p}_i, \theta_{i1}, \ldots, \theta_{iN}, \gamma_i)\}_{i=1}^m) = \max_{i \in \{1, \ldots, m\}} \left\{ \langle \boldsymbol{p}_i, \boldsymbol{x} \rangle - \sum_{j=1}^{N} t_j \theta_{ij} - \gamma_i \right\}, \tag{29}$$

where $\{(\boldsymbol{p}_i, \theta_{i1}, \ldots, \theta_{iN}, \gamma_i)\}_{i=1}^m \subset \mathbb{R}^n \times \mathbb{R}^N \times \mathbb{R}$ is the set of parameters. The generalized Hopf formula (28) suggests that the neural network architecture depicted in Fig. 7 is a good candidate for representing the solution to (27) under appropriate conditions on the parameters of the network.

As mentioned in [105], the multi-time HJ equation (27) may not have viscosity solutions. However, under suitable assumptions [12,27,39,119], the generalized Hopf formula (28) is a viscosity solution of the multi-time HJ equation. We intend to clarify the connections between the generalized Hopf formula, multi-time HJ PDEs, viscosity solutions, and general solutions in a future work.

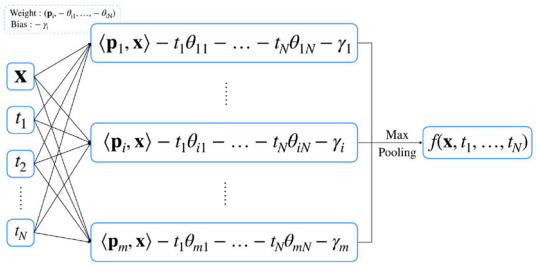

Fig. 7 Illustration of the structure of the neural network (29) that can represent solutions to some first-order multi-time HJ equations

In [38,39], it is shown that when the Hamiltonian H and the initial data J are both convex, and under appropriate assumptions, the solution S to the following HJ PDE

$$\begin{cases} \dfrac{\partial S}{\partial t}(\boldsymbol{x}, t) + H(\nabla_{\boldsymbol{x}} S(\boldsymbol{x}, t)) = 0 & \text{in } \mathbb{R}^n \times (0, +\infty), \\ S(\boldsymbol{x}, 0) = J(\boldsymbol{x}) & \text{in } \mathbb{R}^n, \end{cases}$$

is represented by the Hopf [70] and Lax–Oleinik formulas [47, Sect. 10.3.4]. These formulas read

$$\begin{aligned} S(\boldsymbol{x}, t) &= \max_{\boldsymbol{p} \in \mathbb{R}^n} \left\{ \langle \boldsymbol{p}, \boldsymbol{x} \rangle - J^*(\boldsymbol{p}) - tH(\boldsymbol{p}) \right\} && \text{(Hopf formula)} \\ &= \min_{\boldsymbol{u} \in \mathbb{R}^n} \left\{ J(\boldsymbol{u}) + tH^* \left(\frac{\boldsymbol{x} - \boldsymbol{u}}{t} \right) \right\}. && \text{(Lax–Oleinik formula)} \end{aligned}$$

Let $\boldsymbol{p}(\boldsymbol{x}, t)$ be the maximizer in the Hopf formula and $\boldsymbol{u}(\boldsymbol{x}, t)$ be the minimizer in the Lax–Oleinik formula. Then, they satisfy the following relation [38,39]

$$\boldsymbol{u}(\boldsymbol{x}, t) = \boldsymbol{x} - t \nabla H(\boldsymbol{p}(\boldsymbol{x}, t)).$$

Figure 8 depicts an architecture of a neural network that implements the formula above for the minimizer $\boldsymbol{u}(\boldsymbol{x}, t)$. In other words, we consider the ResNet-type neural network defined by

$$\boldsymbol{u}(\boldsymbol{x}, t) = \boldsymbol{x} - t \nabla H(\boldsymbol{p}_j), \text{ where } j \in \operatorname*{arg\,max}_{i \in \{1, \ldots, m\}} \left\{ \langle \boldsymbol{p}_i, \boldsymbol{x} \rangle - t\theta_i - \gamma_i \right\}. \tag{30}$$

Note that this proposed neural network suggests an interpretation of some ResNet architecture (for details on the ResNet architecture, see [66]) in terms of HJ PDEs. The activation functions of the proposed ResNet architecture are a composition of an argmax-based function and $t \nabla H$, where H is the Hamiltonian in the corresponding HJ equation. Moreover, when the time variable is fixed, the input $\boldsymbol{x}$ and the output $\boldsymbol{u}$ are in the same space $\mathbb{R}^n$; hence, one can chain the ResNet structure in Fig. 8 to obtain a deep neural network architecture by specifying a sequence of time variables $t_1, t_2, \ldots, t_N$. The deep neural network is given by

$$\boldsymbol{u}_k = \boldsymbol{u}_{k-1} - t_k \nabla H(\boldsymbol{p}_{j_k}^k), \quad \text{for each } k \in \{1, \ldots, N\}, \tag{31}$$

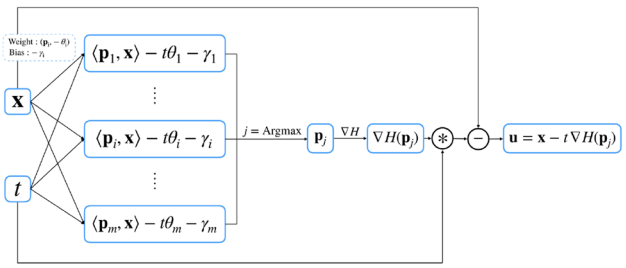

Fig. 8 Illustration of the structure of the ResNet-type neural network (30) that can represent the minimizer $\boldsymbol{u}$ in the Lax–Oleinik formula. Note that the activation function is defined using the gradient of the Hamiltonian H, i.e., ∇H

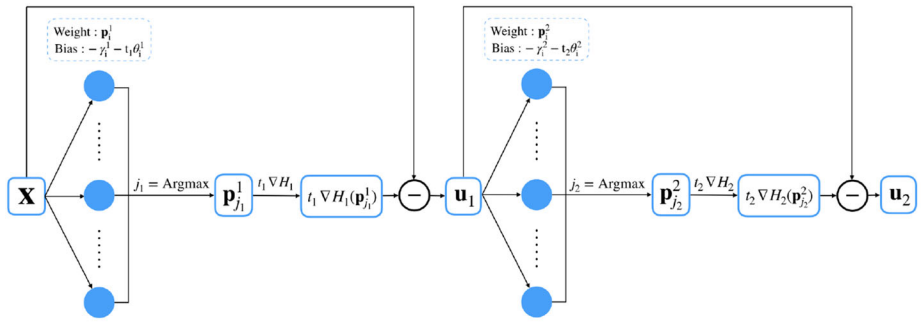

Fig. 9 Illustration of the structure of the ResNet-type deep neural network (31) that can represent the minimizers in the generalized Lax–Oleinik formula for the multi-time HJ PDEs. Note that the activation function in the k^{th} layer is defined using the gradient of one Hamiltonian H_k, i.e., ∇H_k. This figure only depicts two layers

where $\boldsymbol{u}_0 = \boldsymbol{x}$ and $\boldsymbol{p}_{j_k}^k$ is the output of the argmax based activation function in the k^{th} layer. For the case when $N = 2$, an illustration of this deep ResNet architecture with two layers is shown in Fig. 9. In fact, this deep ResNet architecture can be formulated as follows

$$\boldsymbol{u}_N = \boldsymbol{x} - \sum_{k=1}^{N} t_k \nabla H(\boldsymbol{p}_{j_k}^k).$$

This formulation suggests that this architecture should also provide the minimizers of the generalized Lax–Oleinik formula for the multi-time HJ PDEs [39]. These ideas and perspectives will be presented in detail in a forthcoming paper.

Applications of these neural architectures that can represent viscosity solutions of certain HJ PDEs to certain optimal control problems will be presented elsewhere.

Conflict of interest
The authors declare that they have no conflict of interest.

A Proofs of lemmas in Section 3.1

A.1 Proof of Lemma 3.1

Proof of (i): The convex and lower semicontinuous function J^* satisfies Eq. (12) by [68, Prop. X.3.4.1]. It is also finite and continuous over its polytopal domain $\text{dom}\, J^* =$

conv $\left(\{\boldsymbol{p}_i\}_{i=1}^m\right)$ [133, Thms. 10.2 and 20.5], and moreover, the subdifferential $\partial J^*(\boldsymbol{p})$ is non-empty by [133, Thm. 23.10].

Proof of (ii): First, suppose the vector $(\alpha_1, \ldots, \alpha_m) \in \mathbb{R}^m$ satisfies the constraints (a)–(c). Since $\boldsymbol{x} \in \partial J^*(\boldsymbol{p})$, there holds $J^*(\boldsymbol{p}) = \langle \boldsymbol{p}, \boldsymbol{x} \rangle - J(\boldsymbol{x})$ [68, Cor. X.1.4.4], and using the definition of the set $I_{\boldsymbol{x}}$ (11) and constraints (a)–(c) we deduce that

$$\begin{aligned} J^*(\boldsymbol{p}) &= \langle \boldsymbol{p}, \boldsymbol{x} \rangle - J(\boldsymbol{x}) = \langle \boldsymbol{p}, \boldsymbol{x} \rangle - \sum_{i \in I_{\boldsymbol{x}}} \alpha_i J(\boldsymbol{x}) \\ &= \langle \boldsymbol{p}, \boldsymbol{x} \rangle - \sum_{i \in I_{\boldsymbol{x}}} \alpha_i (\langle \boldsymbol{p}_i, \boldsymbol{x} \rangle - \gamma_i) \\ &= \left\langle \boldsymbol{p} - \sum_{i \in I_{\boldsymbol{x}}} \alpha_i \boldsymbol{p}_i, \boldsymbol{x} \right\rangle + \sum_{i \in I_{\boldsymbol{x}}} \alpha_i \gamma_i = \sum_{i=1}^m \alpha_i \gamma_i. \end{aligned}$$

Therefore, $(\alpha_1, \ldots, \alpha_m)$ is a minimizer in Eq. (12). Second, let $(\alpha_1, \ldots, \alpha_m)$ be a minimizer in Eq. (12). Then, (a)–(b) follow directly from the constraints in Eq. (12). A similar argument as above yields

$$J(\boldsymbol{x}) = \langle \boldsymbol{p}, \boldsymbol{x} \rangle - J^*(\boldsymbol{p}) = \left\langle \sum_{i=1}^m \alpha_i \boldsymbol{p}_i, \boldsymbol{x} \right\rangle - \sum_{i=1}^m \alpha_i \gamma_i = \sum_{i=1}^m \alpha_i \left(\langle \boldsymbol{p}_i, \boldsymbol{x} \rangle - \gamma_i \right).$$

But $J(\boldsymbol{x}) = \max_{i \in \{1,\ldots,m\}} \{\langle \boldsymbol{p}_i, \boldsymbol{x} \rangle - \gamma_i\}$ by definition, and so there holds $\alpha_i = 0$ whenever $J(\boldsymbol{x}) \neq \langle \boldsymbol{p}_i, \boldsymbol{x} \rangle - \gamma_i$. In other words, $\alpha_i = 0$ whenever $i \notin I_{\boldsymbol{x}}$.

Proof of (iii): Let $(\beta_1, \ldots, \beta_m) \in \Lambda_m$ satisfy $\sum_{i=1}^m \beta_i \boldsymbol{p}_i = \boldsymbol{p}_k$. By assumption (A2), we have $\gamma_k = g(\boldsymbol{p}_k)$ with g convex, and hence, Jensen's inequality yields

$$\sum_{i=1}^m \delta_{ik} \gamma_i = \gamma_k = g(\boldsymbol{p}_k) = g\left(\sum_{i=1}^m \beta_i \boldsymbol{p}_i \right) \leqslant \sum_{i=1}^m \beta_i g(\boldsymbol{p}_i) = \sum_{i=1}^m \beta_i \gamma_i.$$

Therefore, the vector $(\delta_{1k}, \ldots, \delta_{mk})$ is a minimizer in Eq. (12) at the point $\boldsymbol{p}_k$, and $J^*(\boldsymbol{p}_k) = \gamma_k$ follows.

A.2 Proof of Lemma 3.2

Proof of (i): Let $\boldsymbol{p} \in \operatorname{dom} J^*$. The set $\mathcal{A}(\boldsymbol{p}) \subseteq \Lambda_m$ is non-empty and bounded by Lemma 3.1(i), and it is closed since $\mathcal{A}(\boldsymbol{p})$ is the solution set to the linear programming problem (12). Hence, $\mathcal{A}(\boldsymbol{p})$ is compact. As a result, we immediately have that $H(\boldsymbol{p}) < +\infty$. Moreover, for each $(\alpha_1, \ldots, \alpha_m) \in \mathcal{A}(\boldsymbol{p})$ there holds

$$-\infty < \min_{i=\{1,\ldots,m\}} \theta_i \leqslant \sum_{i=1}^m \alpha_i \theta_i \leqslant \max_{i=\{1,\ldots,m\}} \theta_i < +\infty$$

from which we conclude that H is a bounded function on $\operatorname{dom} J^*$. Since the target function in the minimization problem (14) is continuous, existence of a minimizer follows by compactness of $\mathcal{A}(\boldsymbol{p})$.

Proof of (ii): We have already shown in the proof of (i) that the restriction of H to $\operatorname{dom} J^*$ is bounded, and so it remains to prove its continuity. For any $\boldsymbol{p} \in \operatorname{dom} J^*$, we

have that $(\alpha_1, \ldots, \alpha_m) \in \mathcal{A}(\boldsymbol{p})$ if and only if $(\alpha_1, \ldots, \alpha_m) \in \Lambda_m$, $\sum_{i=1}^m \alpha_i \boldsymbol{p}_i = \boldsymbol{p}$, and $\sum_{i=1}^m \alpha_i \gamma_i = J^*(\boldsymbol{p})$. As a result, we have

$$H(\boldsymbol{p}) = \min \left\{ \sum_{i=1}^m \alpha_i \theta_i : \ (\alpha_1, \ldots, \alpha_m) \in \Lambda_m, \ \sum_{i=1}^m \alpha_i \boldsymbol{p}_i = \boldsymbol{p}, \ \sum_{i=1}^m \alpha_i \gamma_i = J^*(\boldsymbol{p}) \right\}. \quad (32)$$

Define the function $h\colon \ \mathbb{R}^{n+1} \to \mathbb{R} \cup \{+\infty\}$ by

$$h(\boldsymbol{p}, r) := \min \left\{ \sum_{i=1}^m \alpha_i \theta_i : (\alpha_1, \ldots, \alpha_m) \in \Lambda_m, \ \sum_{i=1}^m \alpha_i \boldsymbol{p}_i = \boldsymbol{p}, \ \sum_{i=1}^m \alpha_i \gamma_i = r \right\}, \quad (33)$$

for any $\boldsymbol{p} \in \mathbb{R}^n$ and $r \in \mathbb{R}$. Using the same argument as in the proof of Lemma 3.1(i), we conclude that h is a convex lower semicontinuous function, and in fact continuous over its domain $\operatorname{dom} h = \operatorname{conv} \{(\boldsymbol{p}_i, \gamma_i)\}_{i=1}^m$. Comparing Eq. (32) and the definition of h in (33), we deduce that $H(\boldsymbol{p}) = h(\boldsymbol{p}, J^*(\boldsymbol{p}))$ for any $\boldsymbol{p} \in \operatorname{dom} J^*$. Continuity of H in $\operatorname{dom} J^*$ then follows from the continuity of h and J^* in their own domains.

Proof of (iii): Let $k \in \{1, \ldots, m\}$. On the one hand, Lemma 3.1(iii) implies $(\delta_{1k}, \ldots, \delta_{mk}) \in \mathcal{A}(\boldsymbol{p}_k)$, so that

$$H(\boldsymbol{p}_k) \leqslant \sum_{i=1}^m \delta_{ik} \theta_i = \theta_k. \quad (34)$$

On the other hand, let $(\alpha_1, \ldots, \alpha_m) \in \mathcal{A}(\boldsymbol{p}_k)$ be a vector different from $(\delta_{k1}, \ldots, \delta_{km})$. Then, $(\alpha_1, \ldots, \alpha_m) \in \Lambda_m$ satisfies $\sum_{i=1}^m \alpha_i \boldsymbol{p}_i = \boldsymbol{p}$, $\sum_{i=1}^m \alpha_i \gamma_i = J^*(\boldsymbol{p})$, and $\alpha_k < 1$. Define $(\beta_1, \ldots, \beta_m) \in \Lambda_m$ by

$$\beta_j := \begin{cases} \dfrac{\alpha_j}{1 - \alpha_k} & \text{if} \quad j \neq k, \\ 0 & \text{if} \quad j = k. \end{cases}$$

A straightforward computation using the properties of $(\alpha_1, \ldots, \alpha_m)$, Lemma 3.1(iii), and the definition of $(\beta_1, \ldots, \beta_m)$ yields

$$\begin{cases} (\beta_1, \ldots, \beta_m) \in \Lambda_m \quad \text{with } \beta_k = 0, \\ \displaystyle\sum_{i \neq k} \beta_i \boldsymbol{p}_i = \sum_{i \neq k} \frac{\alpha_i \boldsymbol{p}_i}{1 - \alpha_k} = \frac{\boldsymbol{p}_k - \alpha_k \boldsymbol{p}_k}{1 - \alpha_k} = \boldsymbol{p}_k, \\ \displaystyle\sum_{i \neq k} \beta_i \gamma_i = \sum_{i \neq k} \frac{\alpha_i \gamma_i}{1 - \alpha_k} = \frac{J^*(\boldsymbol{p}_k) - \alpha_k \gamma_k}{1 - \alpha_k} = \frac{\gamma_k - \alpha_k \gamma_k}{1 - \alpha_k} = \gamma_k. \end{cases}$$

In other words, Eq. (9) holds at index k, which, by assumption (A3), implies that $\sum_{i \neq k} \beta_i \theta_i > \theta_k$. As a result, we have

$$\sum_{i=1}^m \alpha_i \theta_i = \alpha_k \theta_k + (1 - \alpha_k) \sum_{i \neq k} \beta_i \theta_i > \alpha_k \theta_k + (1 - \alpha_k) \theta_k = \theta_k = \sum_{i=1}^m \delta_{ik} \theta_i.$$

Taken together with Eq. (34), we conclude that $(\delta_{1k}, \ldots, \delta_{mk})$ is the unique minimizer in (14), and hence, we obtain $H(\boldsymbol{p}_k) = \theta_k$.

B Proof of Theorem 3.1

To prove this theorem, we will use three lemmas whose statements and proofs are given in Sect. B.1, B.2, and B.3, respectively. The proof of Theorem 3.1 is given in Sect. B.4.

B.1 Statement and proof of Lemma B.1

Lemma B.1 *Suppose the parameters* $\{(\boldsymbol{p}_i, \theta_i, \gamma_i)\}_{i=1}^m \subset \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}$ *satisfy assumptions (A1)-(A3). Let* J *and* H *be the functions defined in Eqs. (10) and (14), respectively. Let* $\tilde{H}\colon \mathbb{R}^n \to \mathbb{R}$ *be a continuous function satisfying* $\tilde{H}(\boldsymbol{p}_i) = H(\boldsymbol{p}_i)$ *for each* $i \in \{1, \dots, m\}$ *and* $\tilde{H}(\boldsymbol{p}) \geqslant H(\boldsymbol{p})$ *for all* $\boldsymbol{p} \in \operatorname{dom} J^*$. *Then, the neural network* f *defined in Eq. (8) satisfies*

$$f(\boldsymbol{x}, t) := \max_{i\in\{1,\dots,m\}} \{\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - t\theta_i - \gamma_i\} = \sup_{\boldsymbol{p}\in\operatorname{dom} J^*} \left\{\langle \boldsymbol{p}, \boldsymbol{x}\rangle - t\tilde{H}(\boldsymbol{p}) - J^*(\boldsymbol{p})\right\}. \tag{35}$$

Proof Let $\boldsymbol{x} \in \mathbb{R}^n$ and $t \geqslant 0$. Since $\tilde{H}(\boldsymbol{p}) \geqslant H(\boldsymbol{p})$ for every $\boldsymbol{p} \in \operatorname{dom} J^*$, we get

$$\langle \boldsymbol{p}, \boldsymbol{x}\rangle - t\tilde{H}(\boldsymbol{p}) - J^*(\boldsymbol{p}) \leqslant \langle \boldsymbol{p}, \boldsymbol{x}\rangle - tH(\boldsymbol{p}) - J^*(\boldsymbol{p}). \tag{36}$$

Let $(\alpha_1, \dots, \alpha_m)$ be a minimizer in (14). By Eqs. (12), (13), and (14), we have

$$\boldsymbol{p} = \sum_{i=1}^m \alpha_i \boldsymbol{p}_i, \quad H(\boldsymbol{p}) = \sum_{i=1}^m \alpha_i \theta_i, \quad \text{and} \quad J^*(\boldsymbol{p}) = \sum_{i=1}^m \alpha_i \gamma_i. \tag{37}$$

Combining Eqs. (36), (37), and (8), we get

$$\begin{aligned}\langle \boldsymbol{p}, \boldsymbol{x}\rangle - t\tilde{H}(\boldsymbol{p}) - J^*(\boldsymbol{p}) &\leqslant \sum_{i=1}^m \alpha_i(\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - t\theta_i - \gamma_i)\\ &\leqslant \max_{i\in\{1,\dots,m\}} \{\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - t\theta_i - \gamma_i\} = f(\boldsymbol{x}, t),\end{aligned}$$

where the second inequality follows from the constraint $(\alpha_1, \dots, \alpha_m) \in \Lambda_m$. Since $\boldsymbol{p} \in \operatorname{dom} J^*$ is arbitrary, we obtain

$$\sup_{\boldsymbol{p}\in\operatorname{dom} J^*} \left\{\langle \boldsymbol{p}, \boldsymbol{x}\rangle - t\tilde{H}(\boldsymbol{p}) - J^*(\boldsymbol{p})\right\} \leqslant f(\boldsymbol{x}, t). \tag{38}$$

Now, by Lemmas 3.1(iii), 3.2(iii), and the assumptions on $\tilde{H}$, we have

$$\tilde{H}(\boldsymbol{p}_k) = H(\boldsymbol{p}_k) = \theta_k \quad \text{and} \quad J^*(\boldsymbol{p}_k) = \gamma_k,$$

for each $k \in \{1, \dots, m\}$. A straightforward computation yields

$$\begin{aligned} f(\boldsymbol{x}, t) &= \max_{i\in\{1,\dots,m\}} \{\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - t\theta_i - \gamma_i\}\\ &= \max_{i\in\{1,\dots,m\}} \left\{\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - t\tilde{H}(\boldsymbol{p}_i) - J^*(\boldsymbol{p}_i)\right\}\\ &\leqslant \sup_{\boldsymbol{p}\in\operatorname{dom} J^*} \left\{\langle \boldsymbol{p}, \boldsymbol{x}\rangle - t\tilde{H}(\boldsymbol{p}) - J^*(\boldsymbol{p})\right\}, \end{aligned} \tag{39}$$

where the inequality holds since $\boldsymbol{p}_i \in \operatorname{dom} J^*$ for every $i \in \{1, \dots, m\}$. The conclusion then follows from Eqs. (38) and (39). □

B.2 Statement and proof of Lemma B.2

Lemma B.2 *Suppose the parameters* $\{(\boldsymbol{p}_i, \theta_i, \gamma_i)\}_{i=1}^m \subset \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}$ *satisfy assumptions (A1)-(A3). For every* $k \in \{1, \ldots, m\}$, *there exist* $\boldsymbol{x} \in \mathbb{R}^n$ *and* $t > 0$ *such that* $f(\cdot, t)$ *is differentiable at* $\boldsymbol{x}$ *and* $\nabla_{\boldsymbol{x}} f(\boldsymbol{x}, t) = \boldsymbol{p}_k$.

Proof Since f is the supremum of a finite number of affine functions by definition (8), it is finite-valued and convex for $t \geqslant 0$. As a result, $\nabla_{\boldsymbol{x}} f(\boldsymbol{x}, t) = \boldsymbol{p}_k$ is equivalent to $\partial(f(\cdot, t))(\boldsymbol{x}) = \{\boldsymbol{p}_k\}$, and so it suffices to prove that $\partial(f(\cdot, t))(\boldsymbol{x}) = \{\boldsymbol{p}_k\}$ for some $\boldsymbol{x} \in \mathbb{R}^n$ and $t > 0$. To simplify the notation, we use $\partial_{\boldsymbol{x}} f(\boldsymbol{x}, t)$ to denote the subdifferential of $f(\cdot, t)$ at $\boldsymbol{x}$.

By [67, Thm. VI.4.4.2], the subdifferential of $f(\cdot, t)$ at $\boldsymbol{x}$ is the convex hull of the $\boldsymbol{p}_i$'s whose indices i's are maximizers in (8), that is,

$$\partial_{\boldsymbol{x}} f(\boldsymbol{x}, t) = \operatorname{co}\{\boldsymbol{p}_i : i \text{ is a maximizer in (8)}\}.$$

It suffices then to prove the existence of $\boldsymbol{x} \in \mathbb{R}^n$ and $t > 0$ such that

$$\langle \boldsymbol{p}_k, \boldsymbol{x}\rangle - t\theta_k - \gamma_k > \langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - t\theta_i - \gamma_i \quad \text{for every } i \neq k. \tag{40}$$

First, consider the case when there exists $\boldsymbol{x} \in \mathbb{R}^n$ such that $\langle \boldsymbol{p}_k, \boldsymbol{x}\rangle - \gamma_k > \langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - \gamma_i$ for every $i \neq k$. In that case, by continuity, there exists small $t > 0$ such that $\langle \boldsymbol{p}_k, \boldsymbol{x}\rangle - t\theta_k - \gamma_k > \langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - t\theta_i - \gamma_i$ for every $i \neq k$ and so (40) holds.

Now, consider the case when there does not exist $\boldsymbol{x} \in \mathbb{R}^n$ such that $\langle \boldsymbol{p}_k, \boldsymbol{x}\rangle - \gamma_k > \max_{i\neq k}\{\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - \gamma_i\}$. In other words, we assume

$$J(\boldsymbol{x}) = \max_{i\neq k}\{\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - \gamma_i\} \quad \text{for every } \boldsymbol{x} \in \mathbb{R}^n. \tag{41}$$

We apply Lemma 3.1(i) to the formula above and obtain

$$J^*(\boldsymbol{p}_k) = \min\left\{\sum_{i=1}^m \alpha_i\gamma_i : (\alpha_1, \ldots, \alpha_m) \in \Lambda_m,\ \sum_{i=1}^m \alpha_i\boldsymbol{p}_i = \boldsymbol{p}_k,\ \alpha_k = 0\right\}. \tag{42}$$

Let $\boldsymbol{x}_0 \in \partial J^*(\boldsymbol{p}_k)$. Denote by $I_{\boldsymbol{x}_0}$ the set of maximizers in Eq. (41) at the point $\boldsymbol{x}_0$, i.e.,

$$I_{\boldsymbol{x}_0} := \arg\max_{i\neq k}\{\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - \gamma_i\}. \tag{43}$$

Note that we have $k \notin I_{\boldsymbol{x}_0}$ by definition of $I_{\boldsymbol{x}_0}$. Define a function $h\colon \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ by

$$h(\boldsymbol{p}) := \begin{cases} \theta_i & \text{if } \boldsymbol{p} = \boldsymbol{p}_i \text{ and } i \in I_{\boldsymbol{x}_0}, \\ +\infty & \text{otherwise.} \end{cases} \tag{44}$$

Denote the convex lower semicontinuous envelope of h by $\overline{\operatorname{co}}\, h$. Since $\boldsymbol{x}_0 \in \partial J^*(\boldsymbol{p}_k)$, we can use [67, Thm. VI.4.4.2] and the definition of $I_{\boldsymbol{x}_0}$ and h in Eqs. (43) and (44) to deduce

$$\boldsymbol{p}_k \in \partial J(\boldsymbol{x}_0) = \operatorname{co}\{\boldsymbol{p}_i : i \in I_{\boldsymbol{x}_0}\} = \operatorname{dom} \overline{\operatorname{co}}\, h. \tag{45}$$

Hence, the point $\boldsymbol{p}_k$ is in the domain of the polytopal convex function $\overline{\text{co}}\, h$. Then, [133, Thm. 23.10] implies $\partial(\overline{\text{co}}\, h)(\boldsymbol{p}_k) \neq \emptyset$. Let $\boldsymbol{v}_0 \in \partial(\overline{\text{co}}\, h)(\boldsymbol{p}_k)$ and $\boldsymbol{x} = \boldsymbol{x}_0 + t\boldsymbol{v}_0$. It remains to choose suitable positive t such that (40) holds. Letting $\boldsymbol{x} = \boldsymbol{x}_0 + t\boldsymbol{v}_0$ in (40) yields

$$\begin{aligned}
&\langle \boldsymbol{p}_k, \boldsymbol{x}\rangle - t\theta_k - \gamma_k - (\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - t\theta_i - \gamma_i) \\
&\quad = \langle \boldsymbol{p}_k, \boldsymbol{x}_0 + t\boldsymbol{v}_0\rangle - t\theta_k - \gamma_k - (\langle \boldsymbol{p}_i, \boldsymbol{x}_0 + t\boldsymbol{v}_0\rangle - t\theta_i - \gamma_i) \\
&\quad = \langle \boldsymbol{p}_k, \boldsymbol{x}_0\rangle - \gamma_k - (\langle \boldsymbol{p}_i, \boldsymbol{x}_0\rangle - \gamma_i) + t(\theta_i - \theta_k - \langle \boldsymbol{p}_i - \boldsymbol{p}_k, \boldsymbol{v}_0\rangle).
\end{aligned} \tag{46}$$

Now, we consider two situations, the first when $i \notin I_{\boldsymbol{x}_0} \cup \{k\}$ and the second when $i \in I_{\boldsymbol{x}_0}$. It suffices to prove (40) hold in each case for small enough positive t.

If $i \notin I_{\boldsymbol{x}_0} \cup \{k\}$, then i is not a maximizer in Eq. (41) at the point $\boldsymbol{x}_0$. By (45), $\boldsymbol{p}_k$ is a convex combination of the set $\{\boldsymbol{p}_i : i \in I_{\boldsymbol{x}_0}\}$. In other words, there exists $(c_1, \ldots, c_m) \in \Lambda_m$ such that $\sum_{j=1}^m c_j \boldsymbol{p}_j = \boldsymbol{p}_k$ and $c_j = 0$ whenever $j \notin I_{\boldsymbol{x}_0}$. Taken together with assumption (A2) and Eqs. (10), (41), (43), we have

$$\begin{aligned}
J(\boldsymbol{x}_0) \geqslant \langle \boldsymbol{p}_k, \boldsymbol{x}_0\rangle - \gamma_k = \langle \boldsymbol{p}_k, \boldsymbol{x}_0\rangle - g(\boldsymbol{p}_k) = \left\langle \sum_{j\in I_{\boldsymbol{x}_0}} c_j \boldsymbol{p}_j, \boldsymbol{x}_0 \right\rangle - g\left(\sum_{j\in I_{\boldsymbol{x}_0}} c_j \boldsymbol{p}_j\right) \\
\geqslant \sum_{j\in I_{\boldsymbol{x}_0}} c_j(\langle \boldsymbol{p}_j, \boldsymbol{x}_0\rangle - g(\boldsymbol{p}_j)) = \sum_{j\in I_{\boldsymbol{x}_0}} c_j J(\boldsymbol{x}_0) = J(\boldsymbol{x}_0).
\end{aligned}$$

Thus, the inequalities become equalities in the equation above. As a result, we have

$$\langle \boldsymbol{p}_k, \boldsymbol{x}_0\rangle - \gamma_k = J(\boldsymbol{x}_0) > \langle \boldsymbol{p}_i, \boldsymbol{x}_0\rangle - \gamma_i,$$

where the inequality holds because $i \notin I_{\boldsymbol{x}_0} \cup \{k\}$ by assumption. This inequality implies that the constant $\langle \boldsymbol{p}_k, \boldsymbol{x}_0\rangle - \gamma_k - (\langle \boldsymbol{p}_i, \boldsymbol{x}_0\rangle - \gamma_i)$ is positive, and taken together with (46), we conclude that the inequality in (40) holds for $i \notin I_{\boldsymbol{x}_0} \cup \{k\}$ when t is small enough.

If $i \in I_{\boldsymbol{x}_0}$, then both i and k are maximizers in Eq. (10) at $\boldsymbol{x}_0$, and hence, we have

$$\langle \boldsymbol{p}_k, \boldsymbol{x}_0\rangle - \gamma_k = J(\boldsymbol{x}_0) = \langle \boldsymbol{p}_i, \boldsymbol{x}_0\rangle - \gamma_i. \tag{47}$$

Together with Eq. (46) and the definition of h in Eq. (44), we obtain

$$\begin{aligned}
\langle \boldsymbol{p}_k, \boldsymbol{x}\rangle - t\theta_k - \gamma_k - (\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - t\theta_i - \gamma_i) &= 0 + t(h(\boldsymbol{p}_i) - \theta_k - \langle \boldsymbol{p}_i - \boldsymbol{p}_k, \boldsymbol{v}_0\rangle) \\
&\geqslant t(\overline{\text{co}}\, h(\boldsymbol{p}_i) - \theta_k - \langle \boldsymbol{p}_i - \boldsymbol{p}_k, \boldsymbol{v}_0\rangle).
\end{aligned} \tag{48}$$

In addition, since $\boldsymbol{v}_0 \in \partial(\overline{\text{co}}\, h)(\boldsymbol{p}_k)$, we have

$$\overline{\text{co}}\, h(\boldsymbol{p}_i) \geqslant \overline{\text{co}}\, h(\boldsymbol{p}_k) + \langle \boldsymbol{p}_i - \boldsymbol{p}_k, \boldsymbol{v}_0\rangle. \tag{49}$$

Combining Eqs. (48) and (49), we obtain

$$\langle \boldsymbol{p}_k, \boldsymbol{x}\rangle - t\theta_k - \gamma_k - (\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - t\theta_i - \gamma_i) \geqslant t(\overline{\text{co}}\, h(\boldsymbol{p}_k) - \theta_k). \tag{50}$$

To prove the result, it suffices to show $\overline{\text{co}}\, h(\boldsymbol{p}_k) > \theta_k$. As $\boldsymbol{p}_k \in \overline{\text{co}}\, h$ (as shown before in Eq. (45)), then according to [68, Prop. X.1.5.4] we have

$$\overline{\text{co}}\, h(\boldsymbol{p}_k) = \sum_{j\in I_{\boldsymbol{x}_0}} \alpha_j h(\boldsymbol{p}_j) = \sum_{j\in I_{\boldsymbol{x}_0}} \alpha_j \theta_j, \tag{51}$$

for some $(\alpha_1, \ldots, \alpha_m) \in \Lambda_m$ satisfying $\boldsymbol{p}_k = \sum_{j=1}^m \alpha_j \boldsymbol{p}_j$ and $\alpha_j = 0$ whenever $j \notin I_{\boldsymbol{x}_0}$. Then, by Lemma 3.1(ii) $(\alpha_1, \ldots, \alpha_m)$ is a minimizer in Eq. (42), that is,

$$\gamma_k = J^*(\boldsymbol{p}_k) = \sum_{j=1}^m \alpha_j \gamma_j = \sum_{j \in I_{\boldsymbol{x}_0}} \alpha_i \gamma_i = \sum_{i \neq k} \alpha_i \gamma_i.$$

Hence, Eq. (9) holds for the index k. By assumption (A3), we have $\theta_k < \sum_{j \neq k} \alpha_j \theta_j$. Taken together with the fact that $\alpha_j = 0$ whenever $j \notin I_{\boldsymbol{x}_0}$ and Eq. (51), we find

$$\theta_k < \sum_{j \neq k} \alpha_j \theta_j = \sum_{j \in I_{\boldsymbol{x}_0}} \alpha_j \theta_j = \overline{\mathrm{co}}\, h(\boldsymbol{p}_k). \tag{52}$$

Hence, the right-hand side of Eq. (50) is strictly positive, and we conclude that $\langle \boldsymbol{p}_k, \boldsymbol{x} \rangle - t\theta_k - \gamma_k > \langle \boldsymbol{p}_i, \boldsymbol{x} \rangle - t\theta_i - \gamma_i$ for $t > 0$ if $i \in I_{\boldsymbol{x}_0}$.

Therefore, in this case, when $t > 0$ is small enough and $\boldsymbol{x}$ is chosen as above, we have $\langle \boldsymbol{p}_k, \boldsymbol{x} \rangle - t\theta_k - \gamma_k > \langle \boldsymbol{p}_i, \boldsymbol{x} \rangle - t\theta_i - \gamma_i$ for every $i \neq k$, and the proof is complete. □

B.3 Statement and proof of Lemma B.3

Lemma B.3 *Suppose the parameters* $\{(\boldsymbol{p}_i, \theta_i, \gamma_i)\}_{i=1}^m \subset \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}$ *satisfy assumptions (A1)-(A3). Define a function* $F : \mathbb{R}^{n+1} \to \mathbb{R} \cup \{+\infty\}$ *by*

$$F(\boldsymbol{p}, E^-) := \begin{cases} J^*(\boldsymbol{p}) & \text{if } E^- + H(\boldsymbol{p}) \leqslant 0, \\ +\infty & \text{otherwise,} \end{cases} \tag{53}$$

for all $\boldsymbol{p} \in \mathbb{R}^n$ *and* $E^- \in \mathbb{R}$. *Then, the convex envelope of* F *is given by*

$$\mathrm{co}\, F(\boldsymbol{p}, E^-) = \inf_{(c_1, \ldots, c_m) \in C(\boldsymbol{p}, E^-)} \sum_{i=1}^m c_i \gamma_i, \tag{54}$$

where the constraint set $C(\boldsymbol{p}, E^-)$ *is defined by*

$$C(\boldsymbol{p}, E^-) := \left\{ (c_1, \ldots, c_m) \in \Lambda_m : \sum_{i=1}^m c_i \boldsymbol{p}_i = \boldsymbol{p}, \ \sum_{i=1}^m c_i \theta_i \leqslant -E^- \right\}.$$

Proof First, we compute the convex hull of epi F, which we denote by co (epi F). Let $(\boldsymbol{p}, E^-, r) \in$ co (epi F), where $\boldsymbol{p} \in \mathbb{R}^n$ and $E^-, r \in \mathbb{R}$. Then there exist $k \in \mathbb{N}$, $(\beta_1, \ldots, \beta_k) \in \Lambda_k$ and $(\boldsymbol{q}_i, E_i^-, r_i) \in$ epi F for each $i \in \{1, \ldots, k\}$ such that $(\boldsymbol{p}, E^-, r) = \sum_{i=1}^k \beta_i (\boldsymbol{q}_i, E_i^-, r_i)$. By definition of F in Eq. (53), $(\boldsymbol{q}_i, E_i^-, r_i) \in$ epi F holds if and only if $\boldsymbol{q}_i \in$ dom J^*, $E_i^- + H(\boldsymbol{q}_i) \leqslant 0$ and $r_i \geqslant J^*(\boldsymbol{q}_i)$. In conclusion, we have

$$\begin{cases} (\beta_1, \ldots, \beta_k) \in \Lambda_k, \\ (\boldsymbol{p}, E^-, r) = \sum_{i=1}^k \beta_i (\boldsymbol{q}_i, E_i^-, r_i), \\ \boldsymbol{q}_1, \ldots, \boldsymbol{q}_k \in \mathrm{dom}\, J^*, \\ E_i^- + H(\boldsymbol{q}_i) \leqslant 0 \quad \text{for each } i \in \{1, \ldots, k\}, \\ r_i \geqslant J^*(\boldsymbol{q}_i) \quad \text{for each } i \in \{1, \ldots, k\}. \end{cases} \tag{55}$$

For each i, since we have $\boldsymbol{q}_i \in \operatorname{dom} J^*$, by Lemma 3.2(i) the minimization problem in (14) evaluated at $\boldsymbol{q}_i$ has at least one minimizer. Let $(\alpha_{i1}, \ldots, \alpha_{im})$ be such a minimizer. Using Eqs. (12), (14), and $(\alpha_{i1}, \ldots, \alpha_{im}) \in \Lambda_m$, we have

$$\sum_{j=1}^{m} \alpha_{ij}(1, \boldsymbol{p}_j, \theta_j, \gamma_j) = (1, \boldsymbol{q}_i, H(\boldsymbol{q}_i), J^*(\boldsymbol{q}_i)). \tag{56}$$

Define the real number $c_j := \sum_{i=1}^{k} \beta_i \alpha_{ij}$ for any $j \in \{1, \ldots, m\}$. Combining Eqs. (55) and (56), we get that $c_j \geqslant 0$ for any j and

$$\begin{aligned}
&\sum_{j=1}^{m} c_j(1, \boldsymbol{p}_j, \theta_j, \gamma_j) = \sum_{j=1}^{m} \sum_{i=1}^{k} \beta_i \alpha_{ij}(1, \boldsymbol{p}_j, \theta_j, \gamma_j) \\
&= \sum_{i=1}^{k} \beta_i \left(\sum_{j=1}^{m} \alpha_{ij}(1, \boldsymbol{p}_j, \theta_j, \gamma_j) \right) = \sum_{i=1}^{k} \beta_i (1, \boldsymbol{q}_i, H(\boldsymbol{q}_i), J^*(\boldsymbol{q}_i)).
\end{aligned}$$

We continue the computation using Eq. (55) and get

$$\begin{aligned}
&\sum_{j=1}^{m} c_j(1, \boldsymbol{p}_j) = \sum_{i=1}^{k} \beta_i (1, \boldsymbol{q}_i) = (1, \boldsymbol{p}); \\
&\sum_{j=1}^{m} c_j \theta_j = \sum_{i=1}^{k} \beta_i H(\boldsymbol{q}_i) \leqslant -\sum_{i=1}^{k} \beta_i E_i^- = -E^-; \\
&\sum_{j=1}^{m} c_j \gamma_j = \sum_{i=1}^{k} \beta_i J^*(\boldsymbol{q}_i) \leqslant \sum_{i=1}^{k} \beta_i r_i = r.
\end{aligned}$$

Therefore, we conclude that $(c_1, \ldots, c_m) \in \Lambda_m$ and

$$\begin{cases}
\boldsymbol{p} = \sum_{j=1}^{m} c_j \boldsymbol{p}_j, \\
E^- \leqslant -\sum_{j=1}^{m} c_j \theta_j, \\
r \geqslant \sum_{j=1}^{m} c_j \gamma_j.
\end{cases}$$

As a consequence, $\operatorname{co}(\operatorname{epi} F) \subseteq \operatorname{co}\left(\cup_{j=1}^{m} \left(\{\boldsymbol{p}_j\} \times (-\infty, -\theta_j] \times [\gamma_j, +\infty) \right) \right)$. Now, Lemmas 3.1(iii) and 3.2(iii) imply $\{\boldsymbol{p}_j\} \times (-\infty, -\theta_j] \times [\gamma_j, +\infty) \subseteq \operatorname{epi} F$ for each $j \in \{1, \ldots, m\}$. Therefore, we have

$$\begin{aligned}
\operatorname{co}(\operatorname{epi} F) = \Bigg\{ &(\boldsymbol{p}, E^-, r) \in \mathbb{R}^n \times \mathbb{R} \times \mathbb{R} : \text{there exists } (c_1, \ldots, c_m) \in \Lambda_m \text{ s.t.} \\
&\boldsymbol{p} = \sum_{j=1}^{m} c_j \boldsymbol{p}_j,\ E^- \leqslant -\sum_{j=1}^{m} c_j \theta_j,\ r \geqslant \sum_{j=1}^{m} c_j \gamma_j. \Bigg\}.
\end{aligned} \tag{57}$$

By [68, Def. IV.2.5.3 and Prop. IV.2.5.1], we have

$$\operatorname{co} F(\boldsymbol{p}, E^-) = \inf\{r \in \mathbb{R} : (\boldsymbol{p}, E^-, r) \in \operatorname{co}(\operatorname{epi} F)\}. \tag{58}$$

The conclusion then follows from Eqs. (57) and (58). □

B.4 Proof of Theorem 3.1

Proof of (i): First, the neural network f is the pointwise maximum of m affine functions in $(\boldsymbol{x}, t)$ and therefore is jointly convex in these variables. Second, as the function H is continuous and bounded in $\operatorname{dom} J^*$ by Lemma 3.2(ii), there exists a continuous and bounded function defined in $\mathbb{R}^n$ whose restriction to $\operatorname{dom} J^*$ coincides with H [57, Thm. 4.16]. Then, statement (i) follows by substituting this function for $\tilde{H}$ in statement (ii), and so it suffices to prove the latter.

Proof of (ii) (sufficiency): Suppose $\tilde{H}(\boldsymbol{p}_i) = H(\boldsymbol{p}_i)$ for every $i \in \{1, \dots, m\}$ and $\tilde{H}(\boldsymbol{p}) \geqslant H(\boldsymbol{p})$ for every $\boldsymbol{p} \in \operatorname{dom} J^*$. Since $\tilde{H}$ is continuous on $\mathbb{R}^n$ and J is convex and Lipschitz continuous with Lipschitz constant $L = \max_{i\in\{1,\dots,m\}} \|\boldsymbol{p}_i\|$, [10, Thm. 3.1] implies that $(\boldsymbol{x}, t) \mapsto \sup_{\boldsymbol{p}\in\operatorname{dom} J^*} \left\{\langle \boldsymbol{p}, \boldsymbol{x}\rangle - t\tilde{H}(\boldsymbol{p}) - J^*(\boldsymbol{p})\right\}$ is the unique uniformly continuous viscosity solution to the HJ equation (16). But this function is equivalent to the neural network f by Lemma B.1, and therefore, both sufficiency and statement (i) follow.

Proof of (ii) (necessity): Suppose the neural network f is the unique uniformly continuous viscosity solution to (16). First, we prove that $\tilde{H}(\boldsymbol{p}_k) = H(\boldsymbol{p}_k)$ for every $k \in \{1, \dots, m\}$. Fix $k \in \{1, \dots, m\}$. By Lemma B.2, there exist $\boldsymbol{x} \in \mathbb{R}^n$ and $t > 0$ satisfying $\partial_{\boldsymbol{x}} f(\boldsymbol{x}, t) = \{\boldsymbol{p}_k\}$. Use Lems. 3.1(iii) and 3.2(iii) to write the maximization problem in Eq. (8) as

$$f(\boldsymbol{x}, t) = \max_{\boldsymbol{p}\in\{\boldsymbol{p}_1,\dots,\boldsymbol{p}_m\}} \{\langle \boldsymbol{p}, \boldsymbol{x}\rangle - tH(\boldsymbol{p}) - J^*(\boldsymbol{p})\}, \tag{59}$$

where $(\boldsymbol{p}, t) \mapsto \langle \boldsymbol{p}, \boldsymbol{x}\rangle - tH(\boldsymbol{p}) - J^*(\boldsymbol{p})$ is continuous in $(\boldsymbol{p}, t)$ and differentiable in t. As the feasible set $\{\boldsymbol{p}_1, \dots, \boldsymbol{p}_m\}$ is compact, f is also differentiable with respect to t [21, Prop. 4.12], and its derivative equals

$$\frac{\partial f}{\partial t}(\boldsymbol{x}, t) = \min\left\{-H(\boldsymbol{p}) : \ \boldsymbol{p} \text{ is a maximizer in Eq. (59)}\right\}.$$

Since $\boldsymbol{x}$ and t satisfy $\partial_{\boldsymbol{x}} f(\boldsymbol{x}, t) = \{\boldsymbol{p}_k\}$, [67, Thm. VI.4.4.2] implies that the only maximizer in Eq. (59) is $\boldsymbol{p}_k$. As a result, there holds

$$\frac{\partial f}{\partial t}(\boldsymbol{x}, t) = -H(\boldsymbol{p}_k). \tag{60}$$

Since f is convex on $\mathbb{R}^n$, its subdifferential $\partial f(\boldsymbol{x}, t)$ is non-empty and satisfies

$$\partial f(\boldsymbol{x}, t) \subseteq \partial_{\boldsymbol{x}} f(\boldsymbol{x}, t) \times \partial_t f(\boldsymbol{x}, t) = \{(\boldsymbol{p}_k, -H(\boldsymbol{p}_k))\}.$$

In other words, the subdifferential $\partial f(\boldsymbol{x}, t)$ contains only one element, and therefore, f is differentiable at $(\boldsymbol{x}, t)$ and its gradient equals $(\boldsymbol{p}_k, -H(\boldsymbol{p}_k))$ [133, Thm. 21.5]. Using (16) and (60), we obtain

$$0 = \frac{\partial f}{\partial t}(\boldsymbol{x}, t) + \tilde{H}(\nabla_{\boldsymbol{x}} f(\boldsymbol{x}, t)) = -H(\boldsymbol{p}_k) + \tilde{H}(\boldsymbol{p}_k).$$

As $k \in \{1, \dots, m\}$ is arbitrary, we find that $H(\boldsymbol{p}_k) = \tilde{H}(\boldsymbol{p}_k)$ for every $k \in \{1, \dots, m\}$.

Next, we prove by contradiction that $\tilde{H}(\boldsymbol{p}) \geqslant H(\boldsymbol{p})$ for every $\boldsymbol{p} \in \operatorname{dom} J^*$. It is enough to prove the property only for every $\boldsymbol{p} \in \operatorname{ri} \operatorname{dom} J^*$ by continuity of both $\tilde{H}$ and H (where

continuity of H is proved in Lemma 3.2(ii)). Assume $\tilde{H}(\boldsymbol{p}) < H(\boldsymbol{p})$ for some $\boldsymbol{p} \in \operatorname{ri}\operatorname{dom} J^*$. Define two functions F and $\tilde{F}$ from $\mathbb{R}^n \times \mathbb{R}$ to $\mathbb{R} \cup \{+\infty\}$ by

$$F(\boldsymbol{q}, E^-) := \begin{cases} J^*(\boldsymbol{q}) & \text{if } \ E^- + H(\boldsymbol{q}) \leqslant 0, \\ +\infty & \text{otherwise.} \end{cases} \quad \text{and} \quad \tilde{F}(\boldsymbol{q}, E^-) := \begin{cases} J^*(\boldsymbol{q}) & \text{if } \ E^- + \tilde{H}(\boldsymbol{q}) \leqslant 0, \\ +\infty & \text{otherwise.} \end{cases} \tag{61}$$

for any $\boldsymbol{q} \in \mathbb{R}^n$ and $E^- \in \mathbb{R}$. Denoting the convex envelope of F by $\operatorname{co} F$, Lemma B.3 implies

$$\begin{aligned} &\operatorname{co} F(\boldsymbol{q}, E^-) = \inf_{(c_1,\ldots,c_m)\in C(\boldsymbol{q},E^-)} \sum_{i=1}^m c_i\gamma_i, \text{ where } C \text{ is defined by} \\ &C(\boldsymbol{q}, E^-) := \left\{ (c_1, \ldots, c_m) \in \Lambda_m : \sum_{i=1}^m c_i \boldsymbol{p}_i = \boldsymbol{q}, \ \sum_{i=1}^m c_i\theta_i \leqslant -E^- \right\}. \end{aligned} \tag{62}$$

Let $E_1^- \in \left(-H(\boldsymbol{p}), -\tilde{H}(\boldsymbol{p})\right)$. Now, we want to prove that $\operatorname{co} F(\boldsymbol{p}, E_1^-) \leqslant J^*(\boldsymbol{p})$; this inequality will lead to a contradiction with the definition of H.

Using statement (i) of this theorem and the supposition that f is the unique viscosity solution to the HJ equation (16), we have that

$$f(\boldsymbol{x}, t) = \sup_{\boldsymbol{q}\in\mathbb{R}^n} \{\langle \boldsymbol{q}, \boldsymbol{x}\rangle - tH(\boldsymbol{q}) - J^*(\boldsymbol{q})\} = \sup_{\boldsymbol{q}\in\mathbb{R}^n} \{\langle \boldsymbol{q}, \boldsymbol{x}\rangle - t\tilde{H}(\boldsymbol{q}) - J^*(\boldsymbol{q})\}.$$

Furthermore, a similar calculation as in the proof of [39, Prop. 3.1] yields

$$f = F^* = \tilde{F}^*, \text{ which implies } f^* = \overline{\operatorname{co}}\, F = \overline{\operatorname{co}}\, \tilde{F}.$$

where $\overline{\operatorname{co}}\, F$ and $\overline{\operatorname{co}}\, \tilde{F}$ denotes the convex lower semicontinuous envelopes of F and $\tilde{F}$, respectively. On the one hand, since $f^* = \overline{\operatorname{co}}\, \tilde{F}$, the definition of $\tilde{F}$ in Eq. (61) implies

$$f^*\left(\boldsymbol{p}, -\tilde{H}(\boldsymbol{p})\right) \leqslant \tilde{F}\left(\boldsymbol{p}, -\tilde{H}(\boldsymbol{p})\right) = J^*(\boldsymbol{p}) \quad \text{and} \quad \{\boldsymbol{p}\} \times \left(-\infty, -\tilde{H}(\boldsymbol{p})\right] \subseteq \operatorname{dom} \tilde{F} \subseteq \operatorname{dom} f^*. \tag{63}$$

Recall that $\boldsymbol{p} \in \operatorname{ri}\operatorname{dom} J^*$ and $E_1^- < -\tilde{H}(\boldsymbol{p})$, so that $(\boldsymbol{p}, E_1^-) \in \operatorname{ri}\operatorname{dom} f^*$. As a result, we get

$$\left(\boldsymbol{p}, \alpha E_1^- + (1-\alpha)(-\tilde{H}(\boldsymbol{p}))\right) \in \operatorname{ri}\operatorname{dom} f^* \text{ for all } \alpha \in (0, 1). \tag{64}$$

On the other hand, since $f^* = \operatorname{co} F$, we have $\operatorname{ri}\operatorname{dom} f^* = \operatorname{ri}\operatorname{dom}(\operatorname{co} F)$ and $f^* = \operatorname{co} F$ in $\operatorname{ri}\operatorname{dom} f^*$. Taken together with Eq. (64) and the continuity of f^*, there holds

$$\begin{aligned} f^*\left(\boldsymbol{p}, -\tilde{H}(\boldsymbol{p})\right) &= \lim_{\substack{\alpha\to 0 \\ 0<\alpha<1}} f^*\left(\boldsymbol{p}, \alpha E_1^- + (1-\alpha)(-\tilde{H}(\boldsymbol{p}))\right) \\ &= \lim_{\substack{\alpha\to 0 \\ 0<\alpha<1}} \operatorname{co} F\left(\boldsymbol{p}, \alpha E_1^- + (1-\alpha)(-\tilde{H}(\boldsymbol{p}))\right). \end{aligned} \tag{65}$$

Note that $\operatorname{co} F(\boldsymbol{p}, \cdot)$ is monotone non-decreasing. Indeed, if E_2^- is a real number such that $E_2^- > E_1^-$, by the definition of the set C in Eq. (62) there holds $C(\boldsymbol{p}, E_2^-) \subseteq C(\boldsymbol{p}, E_1^-)$, which

implies co $F(\boldsymbol{p}, E_2^-) \geqslant$ co $F(\boldsymbol{p}, E_1^-)$. Recalling that $E_1^- < -\tilde{H}(\boldsymbol{p})$, monotonicity of co $F(\boldsymbol{p}, \cdot)$ and Eq. (65) imply

$$f^*\left(\boldsymbol{p}, -\tilde{H}(\boldsymbol{p})\right) \geqslant \lim_{\substack{\alpha \to 0 \\ 0<\alpha<1}} \text{co } F\left(\boldsymbol{p}, \alpha E_1^- + (1-\alpha)E_1^-\right) = \text{co } F(\boldsymbol{p}, E_1^-). \tag{66}$$

Combining Eqs. (63) and (66), we get

$$\text{co } F(\boldsymbol{p}, E_1^-) \leqslant J^*(\boldsymbol{p}) < +\infty. \tag{67}$$

As a result, the set $C(\boldsymbol{p}, E_1^-)$ is non-empty. Since it is also compact, there exists a minimizer in Eq. (62) evaluated at the point $(\boldsymbol{p}, E_1^-)$. Let $(c_1, \ldots, c_m)$ be such a minimizer. By Eqs. (62) and (67) and the assumption that $E_1^- \in \left(-H(\boldsymbol{p}), -\tilde{H}(\boldsymbol{p})\right)$, there holds

$$\begin{cases} (c_1, \ldots, c_m) \in \Lambda_m, \\ \sum_{i=1}^m c_i \boldsymbol{p}_i = \boldsymbol{p}, \\ \sum_{i=1}^m c_i \gamma_i = \text{co } F(\boldsymbol{p}, E_1^-) \leqslant J^*(\boldsymbol{p}), \\ \sum_{i=1}^m c_i \theta_i \leqslant -E_1^- < H(\boldsymbol{p}). \end{cases} \tag{68}$$

Comparing the first three statements in Eq. (68) and the formula of J^* in Eq. (12), we deduce that $(c_1, \ldots, c_m)$ is a minimizer in Eq. (12), i.e., $(c_1, \ldots, c_m) \in \mathcal{A}(\boldsymbol{p})$. By definition of H in Eq. (14), we have

$$H(\boldsymbol{p}) = \inf_{\boldsymbol{\alpha} \in \mathcal{A}(\boldsymbol{p})} \sum_{i=1}^m \alpha_i \theta_i \leqslant \sum_{i=1}^m c_i \theta_i,$$

which contradicts the last inequality in Eq. (68). Therefore, we conclude that $\tilde{H}(\boldsymbol{p}) \geqslant H(\boldsymbol{p})$ for any $\boldsymbol{p} \in \text{ri dom } J^*$ and the proof is finished.

C Connections between the neural network (17) and the viscous HJ PDE (18)

Let f_ϵ be the neural network defined by Eq. (17) with parameters $\{(\boldsymbol{p}_i, \theta_i, \gamma_i)\}_{i=1}^m$ and $\epsilon > 0$, which is illustrated in Fig. 3. We will show in this appendix that when the parameter $\theta_i = -\frac{1}{2}\|\boldsymbol{p}_i\|_2^2$ for $i \in \{1, \ldots, m\}$, then the neural network f_ϵ corresponds to the unique, jointly convex smooth solution to the viscous HJ PDE (18). This result will follow immediately from the following lemma.

Lemma C.1 *Let $\{(\boldsymbol{p}_i, \gamma_i)\}_{i=1}^m \subset \mathbb{R}^n \times \mathbb{R}$ and $\epsilon > 0$. Then, the function $w_\epsilon : \mathbb{R}^n \mapsto \mathbb{R}$ defined by*

$$w_\epsilon(\boldsymbol{x}, t) := \sum_{i=1}^m e^{\left(\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle + \frac{t}{2}\|\boldsymbol{p}_i\|_2^2 - \gamma_i\right)/\epsilon} \tag{69}$$

is the unique, jointly log-convex and smooth solution to the Cauchy problem

$$\begin{cases} \dfrac{\partial w_\epsilon}{\partial t}(\boldsymbol{x}, t) = \dfrac{\epsilon}{2} \Delta_{\boldsymbol{x}} w_\epsilon(\boldsymbol{x}, t) & \text{in } \mathbb{R}^n \times (0, +\infty), \\ w_\epsilon(\boldsymbol{x}, 0) = \displaystyle\sum_{i=1}^m e^{(\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle - \gamma_i)/\epsilon} & \text{in } \mathbb{R}^n. \end{cases} \tag{70}$$

Proof A short calculation shows that the function w_ϵ defined in Eq. (69) solves the Cauchy problem (70), and uniqueness holds by strict positiveness of the initial data (see [147, Chap. VIII, Thm. 2.2] and note that the uniqueness result can easily be generalized to $n > 1$).

Now, let $\lambda \in [0,1]$ and $(\boldsymbol{x}_1, t_1)$ and $(\boldsymbol{x}_2, t_2)$ be such that $\boldsymbol{x} = \lambda \boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2$ and $t = \lambda t_1 + (1-\lambda)t_2$. Then, the Hölder's inequality (see, e.g., [57, Thm. 6.2]) implies

$$\begin{aligned}\sum_{i=1}^m e^{\left(\langle \boldsymbol{p}_i, \boldsymbol{x}\rangle + \frac{t}{2}\|\boldsymbol{p}_i\|_2^2 - \gamma_i\right)/\epsilon} &= \sum_{i=1}^m \left(e^{\lambda\left(\langle \boldsymbol{p}_i, \boldsymbol{x}_1\rangle + \frac{t_1}{2}\|\boldsymbol{p}_i\|_2^2 - \gamma_i\right)/\epsilon} e^{(1-\lambda)\left(\langle \boldsymbol{p}_i, \boldsymbol{x}_2\rangle + \frac{t_2}{2}\|\boldsymbol{p}_i\|_2^2 - \gamma_i\right)/\epsilon} \right) \\ &\leqslant \left(\sum_{i=1}^m e^{\left(\langle \boldsymbol{p}_i, \boldsymbol{x}_1\rangle + \frac{t_1}{2}\|\boldsymbol{p}_i\|_2^2 - \gamma_i\right)/\epsilon} \right)^{\lambda} \left(\sum_{i=1}^m e^{\left(\langle \boldsymbol{p}_i, \boldsymbol{x}_2\rangle + \frac{t_2}{2}\|\boldsymbol{p}_i\|_2^2 - \gamma_i\right)/\epsilon} \right)^{1-\lambda},\end{aligned}$$

and we find $w_\epsilon(\boldsymbol{x}, t) \leqslant (w_\epsilon(\boldsymbol{x}_1, t_1))^{\lambda} (w_\epsilon(\boldsymbol{x}_2, t_2))^{1-\lambda}$, which implies that w_ϵ is jointly log-convex in $(\boldsymbol{x}, t)$. □

Thanks to Lemma C.1 and the Cole–Hopf transformation $f_\epsilon(\boldsymbol{x}, t) = \epsilon \log(w_\epsilon(\boldsymbol{x}, t))$ (see, e.g., [47], Sect. 4.4.1), a short calculation immediately implies that the neural network f_ϵ solves the viscous HJ PDE (18), and it is also its unique solution because w_ϵ is the unique solution to the Cauchy problem (70). Joint convexity in $(\boldsymbol{x}, t)$ follows from log-convexity of $(\boldsymbol{x}, t) \mapsto w_\epsilon(\boldsymbol{x}, t)$ for every $\epsilon > 0$.

D Proof of Proposition 3.1

To prove this proposition, we will use three lemmas whose statements and proofs are given in Sect. D.1, D.2, and D.3, respectively. The proof of Prop. 3.1 is given in Sect. D.4.

D.1 Statement and proof of Lemma D.1

Lemma D.1 *Consider the one-dimensional case, i.e., $n = 1$. Let $p_1, \ldots, p_m \in \mathbb{R}$ satisfy $p_1 < \cdots < p_m$ and define the function J using Eq. (10). Suppose assumptions (A1)-(A2) hold. Let $x \in \mathbb{R}$, $p \in \partial J(x)$, and suppose $p \neq p_i$ for any $i \in \{1, \ldots, m\}$. Then, there exists $k \in \{1, \ldots, m\}$ such that $p_k < p < p_{k+1}$ and*

$$k, k+1 \in \operatorname*{arg\,max}_{i \in \{1,\ldots,m\}} \{x p_i - \gamma_i\}. \tag{71}$$

Proof Let I_x denotes the set of maximizers in Eq. (11) at x. Since $p \in \partial J(x)$, $p \neq p_i$ for $i \in \{1, \ldots, m\}$, and $\partial J(x) = \operatorname{co}\{p_i : i \in I_i\}$ by [67, Thm. VI.4.4.2], there exist $j, l \in I_x$ such that $p_j < p < p_l$. Moreover, there exists k with $j \leqslant k < k+1 \leqslant l$ such that $p_j \leqslant p_k < p < p_{k+1} \leqslant p_l$. We will show that $k, k+1 \in I_x$. We only prove $k \in I_x$; the case for $k+1$ is similar.

If $p_j = p_k$, then $k = j \in I_x$ and the conclusion follows directly. Hence suppose $p_j < p_k < p_l$. Then, there exists $\alpha \in (0,1)$ such that $p_k = \alpha p_j + (1-\alpha)p_l$. Using that $j, l \in I_x$, assumption (A2), and Jensen inequality, we get

$$\begin{aligned} x p_k - \gamma_k = x p_k - g(p_k) &= (\alpha p_j + (1-\alpha)p_l)x - g(\alpha p_j + (1-\alpha)p_l) \\ &\geqslant \alpha x p_j + (1-\alpha) x p_l - \alpha g(p_j) - (1-\alpha) g(p_l) \\ &= \alpha(x p_j - \gamma_j) + (1-\alpha)(x p_l - \gamma_l) \\ &= \max_{i \in \{1,\ldots,m\}} \{x p_i - \gamma_i\}, \end{aligned}$$

which implies that $k \in I_x$. A similar argument shows that $k+1 \in I_x$, which completes the proof. □

D.2 Statement and proof of Lemma D.2

Lemma D.2 *Consider the one-dimensional case, i.e., $n = 1$. Let $p_1, \ldots, p_m \in \mathbb{R}$ satisfy $p_1 < \cdots < p_m$ and define the function H using Eq. (14). Suppose assumptions (A1)–(A3) hold. Let $u_0 \in \mathbb{R}$ and $p_k < u_0 < p_{k+1}$ for some index k. Then, there holds*

$$H(u_0) = \beta_k \theta_k + \beta_{k+1}\theta_{k+1}, \tag{72}$$

where

$$\beta_k := \frac{p_{k+1} - u_0}{p_{k+1} - p_k} \quad \text{and} \quad \beta_{k+1} := \frac{u_0 - p_k}{p_{k+1} - p_k}. \tag{73}$$

Proof Let $\boldsymbol{\beta} := (\beta_1, \ldots, \beta_m) \in \Lambda_m$ satisfy

$$\beta_k := \frac{p_{k+1} - u_0}{p_{k+1} - p_k} \quad \text{and} \quad \beta_{k+1} := \frac{u_0 - p_k}{p_{k+1} - p_k},$$

and $\beta_i = 0$ for every $i \in \{1, \ldots, m\} \setminus \{k, k+1\}$. We will prove that $\boldsymbol{\beta}$ is a minimizer in Eq. (14) evaluated at u_0, that is,

$$\boldsymbol{\beta} \in \operatorname*{arg\,min}_{\boldsymbol{\alpha} \in \mathcal{A}(u_0)} \left\{ \sum_{i=1}^m \alpha_i \theta_i \right\},$$

where

$$\mathcal{A}(u_0) := \operatorname*{arg\,min}_{\substack{(\alpha_1, \ldots \alpha_m) \in \Lambda_m \\ \sum_{i=1}^m \alpha_i p_i = u_0}} \left\{ \sum_{i=1}^m \alpha_i \gamma_i \right\}.$$

First, we show that $\boldsymbol{\beta} \in \mathcal{A}(u_0)$. By definition of $\boldsymbol{\beta}$ and Lemma 3.1(ii) with $p = u_0$, the statement holds provided $k, k+1 \in I_x$, where the set I_x contains the maximizers in Eq. (10) evaluated at $x \in \partial J^*(u_0)$. But if $x \in \partial J^*(u_0)$, we have $u_0 \in \partial J(x)$, and Lemma D.1 implies $k, k+1 \in I_x$. Hence, $\boldsymbol{\beta} \in \mathcal{A}(u_0)$.

Now, suppose that $\boldsymbol{\beta}$ is not a minimizer in Eq. (14) evaluated at u_0. By Lemma 3.2(i), there exists a minimizer in Eq. (14) evaluated at the point u_0, which we denote by $(\alpha_1, \ldots, \alpha_m)$. Then there holds

$$\begin{cases} \sum_{i=1}^m \alpha_i = \sum_{i=1}^m \beta_i = 1, \\ \sum_{i=1}^m \alpha_i p_i = \sum_{i=1}^m \beta_i p_i = u_0, \\ \sum_{i=1}^m \alpha_i \gamma_i = \sum_{i=1}^m \beta_i \gamma_i = J^*(u_0), \\ \sum_{i=1}^m \alpha_i \theta_i < \sum_{i=1}^m \beta_i \theta_i. \end{cases} \tag{74}$$

Since $\alpha_i \geqslant 0$ for every i and $\beta_i = 0$ for every $i \in \{1, \ldots, m\} \setminus \{k, k+1\}$, we have $\alpha_k + \alpha_{k+1} \leqslant 1 = \beta_k + \beta_{k+1}$. As $\boldsymbol{\alpha} \neq \boldsymbol{\beta}$, then one or both of the inequalities $\alpha_k < \beta_k$ and $\alpha_{k+1} < \beta_{k+1}$ hold. This leaves three possible cases, and we now show that each case leads to a contradiction.

Case 1: Let $\alpha_k < \beta_k$ and $\alpha_{k+1} \geqslant \beta_{k+1}$. Define the coefficient c_i by

$$c_i := \begin{cases} \dfrac{\alpha_i - \beta_i}{\beta_k - \alpha_k}, & i \neq k, \\ 0, & i = k. \end{cases}$$

The following equations then hold

$$\begin{cases} (c_1, \ldots, c_m) \in \Delta_m \text{ with } c_k = 0, \\ \sum_{i \neq k} c_i p_i = p_k, \\ \sum_{i \neq k} c_i \gamma_i = \gamma_k, \\ \sum_{i \neq k} c_i \theta_i < \theta_k. \end{cases}$$

These equations, however, violate assumption (A3), and so we get a contradiction.

Case 2: Let $\alpha_k \geqslant \beta_k$ and $\alpha_{k+1} < \beta_{k+1}$. A similar argument as in case 1 can be applied here by exchanging the indices k and $k+1$ to derive a contradiction.

Case 3: Let $\alpha_k < \beta_k$ and $\alpha_{k+1} < \beta_{k+1}$. From Eq. (74), we obtain

$$\begin{cases} \beta_k - \alpha_k + \beta_{k+1} - \alpha_{k+1} = \sum_{i \neq k,k+1} \alpha_i, \\ (\beta_k - \alpha_k) p_k + (\beta_{k+1} - \alpha_{k+1}) p_{k+1} = \sum_{i \neq k,k+1} \alpha_i p_i, \\ (\beta_k - \alpha_k) \gamma_k + (\beta_{k+1} - \alpha_{k+1}) \gamma_{k+1} = \sum_{i \neq k,k+1} \alpha_i \gamma_i, \\ (\beta_k - \alpha_k) \theta_k + (\beta_{k+1} - \alpha_{k+1}) \theta_{k+1} > \sum_{i \neq k,k+1} \alpha_i \theta_i. \end{cases} \tag{75}$$

Define two numbers q_k and q_{k+1} by

$$q_k := \frac{\sum_{i<k} \alpha_i p_i}{\sum_{i<k} \alpha_i} \quad \text{and} \quad q_{k+1} := \frac{\sum_{i>k+1} \alpha_i p_i}{\sum_{i>k+1} \alpha_i}. \tag{76}$$

Note that from the first two equations in (74) and the assumption that $\alpha_k < \beta_k$ and $\alpha_{k+1} < \beta_{k+1}$, there exist $i_1 < k$ and $i_2 > k+1$ such that $\alpha_{i_1} \neq 0$ and $\alpha_{i_2} \neq 0$, and hence, the numbers q_k and q_{k+1} are well-defined. By definition, we have $q_k < p_k < p_{k+1} < q_{k+1}$. Therefore, there exist $b_k, b_{k+1} \in (0, 1)$ such that

$$p_k = b_k q_k + (1 - b_k) q_{k+1} \quad \text{and} \quad p_{k+1} = b_{k+1} q_k + (1 - b_{k+1}) q_{k+1}. \tag{77}$$

A straightforward computation yields

$$b_k = \frac{q_{k+1} - p_k}{q_{k+1} - q_k} \quad \text{and} \quad b_{k+1} = \frac{q_{k+1} - p_{k+1}}{q_{k+1} - q_k}. \tag{78}$$

Define the coefficients c_i^k and c_i^{k+1} as follows

$$c_i^k := \begin{cases} \dfrac{b_k \alpha_i}{\sum_{\omega<k} \alpha_\omega}, & i < k, \\ \dfrac{(1 - b_k) \alpha_i}{\sum_{\omega>k+1} \alpha_\omega}, & i > k+1, \\ 0, & \text{otherwise}, \end{cases} \quad \text{and} \quad c_i^{k+1} := \begin{cases} \dfrac{b_{k+1} \alpha_i}{\sum_{\omega<k} \alpha_\omega}, & i < k, \\ \dfrac{(1 - b_{k+1}) \alpha_i}{\sum_{\omega>k+1} \alpha_\omega}, & i > k+1, \\ 0, & \text{otherwise}. \end{cases} \tag{79}$$

These coefficients satisfy $c_i^k, c_i^{k+1} \in [0,1]$ for any i and $\sum_{i=1}^m c_i^k = \sum_{i=1}^m c_i^{k+1} = 1$. In other words, we have

$$(c_1^k, \ldots, c_m^k) \in \Delta_m \text{ with } c_k^k = 0 \quad \text{and} \quad (c_1^{k+1}, \ldots, c_m^{k+1}) \in \Delta_m \text{ with } c_{k+1}^{k+1} = 0. \tag{80}$$

Hence, the first equality in Eq. (9) holds for the coefficients $(c_1^k, \ldots, c_m^k)$ with the index k and also for the coefficients $(c_1^{k+1}, \ldots, c_m^{k+1})$ with the index $k+1$. We show next that these coefficients satisfy the second and third equalities in (9) and draw a contradiction with assumption (A3).

Using Eqs. (76), (77), and (79) to write the formulas for p_k and p_{k+1} via the coefficients c_i^k and c_i^{k+1}, we find

$$\begin{aligned}
p_k &= b_k \frac{\sum_{i<k} \alpha_i p_i}{\sum_{i<k} \alpha_i} + (1-b_k) \frac{\sum_{i>k+1} \alpha_i p_i}{\sum_{i>k+1} \alpha_i} = \sum_{i \neq k, k+1} c_i^k p_i = \sum_{i \neq k} c_i^k p_i, \\
p_{k+1} &= b_{k+1} \frac{\sum_{i<k} \alpha_i p_i}{\sum_{i<k} \alpha_i} + (1-b_{k+1}) \frac{\sum_{i>k+1} \alpha_i p_i}{\sum_{i>k+1} \alpha_i} = \sum_{i \neq k, k+1} c_i^{k+1} p_i = \sum_{i \neq k+1} c_i^{k+1} p_i,
\end{aligned} \tag{81}$$

where the last equalities in the two formulas above hold because $c_{k+1}^k = 0$ and $c_k^{k+1} = 0$ by definition. Hence, the second equality in Eq. (9) also holds for both the index k and $k+1$.

From the third equality in Eq. (75), assumption (A2), Eq. (81), and Jensen's inequality, we have

$$\begin{aligned}
&\sum_{i \neq k, k+1} \alpha_i \gamma_i = (\beta_k - \alpha_k)\gamma_k + (\beta_{k+1} - \alpha_{k+1})\gamma_{k+1} \\
&= (\beta_k - \alpha_k) g(p_k) + (\beta_{k+1} - \alpha_{k+1}) g(p_{k+1}) \\
&= (\beta_k - \alpha_k) g\left(\sum_{i \neq k, k+1} c_i^k p_i \right) + (\beta_{k+1} - \alpha_{k+1}) g\left(\sum_{i \neq k, k+1} c_i^{k+1} p_i \right) \\
&\leqslant (\beta_k - \alpha_k) \left(\sum_{i \neq k, k+1} c_i^k g(p_i) \right) + (\beta_{k+1} - \alpha_{k+1}) \left(\sum_{i \neq k, k+1} c_i^{k+1} g(p_i) \right) \\
&= \sum_{i \neq k, k+1} ((\beta_k - \alpha_k) c_i^k + (\beta_{k+1} - \alpha_{k+1}) c_i^{k+1}) g(p_i) \\
&= \sum_{i \neq k, k+1} ((\beta_k - \alpha_k) c_i^k + (\beta_{k+1} - \alpha_{k+1}) c_i^{k+1}) \gamma_i.
\end{aligned} \tag{82}$$

We now compute and simplify the coefficients $(\beta_k - \alpha_k) c_i^k + (\beta_{k+1} - \alpha_{k+1}) c_i^{k+1}$ in the formula above. First, consider the case when $i < k$. Eqs. (78) and (79) imply

$$\begin{aligned}
&(\beta_k - \alpha_k) c_i^k + (\beta_{k+1} - \alpha_{k+1}) c_i^{k+1} \\
&= (\beta_k - \alpha_k) \frac{b_k \alpha_i}{\sum_{\omega<k} \alpha_\omega} + (\beta_{k+1} - \alpha_{k+1}) \frac{b_{k+1} \alpha_i}{\sum_{\omega<k} \alpha_\omega} \\
&= \frac{\alpha_i}{\sum_{\omega<k} \alpha_\omega} ((\beta_k - \alpha_k) b_k + (\beta_{k+1} - \alpha_{k+1}) b_{k+1})
\end{aligned}$$

$$
\begin{aligned}
&= \frac{\alpha_i}{\sum_{\omega<k}\alpha_\omega}\left((\beta_k-\alpha_k)\frac{q_{k+1}-p_k}{q_{k+1}-q_k}+(\beta_{k+1}-\alpha_{k+1})\frac{q_{k+1}-p_{k+1}}{q_{k+1}-q_k}\right)\\
&= \frac{\alpha_i}{\sum_{\omega<k}\alpha_\omega}\cdot\frac{1}{q_{k+1}-q_k}((\beta_k-\alpha_k+\beta_{k+1}-\alpha_{k+1})q_{k+1}\\
&-(\beta_k-\alpha_k)p_k-(\beta_{k+1}-\alpha_{k+1})p_{k+1}).
\end{aligned}
$$

Applying the first two equalities in Eq. (75) and Eq. (76) to the last formula above, we obtain

$$
\begin{aligned}
&(\beta_k-\alpha_k)c_i^k+(\beta_{k+1}-\alpha_{k+1})c_i^{k+1}\\
&= \frac{\alpha_i}{\sum_{\omega<k}\alpha_\omega}\cdot\frac{1}{q_{k+1}-q_k}\left(\left(\sum_{i\neq k,k+1}\alpha_i\right)q_{k+1}-\sum_{i\neq k,k+1}\alpha_i p_i\right)\\
&= \frac{\alpha_i}{\sum_{\omega<k}\alpha_\omega}\cdot\frac{1}{q_{k+1}-q_k}\left(\sum_{i\neq k,k+1}\alpha_i q_{k+1}-\sum_{i<k}\alpha_i p_i-\sum_{i>k+1}\alpha_i p_i\right)\\
&= \frac{\alpha_i}{\sum_{\omega<k}\alpha_\omega}\cdot\frac{1}{q_{k+1}-q_k}\left(\sum_{i\neq k,k+1}\alpha_i q_{k+1}-\left(\sum_{i<k}\alpha_i\right)q_k-\left(\sum_{i>k+1}\alpha_i\right)q_{k+1}\right)\\
&= \frac{\alpha_i}{\sum_{\omega<k}\alpha_\omega}\cdot\frac{1}{q_{k+1}-q_k}\left(\sum_{i<k}\alpha_i(q_{k+1}-q_k)\right)\\
&= \alpha_i.
\end{aligned}
$$

The same result for the case when $i > k+1$ also holds and the proof is similar. Therefore, we have

$$
(\beta_k-\alpha_k)c_i^k+(\beta_{k+1}-\alpha_{k+1})c_i^{k+1}=\alpha_i \quad \text{for each } i\neq k,k+1. \tag{83}
$$

Combining Eqs. (82) and (83), we have

$$
\sum_{i\neq k,k+1}\alpha_i\gamma_i \leqslant \sum_{i\neq k,k+1}((\beta_k-\alpha_k)c_i^k+(\beta_{k+1}-\alpha_{k+1})c_i^{k+1})\gamma_i=\sum_{i\neq k,k+1}\alpha_i\gamma_i.
$$

Since the left side and right side are the same, the inequality above becomes equality, which implies that the inequality in Eq. (82) also becomes equality. In other words, we have

$$
\begin{aligned}
\gamma_k = g(p_k) &= \sum_{i\neq k,k+1}c_i^k g(p_i)=\sum_{i\neq k,k+1}c_i^k\gamma_i=\sum_{i\neq k}c_i^k\gamma_i,\\
\gamma_{k+1}=g(p_{k+1}) &= \sum_{i\neq k,k+1}c_i^{k+1}g(p_i)=\sum_{i\neq k,k+1}c_i^{k+1}\gamma_i=\sum_{i\neq k+1}c_i^{k+1}\gamma_i,
\end{aligned} \tag{84}
$$

where the last equalities in the two formulas above hold because $c_{k+1}^k=0$ and $c_k^{k+1}=0$ by definition. Hence, the third equality in (9) also holds for both indices k and $k+1$.

In summary, Eqs. (80), (81), and (84) imply that Eq. (9) holds for the index k with coefficients $(c_1^k,\ldots,c_m^k)$ and also for the index $k+1$ with coefficients $(c_1^{k+1},\ldots,c_m^{k+1})$. Hence, by assumption (A3), we find

$$
\sum_{i\neq k}c_i^k\theta_i>\theta_k \quad \text{and} \quad \sum_{i\neq k+1}c_i^{k+1}\theta_i>\theta_{k+1}.
$$

Using the inequalities above with Eq. (83) and the fact that $c_{k+1}^k = 0$ and $c_k^{k+1} = 0$, we find

$$\begin{aligned}(\beta_k - \alpha_k)\theta_k + (\beta_{k+1} - \alpha_{k+1})\theta_{k+1} &< (\beta_k - \alpha_k)\sum_{i\neq k} c_i^k\theta_i + (\beta_{k+1} - \alpha_{k+1})\sum_{i\neq k+1} c_i^{k+1}\theta_i \\ &= \sum_{i\neq k,k+1} ((\beta_k - \alpha_k)c_i^k + (\beta_{k+1} - \alpha_{k+1})c_i^{k+1})\theta_i = \sum_{i\neq k,k+1} \alpha_i\theta_i,\end{aligned}$$

which contradicts the last inequality in Eq. (75).

In conclusion, we obtain contradictions in all the three cases. As a consequence, we conclude that $\boldsymbol{\beta}$ is a minimizer in Eq. (14) evaluated at u_0 and Eq. (72) follows from the definition of H in (14). □

D.3 Statement and proof of Lemma D.3

Lemma D.3 *Consider the one-dimensional case, i.e., $n = 1$. Let $p_1, \ldots, p_m \in \mathbb{R}$ satisfy $p_1 < \cdots < p_m$. Suppose assumptions (A1)-(A2) hold. Let $x \in \mathbb{R}$ and $t > 0$. Assume j, k, l are three indices such that $1 \leqslant j \leqslant k < l \leqslant m$ and*

$$j, l \in \operatorname*{arg\,max}_{i\in\{1,\ldots,m\}} \{xp_i - t\theta_i - \gamma_i\}. \tag{85}$$

Then, there holds

$$\frac{\theta_l - \theta_k}{p_l - p_k} \leqslant \frac{\theta_l - \theta_j}{p_l - p_j}. \tag{86}$$

Proof Note that Eq. (86) holds trivially when $j = k$, so we only need to consider the case when $j < k < l$. On the one hand, Eq. (85) implies

$$xp_j - t\theta_j - \gamma_j = xp_l - t\theta_l - \gamma_l \geqslant xp_k - t\theta_k - \gamma_k,$$

which yields

$$\begin{aligned}\gamma_l - \gamma_k &\leqslant x(p_l - p_k) - t(\theta_l - \theta_k), \\ \gamma_l - \gamma_j &= x(p_l - p_j) - t(\theta_l - \theta_j).\end{aligned} \tag{87}$$

On the other hand, for each $i \in \{j, j+1, \ldots, l-1\}$ let $q_i \in (p_i, p_{i+1})$ and $x_i \in \partial J^*(q_i)$. Such x_i exists because $q_i \in \operatorname{int}\operatorname{dom} J^*$, so that the subdifferential $\partial J^*(q_i)$ is non-empty. Then, $q_i \in \partial J(x_i)$ and Lemma D.1 imply

$$x_ip_i - \gamma_i = x_ip_{i+1} - \gamma_{i+1} = \max_{\omega\in\{1,\ldots,m\}} \{x_ip_\omega - \gamma_\omega\}.$$

A straightforward computation yields

$$\begin{aligned}\gamma_l - \gamma_k &= \sum_{i=k}^{l-1} (\gamma_{i+1} - \gamma_i) = \sum_{i=k}^{l-1} x_i(p_{i+1} - p_i), \\ \gamma_l - \gamma_j &= \sum_{i=j}^{l-1} (\gamma_{i+1} - \gamma_i) = \sum_{i=j}^{l-1} x_i(p_{i+1} - p_i).\end{aligned}$$

Combining the two equalities above with Eq. (87), we conclude that

$$x(p_l - p_k) - t(\theta_l - \theta_k) \geqslant \sum_{i=k}^{l-1} x_i(p_{i+1} - p_i),$$
$$x(p_l - p_j) - t(\theta_l - \theta_j) = \sum_{i=j}^{l-1} x_i(p_{i+1} - p_i).$$

Now, divide the inequality above by $t(p_l - p_k) > 0$ (because by assumption $t > 0$ and $l > k$, which implies that $p_l > p_k$), divide the equality above by $t(p_l - p_j) > 0$ (because $l > j$, which implies that $t(p_l - p_j) \neq 0$), and rearrange the terms to obtain

$$\begin{aligned} \frac{\theta_l - \theta_k}{p_l - p_k} &\leqslant \frac{x}{t} - \frac{1}{t}\frac{\sum_{i=k}^{l-1} x_i(p_{i+1} - p_i)}{p_l - p_k}, \\ \frac{\theta_l - \theta_j}{p_l - p_j} &= \frac{x}{t} - \frac{1}{t}\frac{\sum_{i=j}^{l-1} x_i(p_{i+1} - p_i)}{p_l - p_j}. \end{aligned} \tag{88}$$

Recall that $q_j < q_{j+1} < \cdots < q_{l-1}$ and $x_i \in \partial J^*(q_i)$ for any $j \leqslant i < l$. Since the function J^* is convex, the subdifferential operator ∂J^* is a monotone non-decreasing operator [67, Def. IV.4.1.3, and Prop. VI.6.1.1], which yields $x_j \leqslant x_{j+1} \leqslant \cdots \leqslant x_{l-1}$. Using that $p_1 < p_2 < \cdots < p_m$ and $j < k < l$, we obtain

$$\begin{aligned} &\frac{\sum_{i=k}^{l-1} x_i(p_{i+1} - p_i)}{p_l - p_k} \geqslant \frac{\sum_{i=k}^{l-1} x_k(p_{i+1} - p_i)}{p_l - p_k} = x_k \\ &= \frac{\sum_{i=j}^{k-1} x_k(p_{i+1} - p_i)}{p_k - p_j} \geqslant \frac{\sum_{i=j}^{k-1} x_i(p_{i+1} - p_i)}{p_k - p_j}. \end{aligned} \tag{89}$$

To proceed, we now use that fact that if four real numbers $a, c \in \mathbb{R}$ and $b, d > 0$ satisfy $\frac{a}{b} \geqslant \frac{c}{d}$, then $\frac{a}{b} \geqslant \frac{a+c}{b+d}$. Combining this fact with inequality (89), we find

$$\begin{aligned} &\frac{\sum_{i=k}^{l-1} x_i(p_{i+1} - p_i)}{p_l - p_k} \geqslant \frac{\sum_{i=k}^{l-1} x_i(p_{i+1} - p_i) + \sum_{i=j}^{k-1} x_i(p_{i+1} - p_i)}{p_l - p_k + p_k - p_j} \\ &= \frac{\sum_{i=j}^{l-1} x_i(p_{i+1} - p_i)}{p_l - p_j}. \end{aligned}$$

We combine the inequality above with (88) to obtain

$$\frac{\theta_l - \theta_k}{p_l - p_k} \leqslant \frac{\theta_l - \theta_j}{p_l - p_j}.$$

which concludes the proof. □

D.4 Proof of Proposition 3.1

Proof of (i): First, note that u is piecewise constant. Second, recall that J is defined as the pointwise maximum of a finite number of affine functions. Therefore, the initial data $u(\cdot, 0) = \nabla J(\cdot)$ (recall that here, the gradient ∇ is taken in the sense of distribution) are bounded and of locally bounded variation (see [48, Chap. 5, page 167] for the definition of locally bounded variation). Finally, the flux function H, defined in Eq. (14), is Lipschitz continuous in dom J^* by Lemma D.2. It can therefore be extended to $\mathbb{R}$ while preserving

its Lipschitz property [57, Thm. 4.16]. Therefore, we can invoke [36, Prop. 2.1] to conclude that u is the entropy solution to the conservation law (21) provided it satisfies the two following conditions. Let $\bar{x}(t)$ be any smooth line of discontinuity of u. Fix $t > 0$ and define u^- and u^+ as

$$u^- := \lim_{x \to \bar{x}(t)^-} u(x,t) \quad \text{and} \quad u^+ := \lim_{x \to \bar{x}(t)^+} u(x,t). \tag{90}$$

Then, the two conditions are:

1. The curve $\bar{x}(t)$ is a straight line with the slope

$$\frac{d\bar{x}}{dt} = \frac{H(u^+) - H(u^-)}{u^+ - u^-}. \tag{91}$$

2. For any u_0 between u^+ and u^-, we have

$$\frac{H(u^+) - H(u_0)}{u^+ - u_0} \leqslant \frac{H(u^+) - H(u^-)}{u^+ - u^-}. \tag{92}$$

First, we prove the first condition and Eq. (91). According to the definition of u in Eq. (20), the range of u is the compact set $\{p_1, \ldots, p_m\}$. As a result, u^- and u^+ are in the range of u, i.e., there exist indices j and l such that

$$u^- = p_j \quad \text{and} \quad u^+ = p_l. \tag{93}$$

Let $(\bar{x}(s), s)$ be a point on the curve $\bar{x}$ which is not one of the endpoints. Since u is piecewise constant, there exists a neighborhood $\mathcal{N}$ of $(\bar{x}(s), s)$ such that for any $(x^-, t), (x^+, t) \in \mathcal{N}$ satisfying $x^- < \bar{x}(t) < x^+$, we have $u(x^-, t) = u^- = p_j$ and $u(x^+, t) = u^+ = p_l$. In other words, if x^-, x^+, t are chosen as above, according to the definition of u in Eq. (20), we have

$$j \in \arg\max_{i \in \{1,\ldots,m\}} \{x^- p_i - t\theta_i - \gamma_i\} \quad \text{and} \quad l \in \arg\max_{i \in \{1,\ldots,m\}} \{x^+ p_i - t\theta_i - \gamma_i\}. \tag{94}$$

Define a sequence $\{x_k^-\}_{k=1}^{+\infty} \subset (-\infty, \bar{x}(s))$ such that $(x_k^-, s) \in \mathcal{N}$ for any $k \in \mathbb{N}$ and $\lim_{k \to +\infty} x_k^- = \bar{x}(s)$. By Eq. (94), we have

$$x_k^- p_j - s\theta_j - \gamma_j \geq x_k^- p_i - s\theta_i - \gamma_i \text{ for any } i \in \{1, \ldots, m\}.$$

When k approaches infinity, the above inequality implies

$$\bar{x}(s) p_j - s\theta_j - \gamma_j \geq \bar{x}(s) p_i - s\theta_i - \gamma_i \quad \text{for any } i \in \{1, \ldots, m\}.$$

In other words, we have

$$j \in \arg\max_{i \in \{1,\ldots,m\}} \{\bar{x}(s) p_i - s\theta_i - \gamma_i\}. \tag{95}$$

Similarly, define a sequence $\{x_k^+\}_{k=1}^{+\infty} \subset (\bar{x}(s), +\infty)$ such that $(x_k^+, s) \in \mathcal{N}$ for any $k \in \mathbb{N}$ and $\lim_{k \to +\infty} x_k^+ = \bar{x}(s)$. Using a similar argument as above, we can conclude that

$$l \in \arg\max_{i \in \{1,\ldots,m\}} \{\bar{x}(s) p_i - s\theta_i - \gamma_i\}. \tag{96}$$

By a continuity argument, Eqs. (95) and (96) also hold for the end points of $\tilde{x}$. In conclusion, for any $(\tilde{x}(t), t)$ on the curve $\tilde{x}$, we have

$$j, l \in \underset{i\in\{1,\ldots,m\}}{\arg\max}\{\tilde{x}(t)p_i - t\theta_i - \gamma_i\}, \tag{97}$$

which implies that

$$\tilde{x}(t)p_l - t\theta_l - \gamma_l = \tilde{x}(t)p_j - t\theta_j - \gamma_j.$$

Therefore, the curve $\tilde{x}(t)$ lies on the straight line

$$x(p_l - p_j) - t(\theta_l - \theta_j) - (\gamma_l - \gamma_j) = 0$$

and Eq. (93) and Lemma 3.2(iii) imply that its slope equals

$$\frac{d\tilde{x}}{dt} = \frac{\theta_l - \theta_j}{p_l - p_j} = \frac{H(u^+) - H(u^-)}{u^+ - u^-}.$$

This proves Eq. (91) and the first condition holds.

It remains to show the second condition. Since u equals $\nabla_x f$ and f is convex by Theorem 3.1, its corresponding subdifferential operator u is monotone non-decreasing with respect to x [67, Def. IV.4.1.3 and Prop. VI.6.1.1]. As a result, $u^- < u^+$ and $u_0 \in (u^-, u^+)$, where we still adopt the notation $u^- = p_j$ and $u^+ = p_l$. Recall that Lemma 3.2(iii) implies $H(p_i) = \theta_i$ for any i. Then, Eq. (92) in the second condition becomes

$$\frac{\theta_l - H(u_0)}{p_l - u_0} \leqslant \frac{\theta_l - \theta_j}{p_l - p_j}. \tag{98}$$

Without loss of generality, we may assume that $p_1 < p_2 < \cdots < p_m$. Then, the fact $p_j = u^- < u^+ = p_l$ implies $j < l$. We consider the following two cases.

First, if there exists some k such that $u_0 = p_k$, then $H(u_0) = \theta_k$ by Lemma 3.2(iii). Since $u^- < u_0 < u^+$, we have $j < k < l$. Recall that Eq. (97) holds. Therefore, the assumptions of Lemma D.3 are satisfied, which implies Eq. (98) holds.

Second, suppose $u_0 \neq p_i$ for every $i \in \{1, \ldots, m\}$. Then there exists some $k \in \{j, j+1, \ldots, l-1\}$ such that $p_k < u_0 < p_{k+1}$. Lemma D.2 then implies that Eqs. (72) and (73) hold, that is,

$$H(u_0) = \beta_k\theta_k + \beta_{k+1}\theta_{k+1}, \quad u_0 = \beta_k p_k + \beta_{k+1}p_{k+1}, \quad \text{and} \quad \beta_k + \beta_{k+1} = 1.$$

Using these three equations, we can write the left-hand side of Eq. (98) as

$$\frac{\theta_l - H(u_0)}{p_l - u_0} = \frac{\theta_l - \beta_k\theta_k - \beta_{k+1}\theta_{k+1}}{p_l - \beta_k p_k - \beta_{k+1}p_{k+1}} = \frac{\beta_k(\theta_l - \theta_k) + \beta_{k+1}(\theta_l - \theta_{k+1})}{\beta_k(p_l - p_k) + \beta_{k+1}(p_l - p_{k+1})}. \tag{99}$$

If $k + 1 = l$, then this equation become

$$\frac{\theta_l - H(u_0)}{p_l - u_0} = \frac{\theta_l - \theta_k}{p_l - p_k}.$$

Since $j \leqslant k < l$ and Eq. (97) hold, then the assumptions of Lemma D.3 are satisfied. This allows us to conclude that Eq. (98) holds.

If $k+1 \neq l$, then using Eq. (97), the inequalities $j \leqslant k < k+1 < l$, and Lemma D.3, we obtain

$$\frac{\beta_k(\theta_l-\theta_k)}{\beta_k(p_l-p_k)} = \frac{\theta_l-\theta_k}{p_l-p_k} \leqslant \frac{\theta_l-\theta_j}{p_l-p_j} \quad \text{and} \quad \frac{\beta_{k+1}(\theta_l-\theta_{k+1})}{\beta_{k+1}(p_l-p_{k+1})} = \frac{\theta_l-\theta_{k+1}}{p_l-p_{k+1}} \leqslant \frac{\theta_l-\theta_j}{p_l-p_j}.$$

Note that if $a_i \in \mathbb{R}$ and $b_i \in (0,+\infty)$ for $i \in \{1,2,3\}$ satisfy $\frac{a_1}{b_1} \leqslant \frac{a_3}{b_3}$ and $\frac{a_2}{b_2} \leqslant \frac{a_3}{b_3}$, then $\frac{a_1+a_2}{b_1+b_2} \leqslant \frac{a_3}{b_3}$. Then, since $\beta_k(p_l-p_k)$, $\beta_{k+1}(p_l-p_{k+1})$ and p_l-p_j are positive, we have

$$\frac{\beta_k(\theta_l-\theta_k)+\beta_{k+1}(\theta_l-\theta_{k+1})}{\beta_k(p_l-p_k)+\beta_{k+1}(p_l-p_{k+1})} \leqslant \frac{\theta_l-\theta_j}{p_l-p_j}.$$

Hence, Eq. (98) follows directly from the inequality above and Eq. (99).

Therefore, the two conditions, including Eqs. (91) and (92), are satisfied and we apply [36, Prop 2.1] to conclude that the function u is the entropy solution to the conservation law (21).

Proof of (ii) (sufficiency): Without loss of generality, assume $p_1 < p_2 < \cdots < p_m$. Let $C \in \mathbb{R}$. Suppose $\tilde{H}$ satisfies $\tilde{H}(p_i) = H(p_i)+C$ for each $i \in \{1,\ldots,m\}$ and $\tilde{H}(p) \geqslant H(p)+C$ for any $p \in [p_1,p_m]$. We want to prove that u is the entropy solution to the conservation law (22).

As in the proof of (i), we apply [36, Prop 2.1] and verify that the two conditions hold through Eqs. (91) and (92). Let $\bar{x}(t)$ be any smooth line of discontinuity of u, define u^- and u^+ by Eq. (90) (and recall that $u^- = p_j$ and $u^+ = p_l$), and let $u_0 \in (u^-,u^+)$. We proved in the proof of (i) that $\bar{x}(t)$ is a straight line, and so it suffices to prove that

$$\frac{d\bar{x}}{dt} = \frac{\tilde{H}(u^+)-\tilde{H}(u^-)}{u^+-u^-}, \quad \text{and} \quad \frac{\tilde{H}(u^+)-\tilde{H}(u_0)}{u^+-u_0} \leqslant \frac{\tilde{H}(u^+)-\tilde{H}(u^-)}{u^+-u^-}. \tag{100}$$

We start with proving the equality in Eq. (100). By assumption, there holds

$$\begin{aligned} &\tilde{H}(u^-) = \tilde{H}(p_j) = H(p_j)+C = H(u^-)+C \quad \text{and} \\ &\tilde{H}(u^+) = \tilde{H}(p_l) = H(p_l)+C = H(u^+)+C. \end{aligned} \tag{101}$$

We combine Eq. (101) with Eq. (91), (which we proved in the proof of (i)), we obtain

$$\frac{d\bar{x}}{dt} = \frac{H(u^+)-H(u^-)}{u^+-u^-} = \frac{H(u^+)+C-(H(u^-)+C)}{u^+-u^-} = \frac{\tilde{H}(u^+)-\tilde{H}(u^-)}{u^+-u^-}.$$

Therefore, the equality in (100) holds.

Next, we prove the inequality in Eq. (100). Since $u_0 \in (u^-,u^+) \subseteq [p_1,p_m]$, by assumption there holds $\tilde{H}(u_0) \geqslant H(u_0)+C$. Taken together with Eqs. (92) and (101), we get

$$\begin{aligned} \frac{\tilde{H}(u^+)-\tilde{H}(u_0)}{u^+-u_0} &\leqslant \frac{H(u^+)+C-(H(u_0)+C)}{u^+-u_0} \\ &\leqslant \frac{H(u^+)-H(u^-)}{u^+-u^-} = \frac{\tilde{H}(u^+)-\tilde{H}(u^-)}{u^+-u^-}, \end{aligned}$$

which shows that the inequality in Eq. (100) holds.

Hence, we can invoke [36, Prop 2.1] to conclude that u is the entropy solution to the conservation law (22).

Proof of (ii) (necessity): Suppose that u is the entropy solution to the conservation law (22). We prove that there exists $C \in \mathbb{R}$ such that $\tilde{H}(p_i) = H(p_i) + C$ for any i and $\tilde{H}(p) \geqslant H(p) + C$ for any $p \in [p_1, p_m]$.

By Lemma B.2, for each $i \in \{1, \ldots, m\}$ there exist $x \in \mathbb{R}$ and $t > 0$ such that

$$f(\cdot, t) \text{ is differentiable at } x, \text{ and } \nabla_x f(x, t) = p_i. \tag{102}$$

Moreover, the proof of Lemma B.2 implies there exists $T > 0$ such that for any $0 < t < T$, there exists $x \in \mathbb{R}$ such that Eq. (102) holds. As a result, there exists $t > 0$ such that for each $i \in \{1, \ldots, m\}$, there exists $x_i \in \mathbb{R}$ satisfying Eq. (102) at the point (x_i, t), which implies $u(x_i, t) = p_i$. Note that $p_i \neq p_j$ implies that $x_i \neq x_j$. (Indeed, if $x_i = x_j$, then $p_i = \nabla_x f(x_i, t) = \nabla_x f(x_j, t) = p_j$ which gives a contradiction since $p_i \neq p_j$ by assumption (A1).) As mentioned before, the function $u(\cdot, t) \equiv \nabla_x f$ is a monotone non-decreasing operator and p_i is increasing with respect to i, and therefore $x_1 < x_2 < \cdots < x_m$. Since u is piecewise constant, for each $k \in \{1, \ldots, m-1\}$ there exists a curve of discontinuity of u with $u = p_k$ on the left-hand side of the curve and $u = p_{k+1}$ on the right-hand side of the curve. Let $\tilde{x}(s)$ be such a curve and let u^- and u^+ be the corresponding numbers defined in Eq. (90). The argument above proves that we have $u^- = p_k$ and $u^+ = p_{k+1}$.

Since u is the piecewise constant entropy solution, we invoke [36, Prop 2.1] to conclude that the two aforementioned conditions hold for the curve $\tilde{x}(s)$, i.e., (100) holds with $u^- = p_k$ and $u^+ = p_{k+1}$. From the equality in (100) and Eq. (91) proved in (i), we deduce

$$\frac{\tilde{H}(p_{k+1}) - \tilde{H}(p_k)}{p_{k+1} - p_k} = \frac{\tilde{H}(u^+) - \tilde{H}(u^-)}{u^+ - u^-} = \frac{d\tilde{x}}{dt} = \frac{H(u^+) - H(u^-)}{u^+ - u^-} = \frac{H(p_{k+1}) - H(p_k)}{p_{k+1} - p_k}.$$

Since k is an arbitrary index, the equality above implies that $\tilde{H}(p_{k+1}) - \tilde{H}(p_k) = H(p_{k+1}) - H(p_k)$ holds for any $k \in \{1, \ldots, m-1\}$. Therefore, there exists $C \in \mathbb{R}$ such that

$$\tilde{H}(p_k) = H(p_k) + C \quad \text{for any } k \in \{1, \ldots, m\}. \tag{103}$$

It remains to prove $\tilde{H}(u_0) \geqslant H(u_0) + C$ for all $u_0 \in [p_k, p_{k+1}]$. If this inequality holds, then the statement follows because k is an arbitrary index. We already proved that $\tilde{H}(u_0) \geqslant H(u_0) + C$ for $u_0 = p_k$ with $k \in \{1, \ldots, m\}$. Therefore, we need to prove that $\tilde{H}(u_0) \geqslant H(u_0) + C$ for all $u_0 \in (p_k, p_{k+1})$. Let $u_0 \in (p_k, p_{k+1})$. By Eq. (103) and the inequality in (100), we have

$$\frac{H(p_{k+1}) + C - \tilde{H}(u_0)}{p_{k+1} - u_0} = \frac{\tilde{H}(u^+) - \tilde{H}(u_0)}{u^+ - u_0} \leqslant \frac{\tilde{H}(u^+) - \tilde{H}(u^-)}{u^+ - u^-} = \frac{H(p_{k+1}) - H(p_k)}{p_{k+1} - p_k}. \tag{104}$$

By Lemma D.2 and a straightforward computation, we also have

$$\frac{H(p_{k+1}) - H(u_0)}{p_{k+1} - u_0} = \frac{H(p_{k+1}) - H(p_k)}{p_{k+1} - p_k}. \tag{105}$$

Comparing Eqs. (104) and (105), we obtain $\tilde{H}(u_0) \geqslant H(u_0) + C$. Since k is arbitrary, we conclude that $\tilde{H}(u_0) \geqslant H(u_0) + C$ holds for all $u_0 \in [p_1, p_m]$ and the proof is complete.

Received: 27 October 2019 Accepted: 22 June 2020
Published online: 14 July 2020

References

1. Aaibid, M., Sayah, A.: A direct proof of the equivalence between the entropy solutions of conservation laws and viscosity solutions of Hamilton–Jacobi equations in one-space variable. JIPAM J. Inequal. Pure Appl. Math. **7**(2), 11 (2006)
2. Akian, M., Bapat, R., Gaubert, S.: Max-plus algebra. Handbook of Linear Algebra **39**, (2006)
3. Akian, M., Gaubert, S., Lakhoua, A.: The max-plus finite element method for solving deterministic optimal control problems: basic properties and convergence analysis. SIAM J. Control Optim. **47**(2), 817–848 (2008)
4. Alla, A., Falcone, M., Saluzzi, L.: An efficient DP algorithm on a tree-structure for finite horizon optimal control problems. SIAM J. Sci. Comput. **41**(4), A2384–A2406 (2019)
5. Alla, A., Falcone, M., Volkwein, S.: Error analysis for POD approximations of infinite horizon problems via the dynamic programming approach. SIAM J. Control Optim. **55**(5), 3091–3115 (2017)
6. Arnol'd, V.I.: Mathematical methods of classical mechanics. Graduate Texts in Mathematics, vol. 60. Springer, New York (1989). Translated from the 1974 Russian original by K. Vogtmann and A. Weinstein, Corrected reprint of the second (1989) edition
7. Bachouch, A., Huré, C., Langrené, N., Pham, H.: Deep neural networks algorithms for stochastic control problems on finite horizon: numerical applications. arXiv preprint arXiv:1812.05916 (2018)
8. Banerjee, K., Georganas, E., Kalamkar, D., Ziv, B., Segal, E., Anderson, C., Heinecke, A.: Optimizing deep learning RNN topologies on intel architecture. Supercomput. Front. Innov. **6**(3), 64–85 (2019)
9. Bardi, M., Capuzzo-Dolcetta, I.: Optimal control and viscosity solutions of Hamilton–Jacobi–Bellman equations. Syst. Control Found. Appl. Birkhäuser Boston, Inc., Boston, MA (1997). https://doi.org/10.1007/978-0-8176-4755-1. With appendices by Maurizio Falcone and Pierpaolo Soravia
10. Bardi, M., Evans, L.: On Hopf's formulas for solutions of Hamilton–Jacobi equations. Nonlinear Anal. Theory, Methods Appl. **8**(11), 1373–1381 (1984). https://doi.org/10.1016/0362-546X(84)90020-8
11. Barles, G.: Solutions de viscosité des équations de Hamilton–Jacobi. Mathématiques et Applications. Springer, Berlin (1994)
12. Barles, G., Tourin, A.: Commutation properties of semigroups for first-order Hamilton–Jacobi equations and application to multi-time equations. Indiana Univ. Math. J. **50**(4), 1523–1544 (2001)
13. Barron, E., Evans, L., Jensen, R.: Viscosity solutions of Isaacs' equations and differential games with Lipschitz controls. J. Differ. Equ. **53**(2), 213–233 (1984). https://doi.org/10.1016/0022-0396(84)90040-8
14. Beck, C., Becker, S., Cheridito, P., Jentzen, A., Neufeld, A.: Deep splitting method for parabolic PDEs. (2019). arXiv preprint arXiv:1907.03452
15. Beck, C., Becker, S., Grohs, P., Jaafari, N., Jentzen, A.: Solving stochastic differential equations and Kolmogorov equations by means of deep learning. (2018). arXiv preprint arXiv:1806.00421
16. Beck, C., E, W., Jentzen, A.: Machine learning approximation algorithms for high-dimensional fully nonlinear partial differential equations and second-order backward stochastic differential equations. J. Nonlinear Sci. **29**(4), 1563–1619 (2019)
17. Bellman, R.E.: Adaptive Control Processes: A Guided Tour. Princeton University Press, Princeton (1961)
18. Berg, J., Nyström, K.: A unified deep artificial neural network approach to partial differential equations in complex geometries. Neurocomputing **317**, 28–41 (2018). https://doi.org/10.1016/j.neucom.2018.06.056
19. Bertsekas, D.P.: Reinforcement Learning and Optimal Control. Athena Scientific, Belmont (2019)
20. Bokanowski, O., Garcke, J., Griebel, M., Klompmaker, I.: An adaptive sparse grid semi-Lagrangian scheme for first order Hamilton–Jacobi Bellman equations. J. Sci. Comput. **55**(3), 575–605 (2013)
21. Bonnans, J.F., Shapiro, A.: Perturbation Analysis of Optimization Problems. Springer Series in Operations Research. Springer, New York (2000). https://doi.org/10.1007/978-1-4612-1394-9
22. Brenier, Y., Osher, S.: Approximate Riemann solvers and numerical flux functions. SIAM J. Numer. Anal. **23**(2), 259–273 (1986)
23. Brenier, Y., Osher, S.: The discrete one-sided Lipschitz condition for convex scalar conservation laws. SIAM J. Numer. Anal. **25**(1), 8–23 (1988). https://doi.org/10.1137/0725002
24. Buckdahn, R., Cardaliaguet, P., Quincampoix, M.: Some recent aspects of differential game theory. Dyn. Games Appl. **1**(1), 74–114 (2011). https://doi.org/10.1007/s13235-010-0005-0
25. Carathéodory, C.: Calculus of variations and partial differential equations of the first order. Part I: Partial differential equations of the first order. Translated by Robert B. Dean and Julius J. Brandstatter. Holden-Day, Inc., San Francisco-London-Amsterdam (1965)
26. Carathéodory, C.: Calculus of variations and partial differential equations of the first order. Part II: Calculus of variations. Translated from the German by Robert B. Dean, Julius J. Brandstatter, translating editor. Holden-Day, Inc., San Francisco-London-Amsterdam (1967)
27. Cardin, F., Viterbo, C.: Commuting Hamiltonians and Hamilton–Jacobi multi-time equations. Duke Math. J. **144**(2), 235–284 (2008). https://doi.org/10.1215/00127094-2008-036
28. Caselles, V.: Scalar conservation laws and Hamilton–Jacobi equations in one-space variable. Nonlinear Anal. Theory Methods Appl. **18**(5), 461–469 (1992). https://doi.org/10.1016/0362-546X(92)90013-5
29. Chan-Wai-Nam, Q., Mikael, J., Warin, X.: Machine learning for semi linear PDEs. J. Sci. Comput. **79**(3), 1667–1712 (2019)
30. Chen, T., van Gelder, J., van de Ven, B., Amitonov, S.V., de Wilde, B., Euler, H.C.R., Broersma, H., Bobbert, P.A., Zwanenburg, F.A., van der Wiel, W.G.: Classification with a disordered dopant-atom network in silicon. Nature **577**(7790), 341–345 (2020)
31. Cheng, T., Lewis, F.L.: Fixed-final time constrained optimal control of nonlinear systems using neural network HJB approach. In: Proceedings of the 45th IEEE Conference on Decision and Control, pp. 3016–3021 (2006). https://doi.org/10.1109/CDC.2006.377523
32. Corrias, L., Falcone, M., Natalini, R.: Numerical schemes for conservation laws via Hamilton–Jacobi equations. Math. Comput. **64**(210), 555–580, S13–S18 (1995). https://doi.org/10.2307/2153439
33. Courant, R., Hilbert, D.: Methods of mathematical physics. Vol. II. Wiley Classics Library. Wiley: New York (1989). Partial differential equations, Reprint of the 1962 original, A Wiley-Interscience Publication

34. Crandall, M.G., Ishii, H., Lions, P.L.: User's guide to viscosity solutions of second order partial differential equations. Bull. Am. Math. Soc. **27**(1), 1–67 (1992). https://doi.org/10.1090/S0273-0979-1992-00266-5
35. Cybenko, G.: Approximation by superpositions of a sigmoidal function. Math. Control Signals Syst. **2**(4), 303–314 (1989). https://doi.org/10.1007/BF02551274
36. Dafermos, C.M.: Polygonal approximations of solutions of the initial value problem for a conservation law. J. Math. Anal. Appl. **38**(1), 33–41 (1972). https://doi.org/10.1016/0022-247X(72)90114-X
37. Dafermos, C.M.: Hyperbolic conservation laws in continuum physics, *Grundlehren der Mathematischen Wissenschaften*, vol. 325, 4th Edn. Springer, Berlin (2016). https://doi.org/10.1007/978-3-662-49451-6
38. Darbon, J.: On convex finite-dimensional variational methods in imaging sciences and Hamilton–Jacobi equations. SIAM J. Imaging Sci. **8**(4), 2268–2293 (2015). https://doi.org/10.1137/130944163
39. Darbon, J., Meng, T.: On decomposition models in imaging sciences and multi-time Hamilton-Jacobi partial differential equations. (2019). arXiv preprint arXiv:1906.09502
40. Darbon, J., Osher, S.: Algorithms for overcoming the curse of dimensionality for certain Hamilton–Jacobi equations arising in control theory and elsewhere. Res. Math. Sci. **3**(1), 19 (2016). https://doi.org/10.1186/s40687-016-0068-7
41. Dissanayake, M.W.M.G., Phan-Thien, N.: Neural-network-based approximations for solving partial differential equations. Commun. Numer. Methods Eng. **10**(3), 195–201 (1994). https://doi.org/10.1002/cnm.1640100303
42. Djeridane, B., Lygeros, J.: Neural approximation of PDE solutions: An application to reachability computations. In: Proceedings of the 45th IEEE Conference on Decision and Control, pp. 3034–3039 (2006). https://doi.org/10.1109/CDC.2006.377184
43. Dockhorn, T.: A discussion on solving partial differential equations using neural networks. (2019). arXiv preprint arXiv:1904.07200
44. Dolgov, S., Kalise, D., Kunisch, K.: A tensor decomposition approach for high-dimensional Hamilton-Jacobi-Bellman equations. (2019). arXiv preprint arXiv:1908.01533
45. Dower, P.M., McEneaney, W.M., Zhang, H.: Max-plus fundamental solution semigroups for optimal control problems. In: 2015 Proceedings of the Conference on Control and its Applications, pp. 368–375. SIAM (2015)
46. Elliott, R.J.: Viscosity solutions and optimal control, Pitman research notes in mathematics series, vol. 165. Longman Scientific & Technical, Harlow; Wiley, New York (1987)
47. Evans, L.C.: Partial differential equations, Graduate Studies in Mathematics, vol. 19, second edn. American Mathematical Society, Providence, RI (2010). https://doi.org/10.1090/gsm/019
48. Evans, L.C., Gariepy, R.F.: Measure Theory and Fine Properties of Functions. Textbooks in Mathematics, revised edn. CRC Press, Boca Raton (2015)
49. Evans, L.C., Souganidis, P.E.: Differential games and representation formulas for solutions of Hamilton–Jacobi–Isaacs equations. Indiana Univ. Math. J. **33**(5), 773–797 (1984)
50. Farabet, C., LeCun, Y., Kavukcuoglu, K., Culurciello, E., Martini, B., Akselrod, P., Talay, S.: Large-scale fpga-based convolutional networks. In: Bekkerman, R., Bilenko, M., Langford, J. (eds.) Scaling up Machine Learning: Parallel and Distributed Approaches. Cambridge University Press, Cambridge (2011)
51. Farabet, C., poulet, C., Han, J., LeCun, Y.: CNP: An FPGA-based processor for convolutional networks. In: International Conference on Field Programmable Logic and Applications. IEEE, Prague (2009)
52. Farabet, C., Poulet, C., LeCun, Y.: An FPGA-based stream processor for embedded real-time vision with convolutional networks. In: 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops, pp. 878–885. IEEE Computer Society, Los Alamitos, CA, USA (2009). https://doi.org/10.1109/ICCVW.2009.5457611
53. Farimani, A.B., Gomes, J., Pande, V.S.: Deep Learning the Physics of Transport Phenomena. arXiv e-prints (2017)
54. Fleming, W., McEneaney, W.: A max-plus-based algorithm for a Hamilton–Jacobi–Bellman equation of nonlinear filtering. SIAM J. Control Optim. **38**(3), 683–710 (2000). https://doi.org/10.1137/S0363012998332433
55. Fleming, W.H., Rishel, R.W.: Deterministic and stochastic optimal control. Bull. Am. Math. Soc. **82**, 869–870 (1976)
56. Fleming, W.H., Soner, H.M.: Controlled Markov Processes and Viscosity Solutions, vol. 25. Springer, New York (2006)
57. Folland, G.B.: Real Analysis: Modern Techniques and Their Spplications. Wiley, Hoboken (2013)
58. Fujii, M., Takahashi, A., Takahashi, M.: Asymptotic expansion as prior knowledge in deep learning method for high dimensional BSDEs. Asia-Pacific Financ. Mark. **26**(3), 391–408 (2019). https://doi.org/10.1007/s10690-019-09271-7
59. Garcke, J., Kröner, A.: Suboptimal feedback control of PDEs by solving HJB equations on adaptive sparse grids. J. Sci. Comput. **70**(1), 1–28 (2017)
60. Gaubert, S., McEneaney, W., Qu, Z.: Curse of dimensionality reduction in max-plus based approximation methods: Theoretical estimates and improved pruning algorithms. In: 2011 50th IEEE Conference on Decision and Control and European Control Conference, pp. 1054–1061. IEEE (2011)
61. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, New York (2016)
62. Grohs, P., Jentzen, A., Salimova, D.: Deep neural network approximations for Monte Carlo algorithms. (2019). arXiv preprint arXiv:1908.10828
63. Grüne, L.: Overcoming the curse of dimensionality for approximating lyapunov functions with deep neural networks under a small-gain condition. (2020). arXiv preprint arXiv:2001.08423
64. Han, J., Jentzen, A., E, W.: Solving high-dimensional partial differential equations using deep learning. Proc. Natl. Acad. Sci. **115**(34), 8505–8510 (2018). https://doi.org/10.1073/pnas.1718942115
65. Han, J., Zhang, L., E, W.: Solving many-electron Schrödinger equation using deep neural networks. J. Comput. Phys. 108929 (2019)
66. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778 (2016). https://doi.org/10.1109/CVPR.2016.90
67. Hiriart-Urruty, J.B., Lemaréchal, C.: Convex Analysis and Minimization Algorithms I: Fundamentals, vol. 305. Springer, New York (1993)
68. Hiriart-Urruty, J.B., Lemaréchal, C.: Convex Analysis and Minimization Algorithms II: Advanced Theory and Bundle Methods, vol. 306. Springer, New York (1993)
69. Hirjibehedin, C.: Evolution of circuits for machine learning. Nature **577**, 320–321 (2020). https://doi.org/10.1038/d41586-020-00002-x

70. Hopf, E.: Generalized solutions of non-linear equations of first order. J. Math. Mech. **14**, 951–973 (1965)
71. Hornik, K.: Approximation capabilities of multilayer feedforward networks. Neural Netw. **4**(2), 251–257 (1991). https://doi.org/10.1016/0893-6080(91)90009-T
72. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Netw. **2**(5), 359–366 (1989). https://doi.org/10.1016/0893-6080(89)90020-8
73. Horowitz, M.B., Damle, A., Burdick, J.W.: Linear Hamilton Jacobi Bellman equations in high dimensions. In: 53rd IEEE Conference on Decision and Control, pp. 5880–5887. IEEE (2014)
74. Hsieh, J.T., Zhao, S., Eismann, S., Mirabella, L., Ermon, S.: Learning neural PDE solvers with convergence guarantees. In: International Conference on Learning Representations (2019)
75. Hu, C., Shu, C.: A discontinuous Galerkin finite element method for Hamilton–Jacobi equations. SIAM J. Sci. Comput. **21**(2), 666–690 (1999). https://doi.org/10.1137/S1064827598337282
76. Huré, C., Pham, H., Bachouch, A., Langrené, N.: Deep neural networks algorithms for stochastic control problems on finite horizon, part I: convergence analysis. (2018). arXiv preprint arXiv:1812.04300
77. Huré, C., Pham, H., Warin, X.: Some machine learning schemes for high-dimensional nonlinear PDEs. (2019). arXiv preprint arXiv:1902.01599
78. Hutzenthaler, M., Jentzen, A., Kruse, T., Nguyen, T.A.: A proof that rectified deep neural networks overcome the curse of dimensionality in the numerical approximation of semilinear heat equations. SN Partial Differ. Equ. Appl. **1**(10), (2020)
79. Hutzenthaler, M., Jentzen, A., Kruse, T., Nguyen, T.A., von Wurstemberger, P.: Overcoming the curse of dimensionality in the numerical approximation of semilinear parabolic partial differential equations (2018)
80. Hutzenthaler, M., Jentzen, A., von Wurstemberger, P.: Overcoming the curse of dimensionality in the approximative pricing of financial derivatives with default risks (2019)
81. Hutzenthaler, M., Kruse, T.: Multilevel picard approximations of high-dimensional semilinear parabolic differential equations with gradient-dependent nonlinearities. SIAM J. Numer. Anal. **58**(2), 929–961 (2020). https://doi.org/10.1137/17M1157015
82. Ishii, H.: Representation of solutions of Hamilton–Jacobi equations. Nonlinear Anal. Theory, Methods Appl. **12**(2), 121–146 (1988). https://doi.org/10.1016/0362-546X(88)90030-2
83. Jiang, F., Chou, G., Chen, M., Tomlin, C.J.: Using neural networks to compute approximate and guaranteed feasible Hamilton–Jacobi–Bellman PDE solutions. (2016). arXiv preprint arXiv:1611.03158
84. Jiang, G., Peng, D.: Weighted ENO schemes for Hamilton–Jacobi equations. SIAM J. Sci. Comput. **21**(6), 2126–2143 (2000). https://doi.org/10.1137/S106482759732455X
85. Jianyu, L., Siwei, L., Yingjian, Q., Yaping, H.: Numerical solution of elliptic partial differential equation using radial basis function neural networks. Neural Netw. **16**(5–6), 729–734 (2003)
86. Jin, S., Xin, Z.: Numerical passage from systems of conservation laws to Hamilton–Jacobi equations, and relaxation schemes. SIAM J. Numer. Anal. **35**(6), 2385–2404 (1998). https://doi.org/10.1137/S0036142996314366
87. Jouppi, N.P., Young, C., Patil, N., Patterson, D., Agrawal, G., Bajwa, R., Bates, S., Bhatia, S., Boden, N., Borchers, A., et al.: In-datacenter performance analysis of a tensor processing unit. In: Proceedings of the 44th Annual International Symposium on Computer Architecture, ISCA '17, pp. 1–12. Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3079856.3080246
88. Kalise, D., Kundu, S., Kunisch, K.: Robust feedback control of nonlinear PDEs by numerical approximation of high-dimensional Hamilton–Jacobi–Isaacs equations. (2019). arXiv preprint arXiv:1905.06276
89. Kalise, D., Kunisch, K.: Polynomial approximation of high-dimensional Hamilton–Jacobi–Bellman equations and applications to feedback control of semilinear parabolic PDEs. SIAM J. Sci. Comput. **40**(2), A629–A652 (2018)
90. Kang, W., Wilcox, L.C.: Mitigating the curse of dimensionality: sparse grid characteristics method for optimal feedback control and HJB equations. Comput. Optim. Appl. **68**(2), 289–315 (2017)
91. Karlsen, K., Risebro, H.: A note on front tracking and the equivalence between viscosity solutions of Hamilton–Jacobi equations and entropy solutions of scalar conservation laws. Nonlinear Anal. (2002). https://doi.org/10.1016/S0362-546X(01)00753-2
92. Khoo, Y., Lu, J., Ying, L.: Solving parametric PDE problems with artificial neural networks. (2017). arXiv preprint arXiv:1707.03351
93. Khoo, Y., Lu, J., Ying, L.: Solving for high-dimensional committor functions using artificial neural networks. Res. Math. Sci. **6**(1), 1 (2019)
94. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. In: Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015) (2015)
95. Kružkov, S.N.: Generalized solutions of nonlinear first order equations with several independent variables II. Math. USSR-Sbornik **1**(1), 93–116 (1967). https://doi.org/10.1070/sm1967v001n01abeh001969
96. Kundu, A., Srinivasan, S., Qin, E.C., Kalamkar, D., Mellempudi, N.K., Das, D., Banerjee, K., Kaul, B., Dubey, P.: K-tanh: Hardware efficient activations for deep learning (2019)
97. Kunisch, K., Volkwein, S., Xie, L.: HJB-POD-based feedback design for the optimal control of evolution problems. SIAM J. Appl. Dyn. Syst. **3**(4), 701–722 (2004)
98. Lagaris, I.E., Likas, A., Fotiadis, D.I.: Artificial neural networks for solving ordinary and partial differential equations. IEEE Trans. Neural Netw. **9**(5), 987–1000 (1998). https://doi.org/10.1109/72.712178
99. Lagaris, I.E., Likas, A.C., Papageorgiou, D.G.: Neural-network methods for boundary value problems with irregular boundaries. IEEE Trans. Neural Netw. **11**(5), 1041–1049 (2000). https://doi.org/10.1109/72.870037
100. Lambrianides, P., Gong, Q., Venturi, D.: A new scalable algorithm for computational optimal control under uncertainty. (2019). arXiv preprint arXiv:1909.07960
101. Landau, L., Lifschic, E.: Course of theoretical physics. vol. 1: Mechanics. Oxford, (1978)
102. LeCun, Y.: 1.1 deep learning hardware: Past, present, and future. In: 2019 IEEE International Solid-State Circuits Conference—(ISSCC), pp. 12–19 (2019). https://doi.org/10.1109/ISSCC.2019.8662396
103. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015)
104. Lee, H., Kang, I.S.: Neural algorithm for solving differential equations. J. Comput. Phys. **91**(1), 110–131 (1990)

105. Lions, P.L., Rochet, J.C.: Hopf formula and multitime Hamilton–Jacobi equations. Proc. Am. Math. Soc. **96**(1), 79–84 (1986)
106. Lions, P.L., Souganidis, P.E.: Convergence of MUSCL and filtered schemes for scalar conservation laws and Hamilton–Jacobi equations. Numerische Mathematik **69**(4), 441–470 (1995). https://doi.org/10.1007/s002110050102
107. Long, Z., Lu, Y., Dong, B.: PDE-net 2.0: Learning PDEs from data with a numeric-symbolic hybrid deep network. J. Comput. Phys. **399**, 108925 (2019). https://doi.org/10.1016/j.jcp.2019.108925
108. Long, Z., Lu, Y., Ma, X., Dong, B.: PDE-net: Learning PDEs from data. (2017). arXiv preprint arXiv:1710.09668
109. Lye, K.O., Mishra, S., Ray, D.: Deep learning observables in computational fluid dynamics. (2019). arXiv preprint arXiv:1903.03040
110. McEneaney, W.: Max-Plus Methods for Nonlinear Control and Estimation. Springer, New York (2006)
111. McEneaney, W.: A curse-of-dimensionality-free numerical method for solution of certain HJB PDEs. SIAM J. Control Optim. **46**(4), 1239–1276 (2007). https://doi.org/10.1137/040610830
112. McEneaney, W.M., Deshpande, A., Gaubert, S.: Curse-of-complexity attenuation in the curse-of-dimensionality-free method for HJB PDEs. In: 2008 American Control Conference, pp. 4684–4690. IEEE (2008)
113. McEneaney, W.M., Kluberg, L.J.: Convergence rate for a curse-of-dimensionality-free method for a class of HJB PDEs. SIAM J. Control Optim. **48**(5), 3052–3079 (2009)
114. McFall, K.S., Mahan, J.R.: Artificial neural network method for solution of boundary value problems with exact satisfaction of arbitrary boundary conditions. IEEE Trans. Neural Netw. **20**(8), 1221–1233 (2009). https://doi.org/10.1109/TNN.2009.2020735
115. Meade, A., Fernandez, A.: The numerical solution of linear ordinary differential equations by feedforward neural networks. Math. Comput. Modell. **19**(12), 1–25 (1994). https://doi.org/10.1016/0895-7177(94)90095-7
116. Meng, X., Karniadakis, G.E.: A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse PDE problems. (2019). arXiv preprint arXiv:1903.00104
117. Meng, X., Li, Z., Zhang, D., Karniadakis, G.E.: PPINN: Parareal physics-informed neural network for time-dependent PDEs. (2019). arXiv preprint arXiv:1909.10145
118. van Milligen, B.P., Tribaldos, V., Jiménez, J.A.: Neural network differential equation and plasma equilibrium solver. Phys. Rev. Lett. **75**, 3594–3597 (1995). https://doi.org/10.1103/PhysRevLett.75.3594
119. Motta, M., Rampazzo, F.: Nonsmooth multi-time Hamilton–Jacobi systems. Indiana Univ. Math. J. **55**(5), 1573–1614 (2006)
120. Niarchos, K.N., Lygeros, J.: A neural approximation to continuous time reachability computations. In: Proceedings of the 45th IEEE Conference on Decision and Control, pp. 6313–6318 (2006). https://doi.org/10.1109/CDC.2006.377358
121. Osher, S., Shu, C.: High-order essentially nonoscillatory schemes for Hamilton–Jacobi equations. SIAM J. Numer. Anal. **28**(4), 907–922 (1991). https://doi.org/10.1137/0728049
122. Pang, G., Lu, L., Karniadakis, G.E.: fPINNs: Fractional physics-informed neural networks. SIAM J. Sci. Comput. **41**(4), A2603–A2626 (2019)
123. Pham, H., Pham, H., Warin, X.: Neural networks-based backward scheme for fully nonlinear PDEs. (2019). arXiv preprint arXiv:1908.00412
124. Pinkus, A.: Approximation theory of the MLP model in neural networks. In: Acta numerica, 1999, *Acta Numer.*, vol. 8, pp. 143–195. Cambridge University Press, Cambridge (1999)
125. Plaskacz, S., Quincampoix, M.: Oleinik–Lax formulas and multitime Hamilton–Jacobi systems. Nonlinear Anal. Theory, Methods Appl. **51**(6), 957–967 (2002). https://doi.org/10.1016/S0362-546X(01)00871-9
126. Raissi, M.: Deep hidden physics models: Deep learning of nonlinear partial differential equations. J. Mach. Learn. Res. **19**(1), 932–955 (2018)
127. Raissi, M.: Forward-backward stochastic neural networks: Deep learning of high-dimensional partial differential equations. (2018). arXiv preprint arXiv:1804.07010
128. Raissi, M., Perdikaris, P., Karniadakis, G.: Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J. Comput. Phys. **378**, 686–707 (2019). https://doi.org/10.1016/j.jcp.2018.10.045
129. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations. (2017). arXiv preprint arXiv:1711.10561
130. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Physics informed deep learning (part ii): Data-driven discovery of nonlinear partial differential equations. (2017). arXiv preprint arXiv:1711.10566
131. Reisinger, C., Zhang, Y.: Rectified deep neural networks overcome the curse of dimensionality for nonsmooth value functions in zero-sum games of nonlinear stiff systems. (2019). arXiv preprint arXiv:1903.06652
132. Rochet, J.: The taxation principle and multi-time Hamilton–Jacobi equations. J. Math. Econ. **14**(2), 113–128 (1985). https://doi.org/10.1016/0304-4068(85)90015-1
133. Rockafellar, R.T.: Convex Analysis. Princeton University Press, Princeton (1970)
134. Royo, V.R., Tomlin, C.: Recursive regression with neural networks: Approximating the HJI PDE solution. (2016). arXiv preprint arXiv:1611.02739
135. Rudd, K., Muro, G.D., Ferrari, S.: A constrained backpropagation approach for the adaptive solution of partial differential equations. IEEE Trans. Neural Netw. Learn. Syst. **25**(3), 571–584 (2014). https://doi.org/10.1109/TNNLS.2013.2277601
136. Ruthotto, L., Osher, S., Li, W., Nurbekyan, L., Fung, S.W.: A machine learning framework for solving high-dimensional mean field game and mean field control problems. (2019). arXiv preprint arXiv:1912.01825
137. Schmidhuber, J.: Deep learning in neural networks: an overview. Neural Netw. **61**, 85–117 (2015). https://doi.org/10.1016/j.neunet.2014.09.003
138. Sirignano, J., Spiliopoulos, K.: DGM: A deep learning algorithm for solving partial differential equations. J. Comput. Phys. **375**, 1339–1364 (2018). https://doi.org/10.1016/j.jcp.2018.08.029
139. Tang, W., Shan, T., Dang, X., Li, M., Yang, F., Xu, S., Wu, J.: Study on a Poisson's equation solver based on deep learning technique. In: 2017 IEEE Electrical Design of Advanced Packaging and Systems Symposium (EDAPS), pp. 1–3 (2017). https://doi.org/10.1109/EDAPS.2017.8277017

140. Tassa, Y., Erez, T.: Least squares solutions of the HJB equation with neural network value-function approximators. IEEE Trans. Neural Netw. **18**(4), 1031–1041 (2007). https://doi.org/10.1109/TNN.2007.899249
141. Tho, N.: Hopf-Lax-Oleinik type formula for multi-time Hamilton–Jacobi equations. Acta Math. Vietnamica **30**, 275–287 (2005)
142. Todorov, E.: Efficient computation of optimal actions. Proc. Natl. Acad. Sci. **106**(28), 11478–11483 (2009)
143. Uchiyama, T., Sonehara, N.: Solving inverse problems in nonlinear PDEs by recurrent neural networks. In: IEEE International Conference on Neural Networks, pp. 99–102. IEEE (1993)
144. E, W., Yu, B.: The deep Ritz method: a deep learning-based numerical algorithm for solving variational problems. Commun. Math. Stat. **6**(1), 1–12 (2018)
145. E, W., Han, J., Jentzen, A.: Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations. Commun. Math. Stat. **5**(4), 349–380 (2017). https://doi.org/10.1007/s40304-017-0117-6
146. E, W., Hutzenthaler, M., Jentzen, A., Kruse, T.: Multilevel picard iterations for solving smooth semilinear parabolic heat equations (2016)
147. Widder, D.V.: The Heat Equation, vol. 67. Academic Press, New York (1976)
148. Yadav, N., Yadav, A., Kumar, M.: An introduction to neural network methods for differential equations. SpringerBriefs in Applied Sciences and Technology. Springer, Dordrecht (2015). https://doi.org/10.1007/978-94-017-9816-7
149. Yang, L., Zhang, D., Karniadakis, G.E.: Physics-informed generative adversarial networks for stochastic differential equations. (2018). arXiv preprint arXiv:1811.02033
150. Yang, Y., Perdikaris, P.: Adversarial uncertainty quantification in physics-informed neural networks. J. Comput. Phys. **394**, 136–152 (2019)
151. Yegorov, I., Dower, P.M.: Perspectives on characteristics based curse-of-dimensionality-free numerical approaches for solving Hamilton–Jacobi equations. Appl. Math. Optim. 1–49 (2017)
152. Zhang, D., Guo, L., Karniadakis, G.E.: Learning in modal space: solving time-dependent stochastic PDEs using physics-informed neural networks. (2019). arXiv preprint arXiv:1905.01205
153. Zhang, D., Lu, L., Guo, L., Karniadakis, G.E.: Quantifying total uncertainty in physics-informed neural networks for solving forward and inverse stochastic problems. J. Comput. Phys. **397**, 108850 (2019)

Publisher's Note