NEURAL IMPLICIT SOLUTION FORMULA FOR EFFICIENTLY SOLVING HAMILTON-JACOBI EQUATIONS*

YESOM PARK† AND STANLEY OSHER‡

Abstract. This paper presents an implicit solution formula for the Hamilton-Jacobi partial differential equation (HJ PDE). The formula is derived using the method of characteristics and is shown to coincide with the Hopf and Lax formulas in the case where either the Hamiltonian or the initial function is convex. It provides a simple and efficient numerical approach for computing the viscosity solution of HJ PDEs, bypassing the need for the Legendre transform of the Hamiltonian or the initial condition, and the explicit computation of individual characteristic trajectories. A deep learning-based methodology is proposed to learn this implicit solution formula, leveraging the mesh-free nature of deep learning to ensure scalability for high-dimensional problems. Building upon this framework, an algorithm is developed that approximates the characteristic curves piecewise linearly for state-dependent Hamiltonians. Extensive experimental results demonstrate that the proposed method delivers highly accurate solutions, even for nonconvex Hamiltonians, and exhibits remarkable scalability, achieving computational efficiency for problems up to 40 dimensions.

Key words. Hamilton-Jacobi equations, implicit solution formula, deep learning

MSC codes. 35C99, 65M25, 68T07

1. Introduction.

Hamilton-Jacobi partial differential equations (HJ PDEs) are of paramount importance in various fields of mathematics, physics, and engineering, including optimal control [27, 66, 4], mechanics [23, 21], and the study of dynamic systems [41, 65]. As they provide a powerful framework for modeling systems governed by physical laws, HJ PDEs have a wide range of applications in diverse areas such as geometric optics [56, 53], computer vision [8, 32, 58], robotics [48, 46, 3], trajectory optimization [22, 61], traffic flow modeling [34, 45], and financial strategies [31, 7]. These applications illustrate the versatility and significance of HJ PDEs, emphasizing the necessity for effective methods to solve them in both theoretical and practical contexts. It is well-known that the solutions to HJ PDEs are typically continuous but exhibit discontinuous derivatives, irrespective of the smoothness of the initial conditions or the Hamiltonian. Moreover, such solutions are typically non-unique. In this regard, viscosity solutions [14] are commonly considered as the appropriate notion of solution, as they reflect the physical characteristics inherent to the problem.

Numerical methods for solving HJ PDEs have been extensively developed, with numerous practical applications across various fields. The most prominent methods include essentially non-oscillatory (ENO) and weighted ENO (WENO) type schemes [60, 35, 6, 63], semi-Lagrangian methods [28, 15, 29], and level set approaches [59, 56, 57, 54, 2]. However, they encounter significant scalability challenges as the dimensionality of the state space increases. These methods rely on discretization of the state space with a grid and approximating the Hamiltonian in a discrete form. Consequently, the number of grid points required to obtain accurate solutions grows exponentially with the dimensionality of the problem, resulting in prohibitive computational costs. In high-dimensional settings, particularly those involving more than four dimensions, this scaling issue renders the classical methods impractical for many real-world applications, where high-dimensional state spaces are prevalent.

*Submitted to the editors DATE.
Funding: This work was funded by .
†Department of Mathematics, University of California (yeisom@math.ucla.edu).
‡Department of Mathematics, University of California (sjo@math.ucla.edu).

Several approaches have been proposed to address the curse of dimensionality in solving HJ PDEs. Methods based on max-plus algebra [52, 1, 30] show promise but are restricted to specific classes of optimal control problems and encounter significant challenges in practical implementation due to their complexity. Another promising approach involves the use of Hopf or Lax formulas to represent solutions to HJ PDEs [20, 12, 13, 10]. These formulas offer a causality-free approach, where solutions at each spatial and temporal point can be computed by solving an optimization problem, thus enabling parallel computation. This approach eliminates the reliance on grid-based discretization, making it particularly well-suited for high-dimensional problems. However, these methods require computing the Legendre transform of the Hamiltonian or initial function and are generally applicable only under specific assumptions, such as convexity, or when the problem can be framed as a particular type of control problem. In parallel, algorithms based on Pontryagin's Maximum Principle [38, 39, 73], which employ the method of characteristics, have been proposed. Despite their potential, the practical effectiveness of these methods is often limited by the need to solve a system of ordinary differential equations (ODEs) at each point. Additionally, some of these methods assume that multiple characteristics do not intersect, a condition that may not hold in general scenarios. Furthermore, alternative techniques, such as tensor decomposition [24] and polynomial approximation [37, 36], have been studied for specific control problems.

Recent advancements in deep learning have given rise to a growing interest in leveraging the extensive representational capabilities of neural networks to solve PDEs [67, 74, 64, 51, 47, 72]. The viscosity solution of HJ PDEs is challenging to obtain directly from the PDE itself, which underscores the development of alternative approaches beyond the established methods like physics-informed neural networks (PINNs) [64]. In response, data-driven methods have been proposed for solving HJ PDEs [55, 25, 16]; however, these methods face several challenges, including the need for large amounts of training data, the limitation that their performance cannot exceed the accuracy of the numerical methods used to generate the data, and concerns regarding their ability to generalize to unseen scenarios. Moreover, the integration of reinforcement learning techniques to solve HJ PDEs related to control problems has been studied [76, 49]. Another line of research has focused on the development of specialized neural network architectures that express representation formulas to specific HJ PDEs [18, 19, 17]. One of the most closely related prior works introduces a deep learning approach for learning implicit solution formulas along with characteristics for scalar conservation laws associated with one-dimensional HJ PDEs [75]. However, this method does not ensure the attainment of an entropy solution.

This study presents a novel implicit solution formula for HJ PDEs. The proposed implicit formula is derived through the characteristics of the HJ PDE, with the costate identified as the gradient of the solution at the current spatio-temporal point, leading to an implicit representation formula for the solution. We demonstrate that this new formula coincides with the classical Hopf and Lax formulas, which provides the viscosity solution for HJ PDEs in the case where either the Hamiltonian or the initial function is convex. Notably, the implicit formula is simpler than both the Hopf and Lax formulas, as it does not require the Legendre transform of either the Hamiltonian or the initial function, thereby broadening its practical applicability. Furthermore, although being based on characteristics, the implicit formula alleviates the need to solve the system of characteristic ODEs from the initial state to the present time. From an optimal control perspective, we further explore the connection of the proposed formula with the Pontryagin's maximum principle and Bellman's principle, showing that the proposed implicit solution formula serves as an implicit representation of Bellman's principle.

Building on this foundation, we propose a deep learning-based approach to solve HJ PDEs by learning the implicit solution formula. This method approximates the solution as a Lipschitz continuous function, leveraging the powerful expressive capacity of neural networks. Unlike traditional grid-based methods, our approach does not require discretization of the domain, making it highly scalable and efficient, especially for high-dimensional problems. This effectively mitigates the curse of dimensionality, ensuring that computational time and memory usage scale efficiently with dimensionality. Thanks to the inherent simplicity of the implicit solution formula, it obviates the need for computing the Legendre transform and individual characteristic trajectories, thereby enhancing both its applicability and computational efficiency across a wide range of problems. Through extensive and rigorous experimentation, we show that the proposed algorithm provides accurate solutions even for problems with up to 40 dimensions with negligible increases in computational cost. Importantly, the method also shows robust performance on various nonconvex HJ PDEs, for which mathematical demonstration has not been established, underscoring its versatility and potential.

We extend our approach to handle HJ PDEs with state-dependent Hamiltonians. In such cases, where the characteristic curves are no longer linear, deriving an implicit solution formula becomes more intricate. To address this, we approximate the characteristic curves as piecewise linear segments over short time intervals, applying the proposed implicit solution formula at each interval. This leads to an efficient time-marching algorithm that can handle state-dependent Hamiltonians, which we validate through a series of experiments involving high-dimensional optimal control problems. The results demonstrate that the proposed method is not only simple and efficient but also effectively solving a wide range of high-dimensional, nonconvex HJ PDEs, highlighting its potential as a valuable tool for addressing complex optimal control problems and dynamic systems.

2. Implicit Solution formula of Hamilton-Jacobi Equations.

2.1. Implicit Solution Formula along Characteristics. In this subsection, we introduce a novel implicit solution formula for the Hamilton-Jacobi partial differential equation (HJ PDE) defined in a domain $\Omega \subset \mathbb{R}^d$:

$$\begin{cases} u_t + H(\nabla u) = 0 & \text{in } \Omega \times (0, T) \\ u = g & \text{on } \Omega \times \{t = 0\}, \end{cases} \tag{2.1}$$

where $H : \mathbb{R}^d \to \mathbb{R}$ is the Hamiltonian and $g : \Omega \to \mathbb{R}$ is the initial function. System of characteristic ODEs for (2.1), also known as Hamilton's system, is given by the following:

$$\begin{cases} \dot{\mathbf{x}} = \nabla H & \text{(2.2a)} \\ \dot{u} = q + \mathbf{p}^{\mathrm{T}} \nabla H = -H + \mathbf{p}^{\mathrm{T}} \nabla H & \text{(2.2b)} \\ \dot{q} = 0 & \text{(2.2c)} \\ \dot{\mathbf{p}} = 0, & \text{(2.2d)} \end{cases}$$

where the variables q and $\mathbf{p}$ are shorthand for the partial derivatives $q = u_t$ and $\mathbf{p} = \nabla u$. From (2.2d) it is clear that the value of $\mathbf{p}$, which is the sole argument of the

Hamiltonian, remains constant along the characteristic. Therefore, the characteristic emanated from $\mathbf{x}(0) = \mathbf{x}_0 \in \Omega$ is a straight line

$$\mathbf{x}(t) = t\nabla H(\mathbf{p}) + \mathbf{x}_0,$$

and

$$\begin{aligned} u(t, \mathbf{x}(t)) &= -tH(\mathbf{p}) + t\mathbf{p}^{\mathrm{T}}\nabla H(\mathbf{p}) + u(\mathbf{x}_0, 0) \\ &= -tH(\mathbf{p}) + t\mathbf{p}^{\mathrm{T}}\nabla H(\mathbf{p}) + g(\mathbf{x}_0). \end{aligned}$$

Given the constant nature of $\mathbf{p}$ along each characteristic line, its value can be determined at any intermediate time between the initial and current times. In this context, we adopt $\mathbf{p}$ as the gradient of the solution at the current time. Substituting $\mathbf{p} = \nabla u$ and expressing $\mathbf{x}(t) = \mathbf{x} \in \Omega$ induces that

$$\mathbf{x}_0 = \mathbf{x} - t\nabla H(\nabla u(\mathbf{x}, t)),$$

and hence we attain the following **implicit solution formula** for HJ PDEs (2.1):

$$u(\mathbf{x}, t) = -tH(\nabla u) + t\nabla u^{\mathrm{T}}\nabla H(\nabla u) + g(\mathbf{x} - t\nabla H(\nabla u)). \tag{2.3}$$

It is worth noting that this implicit formula expresses the solution without requiring the Legendre transform of the Hamiltonian H or the initial function g. Moreover, it does not require to compute individual characteristic trajectories by solving the system of characteristic ODEs. Therefore, it provides a highly practical and straightforward approach to solving HJ PDEs. Building upon this formula, we propose a highly simple and effective deep learning-based methodology for solving HJ PDEs in section 3.

A key distinction between conventional approaches based on characteristics and the proposed implicit solution formula lies in the treatment of $\mathbf{p}$, which is chosen as the gradient of the solution ∇u at the current time t. Since $\mathbf{p}$ remains constant along each characteristic line, it can be readily determined from the initial data. Consequently, conventional methods typically express $\mathbf{p}$ in terms of $\nabla g(\mathbf{x}_0)$. However, these approaches are limited in situations where no characteristic traces back to the initial time $t = 0$, resulting in the gradient at the current time not being accessible from the initial data. In contrast, our approach employs the current value of $\mathbf{p}(t) = \nabla u(\mathbf{x}, t)$, allowing the implicit solution formula to effectively handle such scenarios.

It is well-established that under certain assumptions on the Hamiltonian H and the initial function g, a representation formula for the viscosity solution can be derived. The first is the *Hopf-Lax formula*

$$u(\mathbf{x}, t) = \inf_{\mathbf{y}} \left\{ tH^* \left(\frac{\mathbf{x} - \mathbf{y}}{t} \right) + g(\mathbf{y}) \right\}, \tag{2.4}$$

which holds for convex (or concave) H and Lipschitz g [33, 5, 50], or for Lipschitz and convex H and continuous g [68], or also for strictly convex H and lower semicontinuous (l.s.c.) g [42, 43]. Here, where H^* is the Legendre transforms of H. On the other hand, *Hopf formula*

$$u(\mathbf{x}, t) = -\inf_{\mathbf{z}} \left\{ g^*(\mathbf{z}) + tH(\mathbf{z}) - \mathbf{x}^{\mathrm{T}}\mathbf{z} \right\} \tag{2.5}$$

is valid for Lipschitz and convex (or concave) g and merely continuous H [33, 5], or for convex g and Lipschitz H [68]. In the following, we demonstrate that the proposed implicit solution formula (2.3) represents these two respective formulas under the conditions under which they hold.

THEOREM 2.1. *Assume the Hamiltonian H is differentiable and satisfies*

$$\begin{cases} \mathbf{p} \mapsto H(\mathbf{p}) \ \textit{is strictly convex}, \\ \lim_{|\mathbf{p}|\to\infty} \frac{H(\mathbf{p})}{|\mathbf{p}|} = +\infty, \end{cases} \tag{2.6}$$

and the initial function g is l.s.c. Then, the continuous function u that satisfies the implicit solution formula (2.3) *is the viscosity solution of* (2.1) *a.e.*

Proof. First, we can observe that the implicit solution formula (2.3) exactly satisfies the initial condition $u = g$ of (2.1) at the initial time $t = 0$.

Under the assumptions, the viscosity solution of the HJ PDE is described by the Hopf-Lax formula (2.4). By expanding the Legendre transform in the Hopf-Lax formula (2.1), the viscosity solution is expressed as follows

$$u(\mathbf{x},t) = \inf_{\mathbf{y}} \sup_{\mathbf{z}} \left\{ t\left(\mathbf{z}^{\mathrm{T}}\left(\frac{\mathbf{x}-\mathbf{y}}{t}\right) - H(\mathbf{z})\right) + g(\mathbf{y}) \right\} \tag{2.7}$$

$$= \inf_{\mathbf{y}} \sup_{\mathbf{z}} \left\{ \mathbf{z}^{\mathrm{T}}(\mathbf{x}-\mathbf{y}) - tH(\mathbf{z}) + g(\mathbf{y}) \right\}, \tag{2.8}$$

The Euler-Lagrange equation of Hopf-Lax formula leads to

$$\mathbf{y}^\star = \operatorname*{argmin}_{\mathbf{y}} \left\{ tH^*\left(\frac{\mathbf{x}-\mathbf{y}}{t}\right) + g(\mathbf{y}) \right\} = \mathbf{x} - t\nabla H(\nabla u). \tag{2.9}$$

Furthermore, differentiating (2.8) with respect to $\mathbf{z}$ provides that the optimal $\mathbf{z}^\star$ satisfies

$$\mathbf{x} - \mathbf{y}^\star - t\nabla H(\mathbf{z}^\star) = 0.$$

Together with (2.9), we have

$$\mathbf{z}^\star = \nabla u. \tag{2.10}$$

Substituting these (2.9) and (2.10) into (2.8) results in the implicit formula (2.3). ∎

THEOREM 2.2. *Assume the initial function g satisfies*

$$\begin{cases} \mathbf{x} \mapsto g(\mathbf{x}) \ \textit{is convex}, \\ \lim_{|\mathbf{x}|\to\infty} \frac{g(\mathbf{x})}{|\mathbf{x}|} = +\infty, \end{cases} \tag{2.11}$$

that the Hamiltonian H is continuous, and that either the H or g is Lipschitz continuous. Then, the continuous function u that satisfies the implicit solution formula (2.3) *is the viscosity solution of* (2.1) *a.e.*

Proof. Since the viscosity solution is described by the Hopf formula (2.4) under these assumptions, it can be written as follows:

$$u(\mathbf{x},t) = -g^*(\mathbf{z}^\star) - tH(\mathbf{z}^\star) + \mathbf{x}^{\mathrm{T}}\mathbf{z}^\star \tag{2.12}$$

$$= \inf_{\mathbf{y}} \left\{ \mathbf{z}^{\star\mathrm{T}}(\mathbf{x}-\mathbf{y}) - tH(\mathbf{z}^\star) + g(\mathbf{y}) \right\} \tag{2.13}$$

$$= \mathbf{z}^{\star\mathrm{T}}(\mathbf{x}-\mathbf{y}^\star) - tH(\mathbf{z}^\star) + g(\mathbf{y}^\star). \tag{2.14}$$

Differentiating the both side of (2.12) with respect to $\mathbf{x}$ induces

$$\frac{\partial u}{\partial \mathbf{x}} = -\frac{\partial}{\partial \mathbf{z}}\left\{ g^*(\mathbf{z}^\star) + tH(\mathbf{z}^\star) \right\} \cdot \frac{\partial \mathbf{z}^\star}{\partial \mathbf{x}} + \mathbf{z}^\star = \mathbf{z}^\star,$$

where the last equality follows from the optimality of $\mathbf{z}^\star$. Consequently, we have $\mathbf{z}^\star = \nabla u$. Furthermore, differentiating (2.13) with respect to $\mathbf{z}$ provides that the optimal $\mathbf{y}^\star$ satisfies

$$\mathbf{x} - \mathbf{y}^\star - t\nabla H\left(\mathbf{z}^\star\right) = 0,$$

that is,

$$\mathbf{y}^\star = \mathbf{x} - t\nabla H\left(\mathbf{z}^\star\right) = \mathbf{x} - t\nabla H\left(\nabla u\right),$$

which concludes the proof. ∎

Theorems 2.1 and 2.2 offers the theoretical validation of the implicit solution formula (2.3) under the assumption of convexity of the Hamiltonian H or the initial function g. However, this result has not yet been extended to the nonconvex case. Nonetheless, as illustrated in Subsubsection 4.2, we present robust empirical evidence demonstrating the performance of the proposed approach through extensive experiments on a diverse range of nonconvex examples, where neither the Hamiltonian nor the initial function is convex. These results suggest the potential applicability and validity of the proposed formula in such scenarios.

To facilitate comprehension of the implicit solution formula, a simple example is presented.

EXAMPLE 2.1. *Consider a one-dimensional example with a quadratic Hamiltonian and a homogeneous initial condition:*

$$\begin{cases} u_t + u_x^2 = 0 & \text{in } \mathbb{R} \times (0, \infty) \\ u = 0 & \text{on } \mathbb{R} \times \{t = 0\}. \end{cases} \tag{2.15}$$

The viscosity solution to this problem is $u^* = 0$. Note that there are infinitely many Lipschitz functions satisfying (2.15) *a.e. [26], for instance,*

$$v\left(x,t\right) = \begin{cases} 0 & \text{if } \mid x \mid \geq t \\ x - t & \text{if } 0 \leq x \leq t \\ x - t & \text{if } -t \leq x \leq 0. \end{cases}$$

This example shows that, although there are infinitely many Lipschitz functions that satisfy the HJ PDE, the implicit solution formula uniquely characterizes the viscosity solution. The implicit solution formula (2.3) *corresponding to* (2.15) *is written as*

$$u = tu_x. \tag{2.16}$$

For $t = 0$, (2.16) *satisfies the initial condition $u = 0$. For a fixed time $t > 0$,* (2.16) *represents an ordinary differential equation (ODE) with a coefficient that depends on t, and the ODE admits infinitely many solutions*

$$u = Ce^{tx}, \ \forall C \in \mathbb{R}.$$

However, in order to satisfy the initial condition $u = 0$ at $t = 0$, it follows that C must be zero. Therefore, the viscosity solution $u^* = 0$ is the unique continuous function that satisfies the implicit solution formula (2.16).

This example illustrates that, despite the existence of an infinitely many weak solutions to the governing HJ PDE, the continuous function that satisfies the implicit solution formula (2.3) is the unique viscosity solution. However, it also suggests that, at a fixed time $t > 0$, the implicit formula may admit multiple solutions. For a fixed $t > 0$, the implicit solution formula (2.3) describes a first-order nonlinear static PDE (or an ODE in the one-dimensional case) in $\mathbf{x}$, where the time variable t appears as coefficients. The absence of boundary conditions in this static PDE at fixed $t > 0$ naturally leads to the ill-posedness of the PDE with multiple solutions. Therefore, the condition that the implicit formula (2.3) satisfies the initial condition at $t = 0$ is crucial, and finding a continuous function that satisfies the implicit solution formula across the entire spacetime domain from the initial data is essential for obtaining the unique viscosity solution. It is noteworthy, however, that the above example is taken in the unbounded spatial domain $\mathbb{R}$. For the general case of HJ PDEs on a bounded domain Ω, boundary conditions are specified. In such cases, the given boundary condition serves as the boundary condition for the static PDE described by (2.3) at a fixed time, thereby ensuring the uniqueness of the solution.

Remark 2.3. (Level set propagation) If the Hamiltonian H is homogeneous of degree one in its gradient argument, i.e., H takes of the form with a function $\mathbf{v} : \mathbb{R}^d \to \mathbb{R}^d$

$$H\left(\nabla u\right) = \mathbf{v}\left(\frac{\nabla u}{\|\nabla u\|}\right)^{\mathrm{T}} \nabla u, \tag{2.17}$$

then the implicit formula (2.3) comes down to the following simple formula a.e.

$$u\left(\mathbf{x}, t\right) = g\left(\mathbf{x} - t\mathbf{v}\left(\frac{\nabla u}{\|\nabla u\|}\right)\right), \tag{2.18}$$

where the solution u is constant along the characteristics.

2.2. Control Perspectives on the Implicit Solution Formula. In this subsection, we revisit the implicit solution formula (2.3) from the perspective of control theory, elucidating that it represents an implicit formulation of Bellman's principle. This perspective also enables a comprehensive exploration of the relationships between the implicit solution formula and Pontryagin's Maximum Principle (PMP) and the Hopf-Lax formula (2.4), while also highlighting the distinctions between these established approaches and the proposed implicit formula.

Let $L = L\left(\mathbf{q}\right) : \mathbb{R}^d \to \mathbb{R}$ be the corresponding Lagrangian, that is, $L = H^*$, the Legendre transform of H. Under the assumption (2.6) on the Hamiltonian H, it satisfies

$$\begin{cases} \mathbf{q} \mapsto L\left(\mathbf{q}\right) \text{ is convex,} \\ \lim_{|\mathbf{q}| \to \infty} \frac{L(\mathbf{q})}{|\mathbf{q}|} = +\infty, \end{cases}$$

and $H\left(\mathbf{p}\right) = L^*\left(\mathbf{p}\right) = \sup_{\mathbf{q}} \left\{\mathbf{p}^{\mathrm{T}}\mathbf{q} - L\left(\mathbf{q}\right)\right\}$.

It is well-known that the viscosity solution u of the HJ PDE

$$\begin{cases} u_t + H\left(\nabla u\right) = u_t + \sup_{\mathbf{q}} \left\{\nabla u^{\mathrm{T}}\mathbf{q} - L\left(\mathbf{q}\right)\right\} = 0, \\ u\left(\mathbf{x}, 0\right) = g\left(\mathbf{x}\right) \end{cases} \tag{2.19}$$

is represented by the value function of the following corresponding optimal control problem:

$$u(\mathbf{x},t)=\inf_{\mathbf{q}}\left\{\int_0^t L(\mathbf{q}(s))\,\mathrm{d}s+g(\mathbf{y}(0)):\mathbf{y}(t)=\mathbf{x},\dot{\mathbf{y}}(s)=\mathbf{q}(s),0\le s\le t\right\}. \tag{2.20}$$

Pontryagin's maximum principle (PMP) states that the optimal trajectory of state $\mathbf{y}(t)$ arriving at $\mathbf{y}(t)=\mathbf{x}$ and costate $\mathbf{p}(t)$ satisfies

$$\begin{cases}\dot{\mathbf{y}}=\mathbf{q},\ \ \mathbf{y}(t)=\mathbf{x}, & (2.21\text{a})\\ \dot{\mathbf{p}}=0,\ \ \mathbf{p}(0)=\nabla g(\mathbf{y}(0)), & (2.21\text{b})\\ \mathbf{q}=\operatorname*{argmax}_{\mathbf{v}}\left\{\mathbf{p}^{\mathrm{T}}\mathbf{v}-L(\mathbf{v})\right\}. & (2.21\text{c})\end{cases}$$

Note that this is identical to the characteristic ODEs for the state $\mathbf{x}$ (2.2a) and the gradient of the solution (2.2d) of the HJ PDEs. Therefore, PMP implies that the characteristic of the HJ PDE corresponds to the optimal trajectory.

We now establish that both the Hopf-Lax formula and the implicit solution formula can be derived from the PMP in conjunction with the definition of the value function. This facilitates a comprehensive understanding of their relationships and differences.

Hopf-Lax Formula. Since the costate $\mathbf{p}$ is constant along the optimal trajectory (2.21b), the last condition (2.21c) of the PMP implies that $\mathbf{q}$ is also constant. Therefore, from (2.21a), it follows that the optimal trajectory of $\mathbf{y}$ is a straight line, whose solution is given by

$$\mathbf{y}(0)=\mathbf{y}(t)-t\mathbf{q}=\mathbf{x}-t\mathbf{q}. \tag{2.22}$$

Therefore, the optimal $\mathbf{q}$ is expressed by $\mathbf{y}(0)$ as follows:

$$\mathbf{q}=\frac{\mathbf{x}-\mathbf{y}(0)}{t}, \tag{2.23}$$

and hence, the minimization with respect to $\mathbf{q}$ can be transformed into a minimization with respect to $\mathbf{y}(0)\in\mathbb{R}^d$. Substituting this relation (2.23) into the definition of the value function (2.20) leads to the following Hopf-Lax formula:

$$\begin{aligned}u(\mathbf{x},t)&=\inf_{\mathbf{y}\in\mathbb{R}^d}\left\{\int_0^t L\left(\frac{\mathbf{x}-\mathbf{y}}{t}\right)\mathrm{d}s+g(\mathbf{y})\right\}\\&=\inf_{\mathbf{y}\in\mathbb{R}^d}\left\{tL\left(\frac{\mathbf{x}-\mathbf{y}}{t}\right)+g(\mathbf{y})\right\}\\&=\inf_{\mathbf{y}\in\mathbb{R}^d}\left\{tH^*\left(\frac{\mathbf{x}-\mathbf{y}}{t}\right)+g(\mathbf{y})\right\}.\end{aligned}$$

In other words, the Hopf-Lax formula is derived by substituting the control $\mathbf{q}$ in terms of the initial state $\mathbf{y}(0)=\mathbf{y}$, leveraging the fact that the optimal trajectory is linear (2.23), as determined by the characteristic ODE of the PMP.

Implicit Solution formula. The implicit solution formula (2.3) is derived in a manner analogous to the Hopf-Lax formula, but it differs by expressing $\mathbf{p}$ as the gradient of the value function ∇u and additionally removing the Legendre transform. From the optimality of $\mathbf{q}$ in (2.21c), the Hamiltonian is written as

$$H(\mathbf{p})=\mathbf{p}^{\mathrm{T}}\mathbf{q}+L(\mathbf{q}). \tag{2.24}$$

It follows that

$$\nabla_{\mathbf{p}} H = \frac{\partial H}{\partial \mathbf{p}} + \frac{\partial H}{\partial \mathbf{q}} \frac{\partial \mathbf{q}}{\partial \mathbf{p}} = \frac{\partial H}{\partial \mathbf{p}} = \mathbf{q}, \tag{2.25}$$

where $\frac{\partial H}{\partial \mathbf{q}} = 0$ is induced from (2.21c). Let $\mathbf{q}^*$ be the optimal control. Putting (2.21c), (2.22), and (2.23) to the definition of the value function in (2.20) leads to the following formula of the value function:

$$\begin{aligned} u(\mathbf{x}, t) &= tL(\mathbf{q}^*) + g(\mathbf{x} - t\mathbf{q}^*) \\ &= t\left(H(\mathbf{p}) - \mathbf{p}^{\mathrm{T}}\mathbf{q}^*\right) + g(\mathbf{x} - t\mathbf{q}^*) \\ &= t\left(H(\mathbf{p}) - \mathbf{p}^{\mathrm{T}}\nabla_{\mathbf{p}}H(\mathbf{p})\right) + g(\mathbf{x} - t\nabla_{\mathbf{p}}H(\mathbf{p})), \end{aligned} \tag{2.26}$$

where the second and third equalities are derived from (2.24) and (2.25), respectively. In other words, by substituting these two expressions (2.24) and (2.25), we derive the formula for the value function u that is independent of both the control $\mathbf{q}$ and the Legendre transform. Since the optimal $\mathbf{p}$ is the gradient of the value function ∇u, the solution formula (2.26) derived from PMP is identical to the implicit solution formula (2.3). Furthermore, it is important to note that the definition of the value function (2.20) precisely encapsulates the integral of the characteristic ODE of u (2.2b); that is, it directly represents the solution to the characteristic ODE of u (2.2b). In other words, the characteristic ODE of u (2.2b), which is not explicitly included in the PMP formula (2.21), is inherently embedded within the construction of Bellman's value function.

Consequently, the PMP (2.21a)-(2.21c), the Bellman's value function (2.20), the Hopf-Lax formula (2.4), and the proposed implicit solution formula (2.3) are all interconnected. The PMP states that the characteristics of the HJ PDE correspond to the optimal trajectory, and Bellman's principle expresses the value function in terms of the solution to the characteristic ODE of u (2.2b). Together, they imply that the viscosity solution to the HJ PDE (2.1) is defined along the characteristics.

However, there are notable differences in how these formulas yield the solution to (2.1) from a practical perspective. The PMP necessitates the solution of a single trajectory of the characteristic ODEs, which implies that, when attempting to compute the value function, one must solve a system of ODEs for each trajectory, introducing significant computational complexity. The Hopf-Lax formula, by exploiting the linearity of the optimal trajectory, eliminates the need to solve such ODEs. However, it involves the computation of the Legendre transform of the Hamiltonian H, ultimately leading to a challenging min-max problem. In contrast, the implicit solution formula (2.3) alleviates both the ODE solving of PMP and the min-max problem from the Hopf-Lax formula by leveraging the fact that the optimal costate $\mathbf{p}$ is the gradient of the solution ∇u. Consequently, compared to the PMP and the Hopf-Lax formula, the proposed implicit formula provides a more practical and widely applicable approach for solving HJ PDEs.

3. Learning Implicit Solution with Neural Networks.

In this section, we introduce a deep learning-based approach for solving the implicit solution formula (2.3). Building upon the implicit solution formula, we propose the following minimization problem:

$$\min_u \mathcal{L}(u) := \int_0^T \int_\Omega \Big(u + tH(\nabla u) - t\nabla u^{\mathrm{T}} \nabla H(\nabla u) - g(\mathbf{x} - t\nabla H(\nabla u)) \Big)^2 \,\mathrm{d}\mathbf{x}\,\mathrm{d}t. \tag{3.1}$$

The minimization problem (3.1) is inherently complex to be efficiently solved using classical numerical methods. To address this challenge, we propose a deep learning framework that has shown significant effectiveness in optimizing complex problems. This approach enables the scalable learning of the implicit solution formula, even in high-dimensional settings, thereby allowing the solution of the HJ PDEs (2.1) to be represented by a neural network, which is a Lipschitz function.

3.1. Implicit Neural Representation. We represent the solution u of the HJ PDE (2.1) using a standard artificial neural network architecture, a multi-layer perceptron (MLP). The MLP is a function $u_\theta : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}$ defined as the composition of functions, which can be expressed as follows:

$$u_\theta(\mathbf{x}, t) = W\left(h_L \circ \cdots \circ h_0(\mathbf{x}, t)\right) + \mathbf{b}, \ (\mathbf{x}, t) \in \mathbb{R}^n \times \mathbb{R}, \tag{3.2}$$

where $L \in \mathbb{N}$ is a given depth, $W \in \mathbb{R}^{1 \times d_L}$ is a weight of the output layer, $\mathbf{b} \in \mathbb{R}$ is an output bias and the perceptron (also known as the hidden layer) $h_\ell : \mathbb{R}^{d_{\ell-1}} \to \mathbb{R}^{d_\ell}$ is defined by

$$h_\ell(\mathbf{y}) = \sigma\left(W_\ell \mathbf{y} + \mathbf{b}_\ell\right), \ \mathbf{y} \in \mathbb{R}^{d_{\ell-1}}, \text{ for all } \ell = 0, \dots, L,$$

for weight matrices $W_\ell \in \mathbb{R}^{d_\ell \times d_{\ell-1}}$ with the input dimension $d_{-1} = d+1$, bias vectors $\mathbf{b}_\ell \in \mathbb{R}^{d_\ell}$, and a non-linear activation function σ. The dimensions d_ℓ of the hidden layers are also called by the width of the network. A shorthand notation θ is used to refer to all the parameters of the network, including the weights $\{W, W_0, \cdots, W_L\}$ and biases $\{\mathbf{b}, \mathbf{b}_0, \cdots, \mathbf{b}_L\}$. Since Lipschitz continuous activation functions σ are used, the MLP f_θ is a Lipschitz function and is also bounded on a bounded domain. Given the current parameter configuration, the parameters θ are successively adapted by minimizing an assigned loss function explained in the subsequent section.

Representing the solution of HJ PDEs using neural networks offers a scalable and efficient approach for modeling the spatio-temporal dependencies of the solution, offering several advantages over classical numerical schemes. Classical methods discretize the spatial vector field using primitives such as meshes, which scale poorly with the number of spatial samples. In contrast, representing the spatio-temporal function through networks known as implicit neural representations (INRs) encodes spatial and temporal dependencies through neural network parameters θ, each globally influencing the function. Consequently, the memory usage of INRs remains independent of the spatial sample size, being determined solely by the number of network parameters, which enables scalability in high-dimensional settings as evidenced in Table 1 for the proposed method. Additionally, INRs are adaptive, leveraging their capacity to represent arbitrary spatio-temporal locations of interest without requiring memory expansion or structural modifications. The expressivity of non-linear neural networks enables INRs to achieve superior accuracy compared to mesh-based and meshless methods, even under the same memory constraints [69, 9]. Furthermore, INRs represent the solution as a continuous function rather than at discrete points, with activation functions that can be tailored to the solution's regularity. Thanks to the architecture of MLPs, exact derivatives can be computed via the chain rule, eliminating the need for numerical differentiation methods such as finite differences. The partial derivatives of u_θ are efficiently computed using automatic differentiation library (`autograd`) [62].

3.2. Training. Given that the solution u is represented by the neural network u_θ, the minimization problem (3.1) reduces to finding the network parameters θ that minimize $\mathcal{L}$ in (3.1). For notational convenience, we denote

$$\mathcal{S}\left(u\right)=u+tH\left(\nabla u\right)-t\nabla u^{\mathrm{T}}\nabla H\left(\nabla u\right)-g\left(\mathbf{x}-t\nabla H\left(\nabla u\right)\right).$$

The integral of $\mathcal{L}$ is approximated using Monte Carlo methods

$$\hat{\mathcal{L}}\left(\theta\right)=\frac{1}{M}\sum_{j=1}^{M}\mathcal{S}\left(u_\theta\left(\mathbf{x}_j,t_j\right)\right)^2, \tag{3.3}$$

with the M collocation points $\{(\mathbf{x}_j,t_j)\}_{j=1}^M$ randomly sampled from a uniform distribution $U\left(\Omega\times[0,T]\right)$. This empirical loss $\hat{\mathcal{L}}$ serves as the loss function for training the neural network. The current network parameters are updated using a gradient-based optimizer to minimize the loss function $\hat{\mathcal{L}}$. During training, different random collocation points are employed in each iteration to ensure accurate learning across the entire domain. The partial derivatives of the network u_θ are computed through `autograd` when calculating the loss. The training procedure for optimization using gradient descent is summarized in Algorithm 3.1.

Algorithm 3.1 Algorithm for Learning Implicit Solution of HJ PDEs

1: Initialize the network u_θ with an initial network parameter θ_0.
2: **for** $n=0,\cdots,N$ **do**
3: Randomly sample M collocations points $\{(\mathbf{x}_j,t_j)\}_{j=1}^M\sim U\left(\Omega\times[0,T]\right)$.
4: Calculate the loss by Monte Carlo integration

$$\hat{\mathcal{L}}\left(\theta_n\right)=\frac{1}{M}\sum_{j=1}^{M}\mathcal{S}\left(u_{\theta_n}\left(\mathbf{x}_j,t_j\right)\right)^2.$$

5: Update θ_n by gradient descent with a step size $\alpha>0$

$$\theta_{n+1}\leftarrow\theta_n-\alpha\nabla_\theta\hat{\mathcal{L}}\left(\theta_n\right).$$

6: **end for**
7: **return** u_{θ_N} as the predicted viscosity solution to the HJ PDE (2.1).

This algorithm is considerably simpler than existing methodologies in several respects and yields remarkable results, as demonstrated in section 4. Previous approaches [20, 12, 13, 10], which aimed to obtain the viscosity solution via the Hopf or Lax formula, involved calculating the Legendre transform of the Hamiltonian or the initial function. Therefore, these methods were restricted to problems where the Legendre transform was easily computable or required solving numerically intensive min-max problems for each spatio-temporal point, limiting their general applicability. In contrast, our approach bypasses the Legendre transform by using an implicit formula, enabling us to handle a broader class of Hamiltonians and initial functions. Moreover, while prior methods based on characteristics or PMP [38, 39, 73] necessitated solving a system of ODEs to track individual trajectories, the proposed method eliminates the requirement for explicit trajectory computation.

The proposed method also overcomes the limitations of classical grid-based numerical methods, which face challenges when dealing with high-dimensional or large-scale problems due to the increasing number of grid points required as the dimension or domain size grows. Unlike classical methods, the deep learning approach is characterized by its mesh-free nature, which precludes the necessity for a grid discretization of the computational domain. The mesh-free approach allows for the random selection of collocation points in each iteration, with the selected points gradually covering the domain as iterations proceed. Consequently, the computational and memory requirements do not increase significantly with higher dimensions, as evidenced in Table 1 for the proposed method. Furthermore, under certain mild assumptions, it has been demonstrated that this stochastic gradient descent applied using randomly sampled collocation points converges to the minimizer of the original expectation loss [40]. The absence of mesh generation also simplifies the practical implementation of the method.

This approach also offers distinct advantages over existing deep learning methods for solving PDEs. As an unsupervised learning method, it solves HJ PDEs given the Hamiltonian and initial condition, without requiring solution data for training. This addresses the limitations of supervised learning methods [55, 25, 16], which rely on extensive solution data and do not guarantee generalization to unseen problems. The proposed approach also offers strengths compared to the established unsupervised methods, such as PINNs [64] and the DeepRitz method [74]. DeepRitz, which is based on a variational formulation, is not suitable for HJ PDEs. PINNs, on the other hand, use the residual of the PDE itself as the loss function, which cannot guarantee obtaining the viscosity solution for HJ PDEs among multiple solutions. Since the viscosity solution cannot be directly derived from the PDE itself, there are inherent challenges in obtaining it from the PDE residual loss used in PINNs. In comparison, the proposed method learns an implicit formula for the solution that naturally inherits the properties of the viscosity solution through the characteristic equation, enabling effective solutions to HJ PDEs. Furthermore, most deep learning methods for solving PDEs, including PINNs and DeepRitz method, use a training loss function that is the linear sum of the loss term corresponding to the PDE and the loss term for the initial condition. This requires a regularization parameter to balance the two loss terms, which is highly sensitive and difficult to optimize [70]. In contrast, the proposed method employs an implicit solution formula, whereby the initial condition is automatically incorporated by substituting $t = 0$ into (3.3). As a result, our approach eliminates the need for a regularization parameter, using only a single loss function and obviating the distinction between the initial condition and the PDE.

When boundary conditions are specified in the spatial domain Ω, we incorporate additional loss terms to enforce these conditions. For instance, when a Dirichlet boundary condition is imposed with the boundary function $h : \partial\Omega \to [0, T] \to \mathbb{R}$, the following loss function is used:

$$\frac{1}{M_b} \sum_{j=1}^{M_b} \left(u\left(\mathbf{x}_j^b, t_j^b\right) - h\left(\mathbf{x}_j^b, t_j^b\right)\right)^2 ,$$

where the M_b boundary collocation points $\left(\mathbf{x}_j^b, t_j^b\right) \in \partial\Omega \times [0, T]$ are randomly sampled from a uniform distribution. Similarly, for a periodic boundary condition, the boundary loss term is given as follows:

$$\frac{1}{M_b} \sum_{j=1}^{M_b} \left(u\left(\mathbf{x}_j^b, t_j^b\right) - u\left(\mathbf{y}_j^b, t_j^b\right)\right)^2 ,$$

where $\mathbf{y}_i^b \in \partial\Omega$ represents the point on the opposite side of the domain Ω corresponding to $\mathbf{x}_i^b$. This boundary loss, weighted by the regularization parameter $\lambda > 0$, is then added to the implicit solution loss $\hat{\mathcal{L}}$ (3.3) to form the total training loss.

Remark 3.1. If the goal is to obtain a solution at a specific time $t = T$ rather than over the entire temporal evolution, integrating over time in the loss function may not be necessary. However, as shown in Example 2.1 in subsection 2.1, when the PDE lacks boundary conditions, at a fixed time $t > 0$, the the implicit solution formula at a fixed t results in a differential equation without boundary conditions, leading to the possibility of multiple spurious solutions. To address this, it is preferable to incorporate an integral over the entire temporal domain in the loss function, thereby training a continuous network to find a continuous solution that satisfies (2.3) across the entire spacetime domain. On the other hand, when boundary conditions are given in the HJ PDE (2.1), these can serve as the boundary condition for the differential equation (2.3) at the fixed time, ensuring the uniqueness of the solution. In such cases, training the model exclusively with respect to the terminal time $t = T$ may suffice.

3.3. State-dependent Hamiltonian. In this subsection, we propose an algorithm for the case of a state-dependent Hamiltonian, inspired by the implicit solution formula (2.3). Consider the state-dependent HJ PDE defined in a domain $\Omega \subset \mathbb{R}^d$

$$\begin{cases} u_t + H\left(\mathbf{x}, \nabla u\right) = 0 & \text{in } \Omega \times (0, T) \\ u = g & \text{on } \Omega \times \{t = 0\}. \end{cases} \tag{3.4}$$

The system of characteristic ODEs of (3.6) is given by

$$\begin{cases} \dot{\mathbf{x}} = \nabla_{\mathbf{p}} H & (3.5a) \\ \dot{u} = -H + \mathbf{p}^{\mathrm{T}} \nabla_{\mathbf{p}} H & (3.5b) \\ \dot{\mathbf{p}} = -\nabla_{\mathbf{x}} H, & (3.5c) \end{cases}$$

where $\mathbf{p} = \nabla u$. Given that $\mathbf{p}$ is no longer a constant along the characteristic (3.5c), the characteristics are not linear but instead curves. Consequently, computing the integral along these curves becomes highly challenging, making the derivation of an implicit solution formulation difficult.

Piecewise Linear Approximation of Characteristic Curves. We assume that $\mathbf{p}$ remains relatively constant over short time intervals. In other words, we approximate the characteristic curve as linear over short time segments. To this end, we discretize the temporal domain by

$$t_0 = 0 < t_1 = \Delta t < t_2 = 2\Delta t < \cdots, t_N = N\Delta t = T.$$

For each $k = 0, \cdots, N-1$, we can write the solution as follows: for $t \in [t_k, t_k + \Delta t]$,

$$u\left(\mathbf{x}, t\right) = u\left(\mathbf{x}, t_k + \tau\right) = u^k\left(\mathbf{x}, \tau\right)$$

with $t = t_k + \tau$, $\tau \in [0, \Delta t]$. Then u^k can be regarded as the solution of the following HJ PDE for time $0 \leq t \leq \Delta t$ with the initial function $u^k(\cdot, 0) = u^{k-1}(\cdot, \Delta t) = u(\cdot, k\Delta t)$:

$$\begin{cases} u_t^k + H\left(\mathbf{x}, \nabla u^k\right) = 0 & \text{in } \Omega \times (0, \Delta t) \\ u^k\left(\mathbf{x}, 0\right) = u^{k-1}\left(\mathbf{x}, \Delta t\right) & \text{on } \Omega. \end{cases} \tag{3.6}$$

Assuming that $\mathbf{p}$ remains constant within each short time interval $[t_k, t_k + \Delta t]$, similar to the state-independent Hamiltonian discussed in subsection 2.1, we can derive the following *implicit solution formula* for (3.6):

$$u^k(\mathbf{x}, \tau) = -\tau H\left(\mathbf{x}, \nabla u^k(\mathbf{x}, \tau)\right) + \tau \nabla u^k(\mathbf{x}, \tau)^{\mathrm{T}} \nabla_{\mathbf{p}} H\left(\mathbf{x}, \nabla u^k(\mathbf{x}, \tau)\right) \tag{3.7}$$

$$+ u^{k-1}\left(\mathbf{x} - \tau \nabla_{\mathbf{p}} H\left(\mathbf{x}, \nabla u^k\right), \Delta t\right). \tag{3.8}$$

This can be regarded as an implicit Euler discretization of the characteristic ODE (3.5a):

$$\mathbf{x}(\tau) = \mathbf{x}(0) + \tau \nabla_{\mathbf{p}} H\left(\mathbf{x}(\tau), \nabla u(\mathbf{x}(\tau), \tau)\right) + O\left(\tau^2\right)$$

for small $\tau \in [0, \Delta t]$. For notational simplicity, let us denote

$$\begin{aligned}\mathcal{S}\left[u^k, u^{k-1}\right](\mathbf{x}, \tau) =& u^k(\mathbf{x}, \tau) - \tau \nabla u^k(\mathbf{x}, \tau)^{\mathrm{T}} \nabla_{\mathbf{p}} H\left(\mathbf{x}, \nabla u^k(\mathbf{x}, \tau)\right) \\ &+ \tau H\left(\mathbf{x}, \nabla u^k(\mathbf{x}, \tau)\right) - u^{k-1}\left(\mathbf{x} - \tau \nabla_{\mathbf{p}} H\left(\mathbf{x}, \nabla u^k\right), \Delta t\right).\end{aligned}$$

Time Marching Algorithm. Based on these, we propose the following time marching method that solves the HJ PDE (3.6) with the state dependent H sequentially over short time intervals $[t_k, t_k + \Delta t]$:

1. Set the initial condition $u^0 = g$.
2. For $k = 1, \cdots, N$,

$$u^k = \arg\min_{v} \int_0^{\Delta t} \int_{\Omega} \left(\mathcal{S}\left[v, u^{k-1}\right](\mathbf{x}, \tau)\right)^2 \mathrm{d}\mathbf{x}\, \mathrm{d}t. \tag{3.9}$$

For each k, the predicted function u^k approximates the solution u of (3.6) on $t_k \leq t \leq t_{k+1}$, i.e.,

$$u^k(\mathbf{x}, \tau) \approx u(\mathbf{x}, k\Delta t + \tau),\ \forall \tau \in [0, \Delta t], \mathbf{x} \in \Omega.$$

It is important to note that rather than using separate neural networks for each u^k, the model is trained using a single network, ensuring memory efficiency. After training the network for the solution u^{k-1} on the time interval $[t_{k-1}, t_{k-1} + \Delta t]$, the network parameters are saved. These saved parameters are then used as the initial function to train the same network for the subsequent solution u^k by (3.9). As a result, when training u^k, the network is initialized with u^{k-1}, which accelerates the training process. Consequently, although the learning process is divided for time marching, the rapid convergence for each u^k ensures that the overall training time does not increase significantly.

4. Numerical Results.

In this section, we evaluate the performance of the proposed deep learning-based method for learning the implicit solution formula through a series of diverse examples and high-dimensional problems. Experiments are conducted on up to 40 dimensions, and both qualitative and quantitative results are presented. Although theoretical verification has not yet been established, extensive experiments on nonconvex Hamiltonians are also included, demonstrating the effectiveness of the proposed method in learning viscosity solutions.

To assess the scalability of the proposed method, we maintain the same experimental configurations across all cases, regardless of dimensionality or domain size. All experiments are conducted using an MLP (3.2) of a depth $L = 5$ and a width

64 with the softplus activation function $\sigma(x) = \frac{1}{\beta}\log\left(1+e^{100x}\right)$. The network is trained for for $N = 200,000$ epochs using the gradient descent with an initial learning rate of 10^{-3} decayed by a factor of 0.99 whenever the loss decreased. In each epoch, $M = 5,000$ collocation points were uniformly randomly sampled from the domain. When boundary conditions are given, the regularization parameter λ is set to 0.1 and the number of boundary collocation points M_b is set to 200. All experiments are implemented on a single NVIDIA GV100 (TITAN V) GPU.

4.1. Convex Hamiltonians. We begin by measuring the error with respect to the true solution for the theoretically validated convex Hamiltonian. Experiments are conducted in up to 40 dimensions. In addition to accuracy, we also measure computational time and memory consumption to assess the efficiency of the approach for high-dimensional problems.

EXAMPLE 4.1 (Burgers' equation). *Consider the Burgers' equation with the quadratic Hamiltonian* $H(\mathbf{p}) = \frac{1}{2}\|\mathbf{p}\|_2^2$ *and initial function* $g(\mathbf{x}) = \|\mathbf{x}\|_1$*. Experiments were conducted on the 1, 10, and 40 dimensions. The Mean Squared Error (MSE) with respect to the exact solution is summarized in the first row of Table* 1*.*

To assess how the computational cost increases with dimension, we measure both computational time and memory consumption. The time taken for each training epoch was averaged over the total 10,000 training epochs, while the memory consumption is recorded as the maximum memory usage during a single epoch. The average values for these measurements across the three examples, including the two provided below, are reported in Table 1*. It can be observed that neither computational time nor memory consumption increases sharply with dimension. It is important to emphasize that the computational time reported in Table* 1 *refers to the time required to obtain the solution function over the entire spatio-temporal domain, rather than the time taken to compute the solution at discrete points or on a grid. Moreover, as discussed in subsubsection* 3.1*, the memory requirements of implicit neural representations are primarily determined by the size of the network. Increasing the dimension does not significantly alter the overall network size, except for the increase in the input dimension of the input layer. Consequently, the results in Table* 1 *demonstrate that memory usage is nearly independent of dimensionality. These findings demonstrate that deep learning methods are highly scalable with respect to dimensionality, making them well-suited for addressing high-dimensional problems.*

EXAMPLE 4.2 (Concave Hamiltonian). *Consider the quadratic Hamiltonian* $H(\mathbf{p}) = -\frac{1}{2}\|\mathbf{p}\|_2^2$ *and initial function* $g(\mathbf{x}) = \|\mathbf{x}\|_1$*. Experiments were conducted on the 1, 10, and 40 dimensions until time* $T = 1$*. MSE with respect to the exact solution is reported in the second row of Table* 1*.*

EXAMPLE 4.3 (Level set Propagation). *We consider the level set equation [57] that governs the collision of two spheres, initially separated and moving along their respective normal directions, which ultimately results in a collision. The level set propagation is written by*

$$u_t + \|\nabla u\|_2 = 0,$$

where the initial function g *is given as the signed distance function for two circles with centers at* $(-0.3, 0, \cdots, 0)$ *and* $(0.3, 0, \cdots, 0)$*, and radius of* 0.2*. Experiments were conducted on the 1, 10, and 40 dimensions with* $T = 1$*. MSE with respect to the exact solution is summarized in the bottom row of Table* 1*. Figure* 1 *depicts the obtained solution for the two-dimensional case.*

TABLE 1

The mean squared errors with the exact solution, the average computational time per epoch, and the memory usage for Examples 4.1, 4.2, and 4.3 across dimensions $d = 1, 10, 40$ are presented. The computational time is measured as the average across three examples over a total of 10,000 epochs. Maximum memory consumption per iteration is measured. It is observed that the computational time and memory usage do not increase significantly as the dimension increases.

Problem	$d = 1$	$d = 10$	$d = 40$
Example 4.1	1.14E-7	2.56E-5	1.30E-3
Example 4.2	8.59E-6	1.63E-4	1.23E-3
Example 4.3	7.08E-6	5.57E-5	1.13E-3
Time (s) per Epoch	0.01518	0.01630	0.01864
Memory (MB)	42.648	42.648	43.623

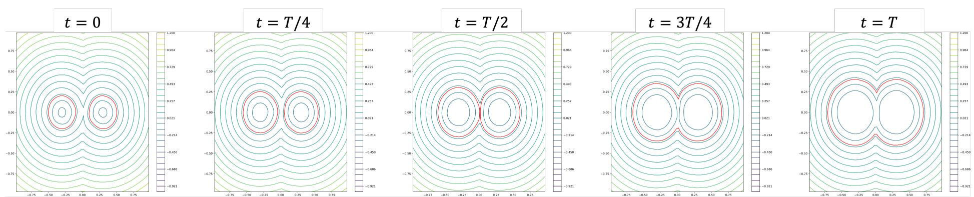


FIG. 1. *Iso-contours of the numerical solution to two-dimensional collision of circles in Example 4.3. The predicted zero leve lsets are represented by red curves.*

4.2. Nonconvex Hamiltonians. In this subsection, we present experimental results for various Hamiltonians that are neither convex nor concave. While the theoretical proof for the proposed implicit solution formula has not yet been established in the nonconvex case, the experiments show that the proposed method effectively yields viscosity solutions.

EXAMPLE 4.4. *We solve the nonlinear equation with a nonconvex Hamiltonian*

$$u_t - \cos\left(\sum_{i=1}^{d} u_{x_i} + 1\right) = 0,$$

with the initial function $g(\mathbf{x}) = -\cos\left(\frac{\pi}{d}\sum_{i=1}^{d} x_i\right)$, and periodic boundary conditions. The results for $d = 1, 2$ are depicted in Figure 2. The results are plotted up to time $T = 0.2$, at which point kinks have already emerged in the solution.

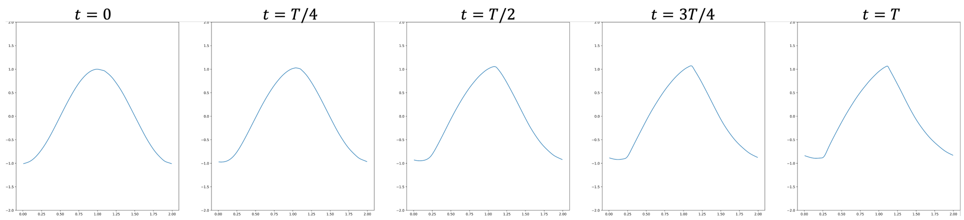
EXAMPLE 4.5. *The two-dimensional Riemann problem with a nonconvex flux*

$$\begin{cases} u_t + \sin(u_x + u_y) = 0, \\ u(x, y, 0) = \pi(|y| - |x|). \end{cases}$$

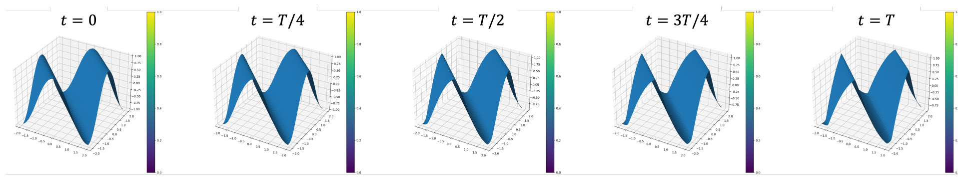
The predicted solution up to $T = 1$ is depicted in Figure 3.

EXAMPLE 4.6. *The above nonconvex examples are actually one-dimensional along the diagonal. To evaluate the performance of the proposed method on fully two-dimensional problems, we solve*

$$\begin{cases} u_t + u_x u_y = 0, \\ u(x, y, 0) = \sin(x) + \cos(y), \end{cases}$$



(a) $d = 1$



(b) $d = 2$

FIG. 2. *The numerical results for Example* 4.4.

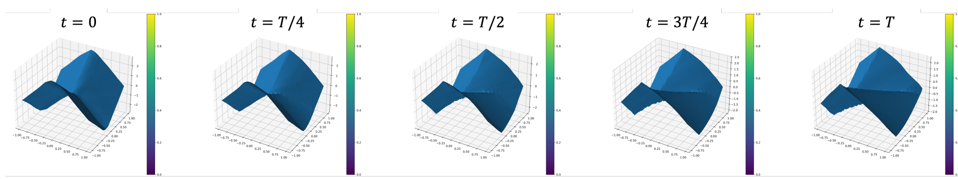


FIG. 3. *The numerical solution for Example* 4.5

with periodic boundary condition and $T = 1.125$. *The solution is smooth for* $t < 1$ *and has kinks for* $t \geq 1$. *Results are shown in Figure* 4.

EXAMPLE 4.7 (Eikonal equation). *Consider a two-dimensional nonconvex problem that arises in geometric optics:*

$$\begin{cases} u_t + \sqrt{u_x + u_y + 1} = 0, \\ u(x, y, 0) = \frac{1}{4}\left(\cos(2\pi x) - 1\right)\left(\cos(2\pi y) - 1\right) - 1. \end{cases}$$

The results up to time $T = 0.45$ *are shown in Figure* 5, *where we can observe the sharp corners in the solution.*

EXAMPLE 4.8. *Consider the combustion problem [44]:*

$$\begin{cases} u_t - \sqrt{u_x + u_y + 1} = 0, \\ u(x, y, 0) = \cos(2\pi x) - \cos(2\pi y). \end{cases}$$

Results up to time 0.27 *are given in Figure* 6.

EXAMPLE 4.9. *Consider the one-dimensional nonconvex problem*

$$\begin{cases} u_t + u_x^3 - u_x = 0, \\ u(x, 0) = -\frac{1}{10}\cos(5x). \end{cases}$$

The results up to time $T = 0.7$ *are shown in Figure* 7, *where we can observe can observe that the sinusoidal wave becomes progressively sharper.*

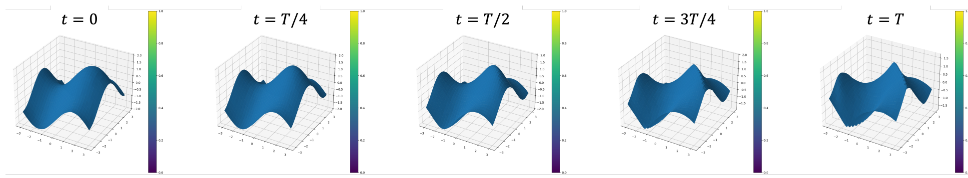


FIG. 4. *The numerical solution for Example* 4.6

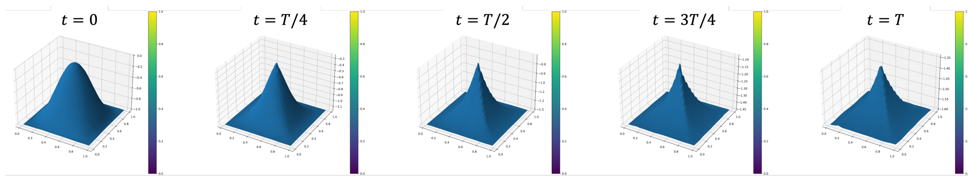


FIG. 5. *The numerical solution for Example* 4.7

4.3. State-dependent Hamiltonians. This subsection provides experimental validation of the methodology proposed for the state-dependent Hamiltonian in subsubsection 3.3. It includes error analyses with respect to Δt and addresses various state-dependent Hamiltonians, including a 10-dimensional optimal control problem.

EXAMPLE 4.10. *We solve the following variable coefficient linear equation [71]:*

$$\begin{cases} u_t + \sin(x)\, u_x = 0, \\ u(x, 0) = \sin(x), \end{cases}$$

with periodic boundary condition. The exact solution is expressed by

$$u(x,t) = \sin\left(2\arctan\left(e^{-t}\tan\left(\frac{x}{2}\right)\right)\right).$$

We compute the solution up to $T = 1$. Since the characteristic curve for the state-dependent Hamiltonian is approximated linearly, the accuracy of the algorithm is influenced by the size of Δt. To verify this, we conducted experiments for various values of $\Delta t = 0.5, 0.25, 0.1$, and the results are summarized in the top row of Table 2. The results show that the linear approximation of the proposed algorithm yields first-order accuracy with respect to Δt.

EXAMPLE 4.11. *We solve the following the two-dimensional linear equation which describes a solid body rotation around the origin [11]:*

$$u_t - y u_x + x u_y = 0, \ (x, y) \in (-1, 1)^2$$

where the initial condition is given by

$$g(x,y) = \begin{cases} 0 & 0.3 \le r, \\ 0.3 - r & 0.1 < r < 0.3 \\ 0.2 & r \le 0.1, \end{cases}$$

where $r = \sqrt{(x-0.4)^2 + (y-0.4)^2}$. We also impose the periodic boundary condition.

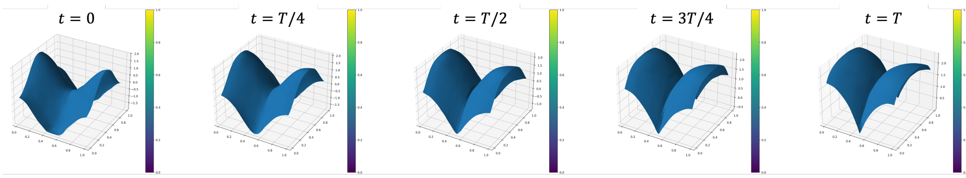


FIG. 6. *The numerical solution for Example* 4.8

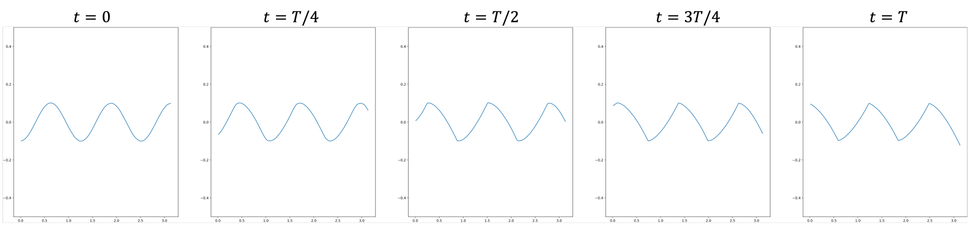


FIG. 7. *The numerical solution for Example* 4.9.

The exact solution is

$$u\left(x,y,t\right)=g\left(x\cos t+y\sin t,-x\sin t+y\cos t\right).$$

We compute the solution up to $T=1$ and the numerical errors for $\Delta t=0.5,0.25,0.1$ are reported in the second row of Table 2. *As in the previous example, we observe that the error increases linearly as Δt increases.*

EXAMPLE 4.12. *We solve an optimal control problem related to cost determination [60]:*

$$\begin{cases} u_t+u_x\sin y+\left(\sin y+sign\left(u_y\right)\right)u_y-\frac{1}{2}\sin^2 y-\left(1-\cos x\right)=0,\\ u\left(x,y,0\right)=0,\end{cases}$$

with periodic conditions. The result at $T=1$ is presented in Figure 8 *and is qualitatively in agreement with [60].*

EXAMPLE 4.13. *We solve the problem associated with the state-dependent Hamiltonian well-known as the harmonic oscillator:*

$$H^{\pm}\left(\mathbf{x},\mathbf{p}\right)=\pm\frac{1}{2}\left(\|\mathbf{x}\|_2^2+\|\mathbf{p}\|_2^2\right).$$

We consider the two-dimensional problem where the initial function is the level set function of an ellipsoid

$$g\left(x,y\right)=\frac{1}{2}\left(\frac{x^2}{2.5^2}+y^2-1\right). \tag{4.1}$$

The results up to $T=0.4$ are depicted in Figure 9.

EXAMPLE 4.14. *We consider a state-dependent nonconvex Hamiltonian of the following form given in [13]:*

$$H\left(\mathbf{x},p\right)=-c\left(\mathbf{x}\right)p_1+2\left|p_2\right|+\|\mathbf{p}\|_2-1,$$

TABLE 2
The mean squared errors (MSE) and relative mean square errors (RMSE) with the exact solution for Examples 4.10 and 4.11 with $\Delta t = 0.5, 0.25, 0.1$.

Problem	$\Delta t = 0.1$		$\Delta t = 0.25$		$\Delta t = 0.5$	
	MSE	RMSE	MSE	RMSE	MSE	RMSE
Example 4.10	3.27E-6	6.57E-6	8.82E-6	1.61E-5	1.26E-5	2.26E-5
Example 4.11	6.29E-7	1.25E-3	3.88E-6	2.10E-3	6.12E-6	4.32E-3

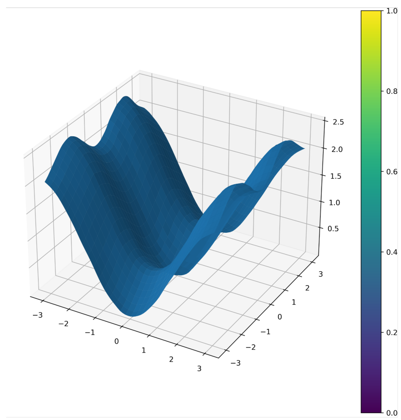


FIG. 8. *The numerical solution for Example* 4.12.

where $\mathbf{p} = (p_1, p_2)$ *and*

$$c(\mathbf{x}) = 2\left(1 + 3\exp\left(-4\left\|\mathbf{x} - (1,1)\right\|_2^2\right)\right). \tag{4.2}$$

We employ the initial function as presented in Example 4.13. *The results up to* $T = 1$ *are presented in Figure* 10, *which are consistent with those reported in [13].*

EXAMPLE 4.15. *We test the proposed method for a state-dependent nonconvex Hamiltonian of the following form given in [13]:*

$$H(\mathbf{x}, p) = -c(\mathbf{x})\,|p_1| - c(-\mathbf{x})\,|p_2|,$$

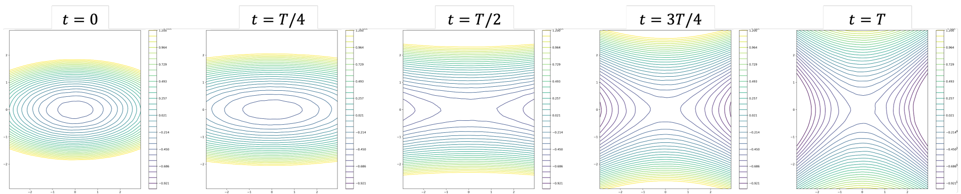
where we write $\mathbf{p} = (p_1, p_2)$ *and* $c(x)$ *is a coefficient function as given in* (4.2). *The initial function* g (4.1) *presented in Example* 4.13 *is employed in this instance. The results up to* $T = 0.3$ *are presented in Figure* 11 *and the results are in agreement with those reported in [13].*

EXAMPLE 4.16. *We solve the following optimal control problem:*

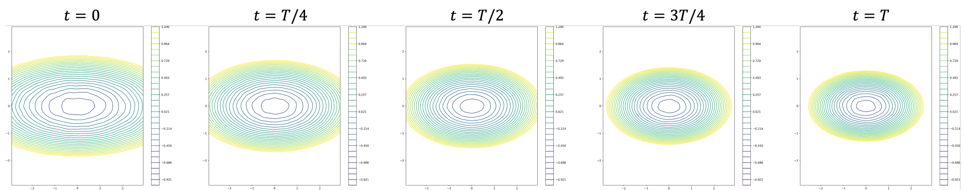
$$u(\mathbf{x}, t) = \inf\left\{g(\mathbf{x}(0))\,;\dot{\mathbf{x}}(t) = f(\mathbf{x}(t))\,\mathbf{a}(t),\ \mathbf{x}(t) = \mathbf{x},\ \|\mathbf{a}(t)\|_2 \leq 1\right\},$$

where g *is defined by*

$$g(\mathbf{x}) = \frac{1}{2}\left(\mathbf{x}^T A \mathbf{x} - 1\right)$$



(a) H^+



(b) H^-

FIG. 9. *The numerical results for Example* 4.13.

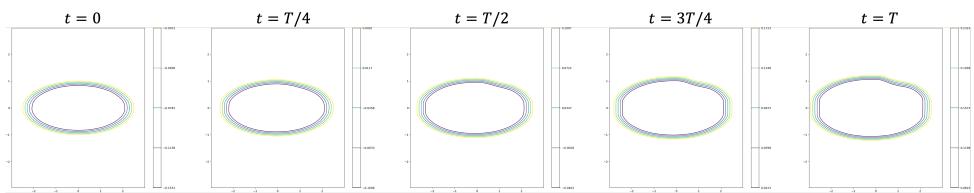


FIG. 10. *The numerical solution for Example* 4.14.

with $A = diag(0.25, 1)$ *and* f *is given by*

$$f(\mathbf{x}) = 1 + 3\exp\left(-4\left\|\mathbf{x} - (1,1)\right\|_2^2\right).$$

This corresponds to the HJ PDE given in [13]:

$$\begin{cases} u_t + f(\mathbf{x})\left\|\nabla u\right\|_2 = 0 \\ u(\mathbf{x}, 0) = g(\mathbf{x}). \end{cases}$$

When solving the maximization problem $\sup\ g(\mathbf{x}(T))$ *with the same constraints, we obtain the following HJ PDE:*

$$\begin{cases} u_t - f(\mathbf{x})\left\|\nabla u\right\|_2 = 0 \\ u(\mathbf{x}, 0) = g(\mathbf{x}). \end{cases}$$

The results for both the minimization (at $T = 0.2$*) and maximization (at* $T = 0.5$*) problems are presented in Figure* 12 *and are consistent with the results in [13].*

EXAMPLE 4.17. *Consider the following 10-dimensional quadratic optimal control problem presented in [10]:*

$$u(\mathbf{x}, t) = \inf\Big\{\int_0^t \left\|\dot{\mathbf{x}}(s)\right\|^2 - \psi(\mathbf{x}(s))\,\mathrm{d}s + g(\mathbf{x}(0))\,;\mathbf{x}(t) = \mathbf{x}\Big\},$$

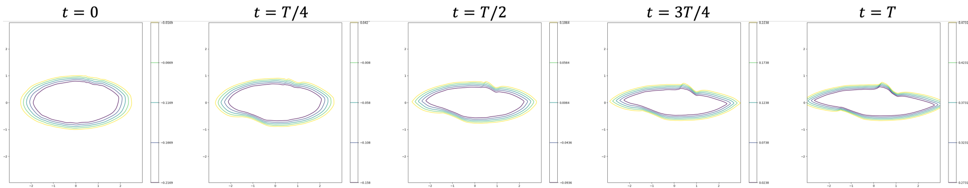
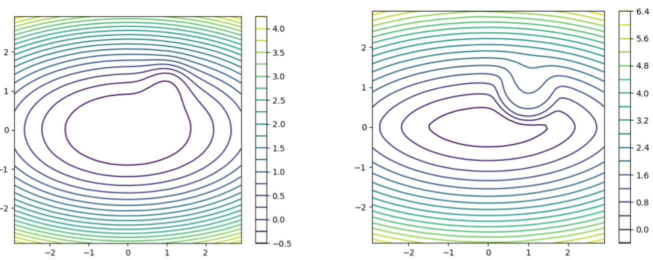


FIG. 11. *The numerical solution for Example* 4.15.



(a) Minimization (b) Maximization

FIG. 12. *The numerical solution for Example* 4.16 *with* $\Delta t = 0.1$.

where the potential function $\psi : \mathbb{R}^d \to (-\infty, 0]$ *is* $\psi(\mathbf{x}) = \sum_{i=1}^{d} \psi_i(\mathbf{x}_i)$, *where each function* $\psi_i : \mathbb{R} \to (-\infty, 0]$ *is a positively 1-homogeneous concave function given by*

$$\psi_i(x) = \begin{cases} -a_i x & x \geq 0, \\ b_i x & x < 0, \end{cases}$$

with parameters $(a_1, \cdots, a_d) = (4, 6, 5, \cdots, 5)$ *and* $(b_1, \cdots, b_d) = (3, 9, 6, \cdots, 6)$. *The corresponding HJ PDE reads:*

$$\begin{cases} u_t + \frac{1}{2} \|\nabla u\|^2 + \psi(\mathbf{x}) = 0 \\ u(\mathbf{x}, 0) = g(\mathbf{x}). \end{cases}$$

We conduct experiments for the two initial cost functions:

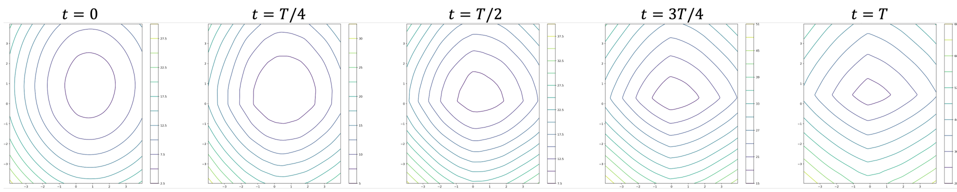
- *A quadratic initial function* $g_1(\mathbf{x}) = \frac{1}{2} \|\mathbf{x} - \mathbf{1}\|_2^2$, *where* $\mathbf{1}$ *denotes the d-dimensional vector whose elements are all one.*

- *A nonconvex initial function*

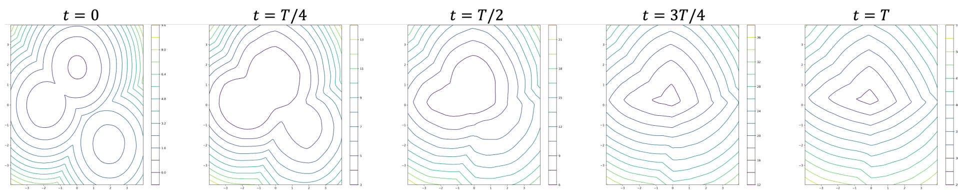
$$g_2(\mathbf{x}) = \min_{j \in \{1,2,3\}} g_j(\mathbf{x}) = \min_{j \in \{1,2,3\}} \frac{1}{2} \|\mathbf{x} - \mathbf{y}_j\|_2^2 - \alpha_j,$$

  where $\mathbf{y}_1 = (-2, 0, \cdots, 0)$, $\mathbf{y}_2 = (2, -2, -1, 0, \cdots, 0)$, $\mathbf{y}_3 = (0, 2, 0, \cdots, 0)$, $\alpha_1 = -0.5$, $\alpha_2 = 0$, *and* $\alpha_3 = -1$.

Figure 13 presents two-dimensional slices of the solutions in the xy plane for both cases up to time $T = 0.5$. The results demonstrate that the evolution of the solution is non-trivial, as evidenced by the nonlinear progression of the level sets over time, exhibiting multiple kinks. These findings are consistent with the experimental results presented in [10].



(a) g_1



(b) g_2

FIG. 13. *The numerical results for Example* 4.17.

5. Conclusion. We have introduced a novel implicit solution method for HJ PDEs derived from the characteristics of the PDE. This formula aligns with the Hopf-Lax formula for convex Hamiltonians but simplifies it by removing the need for Legendre transforms, thereby enhancing computational efficiency and broadening its practical applicability. The proposed formula not only bridges the method of characteristics, the Hopf-Lax formula, and Bellman's principle from control theory but also offers a simple and effective numerical approach for solving HJ PDEs. By integrating deep learning, the formula provides a scalable method that effectively mitigates the curse of dimensionality. Experimental results demonstrate its robustness and effectiveness across various high-dimensional and nonconvex problems without tuning the configuration of the deep learning model. These findings validate the method as a versatile and computationally efficient tool for solving high-dimensional, nonconvex dynamic systems and optimal control problems governed by HJ PDEs.

An important direction for future work includes a rigorous analysis of the proposed implicit solution formula. While experimental results demonstrate the method's effectiveness on various nonconvex problems, a comprehensive analysis is needed to confirm whether the proposed formula describes the viscosity solution of HJ PDEs in nonconvex problems. Since the formula involves the first derivatives and is a composite of multiple terms, the proposed minimization problem (3.1) is nonconvex, making the convergence of gradient descent non-trivial. Consequently, a convergence analysis would be an important future endeavor.

Regarding the deep learning approach, we approximate the expectation loss (3.1) using Monte Carlo integration (3.3), which introduces a discrepancy between the empirical and expectation losses. A valuable research direction could involve investigating whether the stochastic gradient descent process, with its random collocation points at each epoch, converges to the global minimum of the expectation loss in the context of stochastic approximation. Although we focused on scalability by maintaining a fixed model configuration across experiments, future research should explore the optimal selection of collocation points and network size for different problem dimensions. Furthermore, the investigation of using automatic differentiation to compute exact derivatives of the network, rather than finite differences such as ENO/WENO, presents an intriguing avenue for future research, particularly in its ability to capture shocks. For state-dependent Hamiltonians, the development of higher-order methods beyond the proposed first-order linear approximation of the characteristic curve would be a promising direction. Finally, the simplicity and efficiency of the proposed method open up avenues for its application to a wide range of problems, including level set evolutions, optimal transport, mean field games, and inverse problems, which would constitute valuable extensions of this work.

Acknowledgments. Tingwei

REFERENCES

[1] M. Akian, S. Gaubert, and A. Lakhoua, *The max-plus finite element method for solving deterministic optimal control problems: basic properties and convergence analysis*, SIAM Journal on Control and Optimization, 47 (2008), pp. 817–848.

[2] M. Ansari, A. Khajepour, and E. Esmailzadeh, *Application of level set method to optimal vibration control of plate structures*, Journal of Sound and Vibration, 332 (2013), pp. 687–700.

[3] S. Bansal, A. Bajcsy, E. Ratner, A. D. Dragan, and C. J. Tomlin, *A hamilton-jacobi reachability-based framework for predicting and analyzing human motion for safe planning*, in 2020 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2020, pp. 7149–7155.

[4] S. Bansal, M. Chen, S. Herbert, and C. J. Tomlin, *Hamilton-jacobi reachability: A brief overview and recent advances*, in 2017 IEEE 56th Annual Conference on Decision and Control (CDC), IEEE, 2017, pp. 2242–2253.

[5] M. Bardi and L. C. Evans, *On hopf's formulas for solutions of hamilton-jacobi equations*, Nonlinear Analysis: Theory, Methods & Applications, 8 (1984), pp. 1373–1381.

[6] S. Bryson and D. Levy, *High-order central weno schemes for multidimensional hamilton-jacobi equations*, SIAM Journal on Numerical Analysis, 41 (2003), pp. 1339–1369.

[7] Y. Cao and N. Wan, *Optimal proportional reinsurance and investment based on hamilton–jacobi–bellman equation*, Insurance: Mathematics and Economics, 45 (2009), pp. 157–162.

[8] V. Caselles, R. Kimmel, and G. Sapiro, *Geodesic active contours*, International journal of computer vision, 22 (1997), pp. 61–79.

[9] H. Chen, R. Wu, E. Grinspun, C. Zheng, and P. Y. Chen, *Implicit neural spatial representations for time-dependent pdes*, in International Conference on Machine Learning, PMLR, 2023, pp. 5162–5177.

[10] P. Chen, J. Darbon, and T. Meng, *Hopf-type representation formulas and efficient algorithms for certain high-dimensional optimal control problems*, Computers & Mathematics with Applications, 161 (2024), pp. 90–120.

[11] Y. Cheng and C.-W. Shu, *A discontinuous galerkin finite element method for directly solving the hamilton–jacobi equations*, Journal of Computational Physics, 223 (2007), pp. 398–415.

[12] Y. T. Chow, J. Darbon, S. Osher, and W. Yin, *Algorithm for overcoming the curse of dimensionality for time-dependent non-convex hamilton–jacobi equations arising from optimal control and differential games problems*, Journal of Scientific Computing, 73 (2017), pp. 617–643.

[13] Y. T. Chow, J. Darbon, S. Osher, and W. Yin, *Algorithm for overcoming the curse of dimensionality for state-dependent hamilton-jacobi equations*, Journal of Computational Physics, 387 (2019), pp. 376–409.

[14] M. G. Crandall and P.-L. Lions, *Viscosity solutions of hamilton-jacobi equations*, Transactions of the American mathematical society, 277 (1983), pp. 1–42.

[15] E. Cristiani and M. Falcone, *Fast semi-lagrangian schemes for the eikonal equation and applications*, SIAM Journal on Numerical Analysis, 45 (2007), pp. 1979–2011.

[16] J. Cui, S. Liu, and H. Zhou, *A supervised learning scheme for computing hamilton-jacobi equation via density coupling*, arXiv preprint arXiv:2401.15954, (2024).

[17] J. Darbon, P. M. Dower, and T. Meng, *Neural network architectures using min-plus algebra for solving certain high-dimensional optimal control problems and hamilton–jacobi pdes*, Mathematics of Control, Signals, and Systems, 35 (2023), pp. 1–44.

[18] J. Darbon, G. P. Langlois, and T. Meng, *Overcoming the curse of dimensionality for some hamilton–jacobi partial differential equations via neural network architectures*, Research in the Mathematical Sciences, 7 (2020), p. 20.

[19] J. Darbon and T. Meng, *On some neural network architectures that can represent viscosity solutions of certain high dimensional hamilton–jacobi partial differential equations*, Journal of Computational Physics, 425 (2021), p. 109907.
[20] J. Darbon and S. Osher, *Algorithms for overcoming the curse of dimensionality for certain hamilton–jacobi equations arising in control theory and elsewhere*, Research in the Mathematical Sciences, 3 (2016), p. 19.
[21] M. de León, J. C. Marrero, and D. M. de Diego, *Linear almost poisson structures and hamilton-jacobi equation. applications to nonholonomic mechanics*, arXiv preprint arXiv:0801.4358, (2008).
[22] D. Delahaye, S. Puechmorel, P. Tsiotras, and E. Féron, *Mathematical models for aircraft trajectory design: A survey*, in Air Traffic Management and Systems: Selected Papers of the 3rd ENRI International Workshop on ATM/CNS (EIWAC2013), Springer, 2014, pp. 205–247.
[23] H. H. Denman and L. H. Buch, *Solution of the hamilton-jacobi equation for certain dissipative classical mechanical systems*, Journal of Mathematical Physics, 14 (1973), pp. 326–329.
[24] S. Dolgov, D. Kalise, and K. K. Kunisch, *Tensor decomposition methods for high-dimensional hamilton–jacobi–bellman equations*, SIAM Journal on Scientific Computing, 43 (2021), pp. A1625–A1650.
[25] S. Dolgov, D. Kalise, and L. Saluzzi, *Data-driven tensor train gradient cross approximation for hamilton–jacobi–bellman equations*, SIAM Journal on Scientific Computing, 45 (2023), pp. A2153–A2184.
[26] L. C. Evans, *Partial differential equations*, vol. 19, American Mathematical Society, 2022.
[27] L. C. Evans and P. E. Souganidis, *Differential games and representation formulas for solutions of hamilton-jacobi-isaacs equations*, Indiana University mathematics journal, 33 (1984), pp. 773–797.
[28] M. Falcone and R. Ferretti, *Semi-lagrangian schemes for hamilton–jacobi equations, discrete representation formulae and godunov methods*, Journal of computational physics, 175 (2002), pp. 559–575.
[29] M. Falcone and R. Ferretti, *Semi-Lagrangian approximation schemes for linear and Hamilton—Jacobi equations*, SIAM, 2013.
[30] W. H. Fleming and W. M. McEneaney, *A max-plus-based algorithm for a hamilton–jacobi–bellman equation of nonlinear filtering*, SIAM Journal on Control and Optimization, 38 (2000), pp. 683–710.
[31] P. A. Forsyth and G. Labahn, *Numerical methods for controlled hamilton-jacobi-bellman pdes in finance*, Journal of Computational Finance, 11 (2007), p. 1.
[32] G. Gilboa and S. Osher, *Nonlocal operators with applications to image processing*, Multiscale Modeling & Simulation, 7 (2009), pp. 1005–1028.
[33] E. Hopf, *Generalized solutions of non-linear equations of first order*, Journal of Mathematics and Mechanics, 14 (1965), pp. 951–973.
[34] C. Imbert, R. Monneau, and H. Zidani, *A hamilton-jacobi approach to junction problems and application to traffic flows*, ESAIM: Control, Optimisation and Calculus of Variations, 19 (2013), pp. 129–166.
[35] G.-S. Jiang and D. Peng, *Weighted eno schemes for hamilton–jacobi equations*, SIAM Journal on Scientific computing, 21 (2000), pp. 2126–2143.
[36] D. Kalise, S. Kundu, and K. Kunisch, *Robust feedback control of nonlinear pdes by numerical approximation of high-dimensional hamilton–jacobi–isaacs equations*, SIAM Journal on Applied Dynamical Systems, 19 (2020), pp. 1496–1524.
[37] D. Kalise and K. Kunisch, *Polynomial approximation of high-dimensional hamilton–jacobi–bellman equations and applications to feedback control of semilinear parabolic pdes*, SIAM Journal on Scientific Computing, 40 (2018), pp. A629–A652.
[38] W. Kang and L. Wilcox, *A causality free computational method for hjb equations with application to rigid body satellites*, in AIAA Guidance, Navigation, and Control Conference, 2015, p. 2009.
[39] W. Kang and L. C. Wilcox, *Mitigating the curse of dimensionality: sparse grid characteristics method for optimal feedback control and hjb equations*, Computational Optimization and Applications, 68 (2017), pp. 289–315.
[40] R. L. Karandikar and M. Vidyasagar, *Convergence rates for stochastic approximation: Biased noise with unbounded variance, and applications*, Journal of Optimization Theory and Applications, (2024), pp. 1–39.
[41] K. Khanin and A. Sobolevski, *Particle dynamics inside shocks in hamilton–jacobi equations*, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 368 (2010), pp. 1579–1593.

[42] S. N. Kruzhkov, *The cauchy problem in the large for certain non-linear first order differential equations*, in Doklady Akademii Nauk, vol. 132, Russian Academy of Sciences, 1960, pp. 36–39.
[43] S. N. Kruzhkov, *Generalized solutions of nonlinear first order equations with several independent variables. ii*, Matematicheskii Sbornik, 114 (1967), pp. 108–134.
[44] F. Lafon and S. Osher, *High order two dimensional nonoscillatory methods for solving hamilton–jacobi scalar equations*, Journal of Computational Physics, 123 (1996), pp. 235–253.
[45] J. A. Laval and L. Leclercq, *The hamilton–jacobi partial differential equation and the three representations of traffic flow*, Transportation Research Part B: Methodological, 52 (2013), pp. 17–30.
[46] F. L. Lewis, D. M. Dawson, and C. T. Abdallah, *Robot manipulator control: theory and practice*, CRC Press, 2003.
[47] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, *Fourier neural operator for parametric partial differential equations*, arXiv preprint arXiv:2010.08895, (2020).
[48] F. Lin and R. D. Brandt, *An optimal control approach to robust control of robot manipulators*, IEEE Transactions on robotics and automation, 14 (1998), pp. 69–77.
[49] Z. Lin, J. Duan, S. E. Li, H. Ma, J. Li, J. Chen, B. Cheng, and J. Ma, *Policy-iteration-based finite-horizon approximate dynamic programming for continuous-time nonlinear optimal control*, IEEE Transactions on Neural Networks and Learning Systems, 34 (2022), pp. 5255–5267.
[50] P.-L. Lions, *Generalized solutions of hamilton-jacobi equations*, (No Title), (1982).
[51] L. Lu, P. Jin, and G. E. Karniadakis, *Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators*, arXiv preprint arXiv:1910.03193, (2019).
[52] W. M. McEneaney, *Max-plus methods for nonlinear control and estimation*, vol. 2, Springer Science & Business Media, 2006.
[53] J. C. Miñano, P. Benítez, and A. Santamaría, *Hamilton-jacobi equation in momentum space*, Optics Express, 14 (2006), pp. 9083–9092.
[54] I. Mitchell, A. Bayen, and C. J. Tomlin, *Computing reachable sets for continuous dynamic games using level set methods*, Submitted January, (2004).
[55] T. Nakamura-Zimmerer, Q. Gong, and W. Kang, *Adaptive deep learning for high-dimensional hamilton–jacobi–bellman equations*, SIAM Journal on Scientific Computing, 43 (2021), pp. A1221–A1247.
[56] S. Osher, *A level set formulation for the solution of the dirichlet problem for hamilton–jacobi equations*, SIAM Journal on Mathematical Analysis, 24 (1993), pp. 1145–1152.
[57] S. Osher, R. Fedkiw, and K. Piechor, *Level set methods and dynamic implicit surfaces*, Appl. Mech. Rev., 57 (2004), pp. B15–B15.
[58] S. Osher and N. Paragios, *Geometric level set methods in imaging, vision, and graphics*, Springer Science & Business Media, 2007.
[59] S. Osher and J. A. Sethian, *Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations*, Journal of computational physics, 79 (1988), pp. 12–49.
[60] S. Osher and C.-W. Shu, *High-order essentially nonoscillatory schemes for hamilton–jacobi equations*, SIAM Journal on numerical analysis, 28 (1991), pp. 907–922.
[61] C. Parzani and S. Puechmorel, *On a hamilton-jacobi-bellman approach for coordinated optimal aircraft trajectories planning*, Optimal Control Applications and Methods, 39 (2018), pp. 933–948.
[62] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, *Automatic differentiation in pytorch*, (2017).
[63] J. Qiu and C.-W. Shu, *Hermite weno schemes for hamilton–jacobi equations*, Journal of Computational Physics, 204 (2005), pp. 82–99.
[64] M. Raissi, P. Perdikaris, and G. E. Karniadakis, *Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations*, Journal of Computational physics, 378 (2019), pp. 686–707.
[65] S. Rajeev, *A hamilton–jacobi formalism for thermodynamics*, Annals of Physics, 323 (2008), pp. 2265–2285.
[66] A. Sideris and J. E. Bobrow, *An efficient sequential linear quadratic algorithm for solving nonlinear optimal control problems*, in Proceedings of the 2005, American Control Conference, 2005., IEEE, 2005, pp. 2275–2280.
[67] J. Sirignano and K. Spiliopoulos, *Dgm: A deep learning algorithm for solving partial dif-*

ferential equations, Journal of computational physics, 375 (2018), pp. 1339–1364.

[68] A. I. Subbotin, *Generalized solutions of first order PDEs: the dynamical optimization perspective*, Springer Science & Business Media, 2013.

[69] T. Takikawa, J. Litalien, K. Yin, K. Kreis, C. Loop, D. Nowrouzezahrai, A. Jacobson, M. McGuire, and S. Fidler, *Neural geometric level of detail: Real-time rendering with implicit 3d shapes*, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 11358–11367.

[70] S. Wang, X. Yu, and P. Perdikaris, *When and why pinns fail to train: A neural tangent kernel perspective*, Journal of Computational Physics, 449 (2022), p. 110768.

[71] J. Yan and S. Osher, *A local discontinuous galerkin method for directly solving hamilton–jacobi equations*, Journal of Computational Physics, 230 (2011), pp. 232–244.

[72] L. Yang, S. Liu, T. Meng, and S. J. Osher, *In-context operator learning with data prompts for differential equation problems*, Proceedings of the National Academy of Sciences, 120 (2023), p. e2310142120.

[73] I. Yegorov and P. M. Dower, *Perspectives on characteristics based curse-of-dimensionality-free numerical approaches for solving hamilton–jacobi equations*, Applied Mathematics & Optimization, 83 (2021), pp. 1–49.

[74] B. Yu et al., *The deep ritz method: a deep learning-based numerical algorithm for solving variational problems*, Communications in Mathematics and Statistics, 6 (2018), pp. 1–12.

[75] X. Zhang, T. Cheng, and L. Ju, *Implicit form neural network for learning scalar hyperbolic conservation laws*, in Mathematical and Scientific Machine Learning, PMLR, 2022, pp. 1082–1098.

[76] M. Zhou, J. Han, and J. Lu, *Actor-critic method for high dimensional static hamilton–jacobi–bellman partial differential equations based on neural networks*, SIAM Journal on Scientific Computing, 43 (2021), pp. A4043–A4066.