Is Bellman Equation Enough for Learning Control?

Haoxiang You, Lekan Molu, Ian Abraham

Keywords: Bellman equation, Value-based approach, Architecture for RL

Summary

The Bellman equation and its continuous-time counterpart, the Hamilton-Jacobi-Bellman (HJB) equation, serve as necessary conditions for optimality in reinforcement learning and optimal control. While the value function is known to be the unique solution to the Bellman equation in tabular settings, we demonstrate that this uniqueness **fails to hold** in continuous state spaces. Specifically, for linear dynamical systems, we prove the Bellman equation admits at least $\binom{2n}{n}$ solutions, where n is the state dimension. Crucially, **only one** of these solutions yields both an optimal policy and a stable closed-loop system. We then demonstrate a common failure mode in value-based methods: convergence to unstable solutions due to the exponential imbalance between stable and unstable solutions. Finally, we introduce a positive-definite neural architecture that guarantees convergence to the stable solution by construction to address this issue.

Contribution(s)

1. We prove that solutions to the Bellman equation in continuous state and action spaces are generally non-unique, even under standard assumptions (e.g., smooth dynamics, discount factors). Furthermore, we demonstrate that value-based learning approaches frequently converge to unstable solutions, directly linking this non-uniqueness to practical failures in closed-loop control.
   Context: While prior work established uniqueness in finite settings (Bellman, 1957; Blackwell, 1965; Sutton, 1988) and identified edge cases of non-uniqueness in continuous spaces (e.g., undiscounted or nonsmooth problems) (Misztela, 2018; Hosoya, 2024; Xuan et al., 2024), our results show non-uniqueness persists generically—exposing a broader, underappreciated limitation of value-based reinforcement learning.
2. For linear dynamical systems, we derive an exact lower bound of $\binom{2n}{n}$ distinct solutions to the Bellman equation, yet only *one* corresponds to a stable controller. This exponential imbalance explains why learning methods struggle to identify viable policies in high dimensions–a phenomenon we term the *curse of dimensionality in solution space*, mirroring the computational curse of dimensionality in tabular settings.
   Context: Though inspired by the solution to algebraic Riccati equation (Willems, 1971; Lancaster, 1995), we provide novel linear-algebraic proofs (avoiding functional analysis) and explicitly characterize how discount factors affect the results.
3. We introduce the use of positive-definite neural architectures to exclude unstable solutions from being learned and demonstrate their effectiveness in solving *nonlinear* control problems.
   Context: The key insight behind the use of positive-definite models is the close relationship between the value function and Lyapunov function. This opens up a new possibility to connect the value-based approach with the safe reinforcement learning community (Brunke et al., 2022; Dawson et al., 2022; 2023).

Is Bellman Equation Enough for Learning Control?

Haoxiang You[1], **Lekan Molu**[2], **Ian Abraham**[1]
{haoxiang.you,ian.abraham}@yale.edu, lekanmolu@microsoft.com

[1]**Department of Mechanical Engineering, Yale University**
[2]**Microsoft Research**

Abstract

The Bellman equation and its continuous-time counterpart, the Hamilton-Jacobi-Bellman (HJB) equation, serve as necessary conditions for optimality in reinforcement learning and optimal control. While the value function is known to be the unique solution to the Bellman equation in tabular settings, we demonstrate that this uniqueness **fails to hold** in continuous state spaces. Specifically, for linear dynamical systems, we prove the Bellman equation admits at least $\binom{2n}{n}$ solutions, where n is the state dimension. Crucially, **only one** of these solutions yields both an optimal policy and a stable closed-loop system. We then demonstrate a common failure mode in value-based methods: convergence to unstable solutions due to the exponential imbalance between admissible and inadmissible solutions. Finally, we introduce a positive-definite neural architecture that guarantees convergence to the stable solution by construction to address this issue.

1 Introduction

Reinforcement learning (RL) methods are broadly categorized into policy-based approaches (Williams, 1992; Schulman et al., 2015; 2017), which directly optimize policy parameters using gradient estimates, and value-based approaches (Watkins & Dayan, 1992; Mnih et al., 2015; Lillicrap, 2015), which indirectly learn policies by solving optimality conditions encoded in Bellman equations. While policy-based methods excel in high-dimensional dynamical system, value-based approaches benefit from sample efficiency through offline learning. In continuous-time domains, the Hamilton–Jacobi–Bellman (HJB) equation serves as the counterpart to the discrete-time Bellman equation. HJB-based RL enables control at arbitrarily high frequencies (Doya, 2000) and naturally incorporates physical priors for robust policy synthesis (Lutter et al., 2021a). Despite these advantages, practical applications remain largely limited to low-dimensional systems (Nakamura-Zimmerer et al., 2021; Lutter et al., 2023), due to fundamental challenges in solving high-dimensional HJB equations.

Two critical issues hinder the extension of HJB methods to complex systems:

Solution Existence: The value function (optimal performance index) may lack smoothness or even continuity (Bardi et al., 1997; Clarke et al., 2008). Common remedies include viscosity solutions (Crandall & Lions, 1983; Shilova et al., 2024) that generalize differentiability requirements, or stochastic regularization through noise injection (Fleming & Soner, 2006; Tassa & Erez, 2007).

Solution Uniqueness: Since the Bellman equation is only a necessary condition for optimality, it may admit general solutions beyond the value function. Early work (Bellman, 1957; Blackwell, 1965; Sutton, 1988) has shown, value function is indeed the unique solution to Bellman equation for the finite state and action space. Yet, the uniqueness of general solutions in continuous state space is overlooked in the reinforcement learning community. We show that the Bellman equation admits multiple solutions in continuous state spaces, making the performance of value-based approaches

highly dependent on model initialization. This non-uniqueness partially explains why value-based approaches are sensitive to hyperparameter choices and initialization (Ceron et al., 2024).

In this paper, we formally analyze solution spaces for both discrete-time Bellman and continuous-time HJB equations in continuous state and action space. For linear dynamical systems, we prove the Bellman equation admits at least $\binom{2n}{n}$ solutions – a number growing exponentially($\sim 4^n/\sqrt{\pi n}$ asymptotically) with state dimension n. Crucially, **only one** solution leads to a stable closed-loop system. This exponential disparity between admissible solutions and the unique stabilizing optimum fundamentally obstructs value-based learning approaches from synthesizing effective controllers. We characterize this inherent challenge in continuous spaces – the need to identify the optimal solution among exponentially many alternatives – as *the curse of dimensionality in solution space*, complementing the well-known *curse of dimensionality in computation* from tabular setting.

Our key contributions are:

1. A constructive method to derive general solutions of Bellman/HJB equations for linear systems, with theoretical characterization of their relationship to closed-loop stability via spectral analysis.
2. Identification of a canonical failure mode in practice: value-based methods converge to unstable general solutions.
3. A positive-definite neural architecture that provably constrains learning to the stable solution subspace, demonstrating its effectiveness beyond linear dynamics.

The rest of the paper is organized as following: Section 2 covers the background on problem setting, Bellman equation, and learning methods for solving them. Section 3 focus on the general solutions of Bellman equation and how they affect control. Section 4 present a canonical failure mode in value-based learning approach. Section 5 discuss techniques to handle this failure. Finally, Section 6 highlight the difference between this paper and existing work.

2 Background

This section presents the optimal control framework, introduces both continuous-time and discrete-time Bellman equations, and reviews contemporary learning methods for their solution.

2.1 Problem Setting

We consider both continuous-time and discrete-time dynamical systems in the form

$$\mathbf{x}_{k+1} = f(\mathbf{x_k}, \mathbf{u}_k) \text{ and } \dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)). \tag{1}$$

Here, $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^n$ is the state and $\mathbf{u} \in \mathcal{U} \subseteq \mathbb{R}^m$ is the control input, The control policy $\pi : \mathcal{X} \to \mathcal{U}$ maps states to actions,

$$\mathbf{u}(t) = \pi(\mathbf{x}(t)) \text{ or } \mathbf{u}_k = \pi(\mathbf{x}_k), \tag{2}$$

with associated cumulative costs:

$$\mathcal{J}^\pi(\mathbf{x}(t)) = \int_t^\infty e^{-\frac{s-t}{\tau}} l(\mathbf{x}(s), \mathbf{u}(s)) ds \text{ and } \mathcal{J}^\pi(\mathbf{x_k}) = \sum_{i=k}^{\infty} \gamma^{i-k} l(\mathbf{x}_i, \mathbf{u}_i) \tag{3}$$

Here, $l : \mathcal{X} \times \mathcal{U} \to \mathbb{R}_+$ is the running cost, $e^{-\frac{s-t}{\tau}}$ and γ^{i-k} are discount factors for future cost. The states $\mathbf{x}(\cdot)$ evolve under the dynamics (1), influenced by the policy (2). The optimal control problem seeks a policy π^* minimizing $\mathcal{J}^\pi$ for all initial states. The time-invariant formulation ensures stationarity of π^* in infinite-horizon settings.

2.2 The (Discrete-time) Bellman and Hamilton-Jacobi Bellman Equations

Control laws that optimize (3) are established by minimizing a *value function*, which is defined via the following cumulative cost,

$$\mathcal{V}(\mathbf{x}(t)) := \mathcal{J}^{\pi^*}(\mathbf{x}(t)) = \min_{\mathbf{u}[t,\infty)} \left[\int_t^\infty e^{-\frac{s-t}{\tau}} l(\mathbf{x}(s), \mathbf{u}(s)) ds \right], \tag{4}$$

or in discrete-time case,

$$\mathcal{V}(\mathbf{x}_k) := \mathcal{J}^{\pi^*}(\mathbf{x}_k) = \min_{\mathbf{u}_k, \mathbf{u}_{k+1}, \dots} \left[\sum_{i=k}^{\infty} \gamma^{i-k} l(\mathbf{x}_i, \mathbf{u}_i) \right], \tag{5}$$

where $\mathbf{u}[t, \infty)$ denotes the continuous control laws and $\{\mathbf{u}_k, \dots\}$ the discrete control sequence. According to the principle of optimality (Bellman, 1957), the value function (in the discrete-time case) can be obtained via the Bellman equation,

$$\mathcal{V}(\mathbf{x}_k) = \min_{\mathbf{u}_k} \Big[l(\mathbf{x}_k, \mathbf{u}_k) + \gamma \mathcal{V}(\mathbf{x}_{k+1}) \Big], \tag{6}$$

or for the continuous-time case via the Hamilton-Jacobi-Bellman (HJB) equation,

$$\mathcal{V}(\mathbf{x}(t)) = \min_{\mathbf{u}(t) \in \mathcal{U}} \left[l(\mathbf{x}(t), \mathbf{u}(t)) + \left\langle \frac{\partial \mathcal{V}^\top(\mathbf{x}(t))}{\partial \mathbf{x}}, f(t; \mathbf{x}, \mathbf{u}) \right\rangle \right], \quad t \in [0, T), \tag{7}$$

where the notation $f(t; \mathbf{x}, \mathbf{u})$ implies that $\mathbf{x}$ and $\mathbf{u}$ are dependent on t. The optimal policy (for discrete-time case) is obtained by solving the right-hand side of (7), that is,

$$\mathbf{u}_k := \pi^*(\mathbf{x}_k) = \operatorname*{argmin}_{\mathbf{u} \in \mathcal{U}} \Big[l(\mathbf{x}_k, \mathbf{u}) + \gamma \mathcal{V}(\mathbf{x}_{k+1}) \Big], \quad k \in [0, \cdots, K-1), \tag{8}$$

and in the continuous-time case, we have

$$\mathbf{u}^*(t) = \pi^*(\mathbf{x}(t)) = \operatorname*{argmin}_{\mathbf{u} \in \mathcal{U}} \left[l(\mathbf{x}(t), \mathbf{u}) + \left\langle \frac{\partial \mathcal{V}^\top(\mathbf{x}(t))}{\partial \mathbf{x}}, f(t; \mathbf{x}, \mathbf{u}) \right\rangle \right]. \tag{9}$$

2.3 Approximate Solutions to The Bellman and HJB Equations

For many systems, expanding the value function in (6) or (7) is impractical owing to the curse of dimensionality when expanding $\mathcal{V}(\mathbf{x}, t)$ around the state-control pairs. A practical way of obtaining *approximate* solutions is to consider a perturbation around a current system trajectory $\{\mathbf{x}(t), \mathbf{u}(t)\}$ or in discrete-time, $\{\mathbf{x}_k, \mathbf{u}_k\}$. The nominal solution that minimizes the corresponding residual costs around this perturbed trajectory (Doya, 2000; Sutton, 2018) are given as,

$$\delta(\mathbf{x}(t)) \triangleq l(\mathbf{x}(t), \mathbf{u}^*(t)) + \left\langle \frac{\partial \hat{\mathcal{V}}_\theta^\top(\mathbf{x}(t))}{\partial \mathbf{x}}, f(\mathbf{x}(t), \mathbf{u}^*(t)) \right\rangle - \hat{\mathcal{V}}_\theta(\mathbf{x}(t)), \tag{10}$$

and

$$\delta(\mathbf{x}_k) \triangleq l(\mathbf{x}_k, \mathbf{u}_k^*) + \gamma \hat{\mathcal{V}}_\theta(f(\mathbf{x}_k, \mathbf{u}_k^*)) - \hat{\mathcal{V}}_\theta(\mathbf{x}_k), \tag{11}$$

where, $\hat{\mathcal{V}}_\theta : \mathcal{X} \times \Theta \to \mathbb{R}$ is a parameterized candidate solution to (6). In tabular dynamic programming settings, this minimization is applied to the entire state space via value iteration (see Appendix B). For large state spaces or in the continuous-time setting, a subset of the states are updated as the numerical integration evolves. These subsets are collected either through Monte Carlo rollouts (Mnih et al., 2015; Lillicrap, 2015) or via grid sampling (Shilova et al., 2024). The notation $\mathbf{u}^*$ represents the current estimate of the optimal action that minimizes the right-hand side of the Bellman equation. In practice, $\mathbf{u}^*$ is executed by training another policy network (Lillicrap, 2015) or solved analytically (Gu et al., 2016; Lutter et al., 2021b; Molu et al., 2018).

3 General Solutions to The Bellman and HJB Equations

We describe the general solutions to the Bellman equation and relations to learning for control. We begin our analysis with a continuous-time linear dynamical problem and then extend the study to the discrete-time case. We complete this section with an analysis of nonlinear problems.

3.1 Solutions to Linear Continuous-Time Control Problem

let us consider the continuous-time version of the linear quadratic regulator (LQR) optimal control problem,

$$\begin{array}{ll} \min & \int_t^\infty \mathbf{x}(s)^\top Q\mathbf{x}(s) + \mathbf{u}(s)^\top R\mathbf{u}(s)ds \\ \text{subject to} & \dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t), \end{array} \tag{12}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $Q = Q^\top \succeq 0 \in \mathbb{R}^{n\times n}$, and $R = R^\top \succ 0 \in \mathbb{R}^{m\times m}$ are constant matrices. Suppose that the value function is quadratic in the state such that

$$\mathcal{V}(\mathbf{x}(t)) = \mathbf{x}^\top(t)P\mathbf{x}(t), \tag{13}$$

for a symmetric and positive-definite matrix P. Substituting (13) and (12) into (7), we find that

$$0 = \min_{\mathbf{u}(t)} \left[\mathbf{u}(t)^\top R\mathbf{u}(t) + 2\mathbf{x}(t)^\top PB\mathbf{u}(t) + \mathbf{x}(t)^\top Q\mathbf{x}(t) + \mathbf{x}(t)^\top PA\mathbf{x}(t) + \mathbf{x}(t)^\top AP\mathbf{x}(t)\right]. \tag{14}$$

Notably, the minimizing problem on the right-hand side is quadratic in $\mathbf{u}(t)$. Thereforce, we can obtain the analytical optimal control

$$\mathbf{u}^*(t) = -R^{-1}B^\top P\mathbf{x}(t), \tag{15}$$

which yields

$$\mathbf{x}^\top(t)(A^\top P + PA - PBR^{-1}B^\top P + Q)\mathbf{x}(t) = 0. \tag{16}$$

The HJB equation must be satisfied globally; therefore, the following continuous algebraic Riccati equation (ARE) must hold for the matrix P,

$$A^\top P + PA - PBR^{-1}B^\top P + Q = 0. \tag{17}$$

Conversely, any matrix P that satisfies the ARE (17) is a solution to HJB equation of the LQR problem.

Effect of Discount Factor In the formulation of LQR problem (17), we omit the discount factor $e^{-\frac{s}{\tau}}$. This omission is made only for simplicity and by no means affect the generality of the results we will show shortly. Particularly considering the LQR problem with discounted running cost

$$\begin{array}{ll} \min & \int_t^\infty e^{-\frac{s}{\tau}}[\mathbf{x}(s)^\top Q\mathbf{x}(s) + \mathbf{u}(s)^\top R\mathbf{u}(s)]ds \\ \text{subject to} & \dot{\mathbf{x}}(s) = A\mathbf{x}(s) + B\mathbf{u}(s), \quad \mathbf{x}(0) = \mathbf{x}_0 \end{array} \tag{18}$$

and its undiscounted counterpart with modified dynamics

$$\begin{array}{ll} \min & \int_t^\infty \mathbf{x}(s)^\top Q\mathbf{x}(s) + \mathbf{u}(s)^\top R\mathbf{u}(s)ds \\ \text{subject to} & \dot{\mathbf{x}}(s) = (A - \frac{1}{2\tau}I)\mathbf{x}(s) + B\mathbf{u}(s), \quad \mathbf{x}(0) = \mathbf{x}_0 \end{array}, \tag{19}$$

we have the following result.

Theorem 1. *The value function $\mathcal{V}(\mathbf{x}) = \mathbf{x}^\top P\mathbf{x}$ satisfies the HJB equation for the discounted LQR problem (18) if and only if it satisfies the HJB equation for the undiscounted LQR problem with modified dynamics (19).*

Proof. Substituting the discounted LQR problem (18), $\mathcal{V}(\mathbf{x}) = \mathbf{x}^\top P\mathbf{x}$, and policy $\mathbf{u}^*(t) = -R^{-1}B^\top P\mathbf{x}(t)$ into HJB equation (7) yields

$$\mathbf{x}^\top(t)(A^\top P + PA - PBR^{-1}B^\top P + Q)\mathbf{x}^\top(t) = \mathbf{x}^\top(t)P\mathbf{x}(t). \tag{20}$$

Rearranging the equation (20), we have

$$\mathbf{x}^\top(t)[(A - \frac{1}{2\tau}I)^\top P + P(A - \frac{1}{2\tau}I) - PBR^{-1}B^\top P + Q]\mathbf{x}^\top(t) = 0, \tag{21}$$

which matches equation (16) with modified dynamics $\dot{\mathbf{x}}(s) = (A - \frac{1}{2\tau}I)\mathbf{x}(s) + B\mathbf{u}(s)$. □

The stability ofthe opened-loop system $\dot{\mathbf{x}}(s) = A\mathbf{x}(s)$ depends solely on the eigenvalues of A. Specially, the system becomes exponentially stable as the eigenvalues of A become negative. Therefore, an interesting interpretation of Theorem 1 is that, in terms of the value function, incorporating a discount factor into the running cost is equivalent to controlling a more stable system.

Solutions to Algebraic Riccati Equation Here, we introduce a subspace invariant method (Lancaster, 1995) to obtain a general solution for the ARE (17). We begin by construct a $2n \times 2n$ a Hamiltonian matrix:

$$H = \begin{bmatrix} A & -BR^{-1}B^\top \\ -Q & -A^\top \end{bmatrix}. \tag{22}$$

Definition 1. *We say $\mathcal{M}$ is an invariant subspace of H if $H\mathbf{x} \in \mathcal{M}$ for all $\mathbf{x} \in \mathcal{M}$.*

Theorem 2. *Suppose $[P_1^\top, P_2^\top]^\top$ forms a basis of invariant subspace of H (22), where P_1, P_2 are $n \times n$ matrices and P_1 invertible. Then*

$$P = P_2P_1^{-1} \tag{23}$$

is a solution to ARE (17).

Proof. By the definition of the invariant subspace, there exist a $n \times n$ matrix T such that

$$H\begin{bmatrix} P_1 \\ P_2 \end{bmatrix} = \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} T. \tag{24}$$

Since P_1 is invertible, we have

$$\begin{bmatrix} A & -BR^{-1}B^\top \\ -Q & -A^\top \end{bmatrix}\begin{bmatrix} P_1 \\ P_2 \end{bmatrix} P_1^{-1} = \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} P_1^{-1}P_1TP_1^{-1} = \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} P_1^{-1}\hat{T}. \tag{25}$$

so that

$$\begin{bmatrix} A & -BR^{-1}B^\top \\ -Q & -A^\top \end{bmatrix}\begin{bmatrix} I \\ P_2P_1^{-1} \end{bmatrix} = \begin{bmatrix} \hat{T} \\ P_2P_1^{-1}\hat{T} \end{bmatrix} \tag{26}$$

These give two set of matrix equation

$$\hat{T} = A - BR^{-1}B^\top P_2P_1^{-1}, \tag{27a}$$

$$-Q - A^\top P_2P_1^{-1} = P_2P_1^{-1}\hat{T}. \tag{27b}$$

Substituting equation (27a) into (27b) we find that

$$Q + A^\top P_2P_1^{-1} + P_2P_1^{-1}A - P_2P_1^{-1}BR^{-1}B^\top P_2P_1^{-1} = 0, \tag{28}$$

which follow the ARE form (17) and complete the proof. □

We can then apply eigen-decomposition[1] to obtain invariant subspace for H,

$$H = V\Lambda V^{-1} = \begin{bmatrix} P_1 & S_1 \\ P_2 & S_2 \end{bmatrix} \begin{bmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{bmatrix} \begin{bmatrix} P_1 & S_1 \\ P_2 & S_2 \end{bmatrix}^{-1}, \tag{29}$$

where $[S_1^\top, S_2^\top]^\top$ is a $2n \times n$ matrices that stores other n eigenvectors. By choosing different combination of columns in the eigen-space, we can form $\binom{2n}{n}$ different invariant subspaces, with each of them yielding at least a solution to the HJB equation.

3.2 Behavior of The Closed-Loop System under General Solutions

Substituting the policy (15) into the open-loop dynamics in (12) yields the dynamics of the closed-loop system:

$$\dot{\mathbf{x}}(t) = (A - BR^{-1}B^\top P)\,\mathbf{x}(t) = A_{cl}\mathbf{x}(t), \tag{30}$$

where A_{cl} is the closed-loop transition matrix. Here, we study the behavior of closed-loop system (30) by relating the eigenvalues of A_{cl} to H.

Lemma 3. *If the pair (A, B) is controllable and the pair (Q, A) is observable, the Hamiltonian matrix H (22) has no pure imaginary or zero eigenvalues.*

Proof. See Appendix A. □

Theorem 4. *If λ is an eigenvalue of H, then $-\lambda$ is also an eigenvalues. Furthermore, if the pair (A, B) is controllable and the pair (Q, A) is observable, H has n positive eigenvalues and n negative eigenvalues.*

Proof. Let $J = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}$, then we must have $HJ = (HJ)^\top$.

So $H\mathbf{x} = \lambda\mathbf{x}$ implies

$$J^\top HJJ^\top\mathbf{x} = \lambda J^\top\mathbf{x}$$
$$H^\top(J^\top\mathbf{x}) = -\lambda(J^\top\mathbf{x})$$

Hence, $-\lambda$ is an eigenvalue of $H^\top$ and must be an eigenvalue of H.

By Lemma 3, the eigenvalues only contains real number, so it must have n positive eigenvalues and n negative eigenvalues. □

Theorem 5. *Suppose $[P_1^\top, P_2^\top]^\top$ are invariant subspace of H (22) with eigenvalues $\Lambda_1 = diag(\lambda_1, \ldots, \lambda_n)$ and $P = P_2P_1^{-1}$, then the characteristic matrix of closed-loop system $A_{cl} = (A - BR^{-1}B^\top P)$ has eigenvalues Λ_1.*

Proof. By invariant subspace, we have

$$\begin{bmatrix} A & -BR^{-1}B^\top \\ -Q & -A^\top \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} = \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} \Lambda_1, \tag{31}$$

which give

$$\begin{aligned} AP_1 - BR^{-1}B^\top P_2 &= P_1\Lambda_1 \\ -QP_1 - A^\top P_2 &= P_2\Lambda_1 \end{aligned} \tag{32}$$

Substituting $P_2 = PP_1$ into the first equation, we obtain

$$(A - BR^{-1}B^\top P)P_1 = P_1\Lambda_1, \tag{33}$$

which completes the proof. □

[1] If H is not diagonalizable, we can still apply the Schur decomposition. The results remain the same in this case.

Corollary 6. *Given the conditions in Lemma 3, and assuming that H is diagonalizable, there exists at least $\binom{2n}{n}$ P matrices that satisfy the CARE (17). In addition,* ***only one*** *of the solutions contributed to a closed-loop stable system (30).*

Proof. By picking different sets of n eigenvector from $2n$-dimensional eigenspace of (29), we can construct $\binom{2n}{n}$ invariant subspaces $\mathcal{M}$s, each invariant subspace corresponding to a solution P. By Theorem 4 and Theorem 5, the closed-loop system is stable, only when eigenvectors with all negative λ are selected. □

A similar result to Corollary 6 holds for the discounted cost setting, where at most one solution leads to a stable closed-loop system. More details are provided in Appendix A.

3.3 The Discrete-Time Case

The results for the discrete-time case are similar to their continuous-time counterpart; however, the proof is more involved. Therefore, we present the main theorems for the discrete-time system in the main text and leave the proof in Appendix A.

We consider the discrete-time LQR prblem:

$$\begin{array}{ll} \min & \sum_{i=t}^{\infty} \mathbf{x}_i^\top Q \mathbf{x}_i + \mathbf{u}_i^\top R \mathbf{u}_i \\ \text{subject to} & \mathbf{x}_{i+1} = A\mathbf{x}_i + B\mathbf{u}_i, \end{array} . \tag{34}$$

The discrete-time algebraic-Riccati equation of (34) is given by

$$P = Q + A^\top P A - A^\top P B (R + B^\top P B)^{-1} B^\top P A. \tag{35}$$

Similar to the continuous-time case, we construct the Hamiltonian matrix as:

$$H = \begin{bmatrix} A + BR^{-1}B^\top A^{-\top} Q & -BR^{-1}B^\top A^{-\top} \\ -A^{-\top} Q & A^{-\top} \end{bmatrix}. \tag{36}$$

Theorem 7. *Suppose $[P_1^\top, P_2^\top]^\top$ are invariant subspace of discrete-time Hamitonian matrix H (36), then $P = P_2 P_1^{-1}$ is the solution to discrete-time algebraic-Riccati equation (35).*

Proof. See Appendix A. □

In the discrete-time problem, the dynamics of closed-loop is given by

$$\mathbf{x}_{i+1} = (A - B(R + B^\top P B)^{-1} B^\top P A), \quad \mathbf{x}_i = A_{cl} x_i, \tag{37}$$

where stability is determined by the eigenvalues of A_{cl} being within the unit circle. We now relate the eigenvalues of closed-loop system matrix A_{cl} to the Hamiltonian H for the discrete-time system.

Theorem 8. *If λ is an eigenvalue of the discrete-time Hamiltonian matrix H (36), then $\frac{1}{\lambda}$ is also a eigenvalue of H. Moreover, if the pair (A, B) is controllable and the pair (Q, A) is observable, then H has n eigenvalues within the unit circle and n eigenvalues outside the unit cycle.*

Proof. See Appendix A. □

Theorem 9. *Suppose $[P_1^\top, P_2^\top]^\top$ is the invariant subspace of H (36) with eigenvalues $\Lambda_1 = diag(\lambda_1, \ldots, \lambda_2)$ and $P = P_2 P_1^{-1}$, then the characteristic matrix of closed-loop system $A_{cl} = (A - B(R + B^\top P B)^{-1} B^\top P A)$ has eigenvalues Λ_1.*

Proof. See Appendix A □

Given Theorem 8 and Theorem 9, we have similar results for the discrete-time system, i.e., only one among $\binom{2n}{n}$ general solution to Bellman equation leads to stable closed-loop system.

3.4 The Nonlinear Case

Here, we discuss the general solutions to HJB equation for nonlinear dynamics. In many applications, optimal control problems are highly structured. For example, in robotics, a common assumption is that the dynamics are control-affine and the running cost is separable and convex in the control input. In such scenarios, the optimal control problem can often be solved analytically (Lutter et al., 2021b). Specifically, consider the optimal control problem formulated as follows:

$$\begin{array}{ll} \min & \int_t^\infty e^{-\frac{s}{\tau}}[l_1(\mathbf{x}(s)) + \mathbf{u}(s)^\top R\mathbf{u}(s)]ds \\ \text{subject to} & \dot{\mathbf{x}}(s) = f_1(\mathbf{x}(s)) + f_2(\mathbf{x}(s))\mathbf{u}(s), \quad \mathbf{x}(0) = x_0, \end{array} \tag{38}$$

where $f_1 : \mathcal{X} \to \mathbb{R}^n, f_2 : \mathcal{X} \to \mathbb{R}^{n\times m}, l_1 : \mathcal{X} \to \mathbb{R}_+$ and $R = R^\top \succ 0 \in \mathbb{R}^{m\times m}$.

The optimal policy synthesis problem (9) is quadratic in control and can be obtained analytically:

$$\begin{aligned} \mathbf{u}^*(t) =& \underset{\mathbf{u}\in\mathcal{U}}{\operatorname{argmin}} \left[l_1(\mathbf{x}(t)) + \mathbf{u}^\top R\mathbf{u} + \left\langle \frac{\partial \mathcal{V}^\top(\mathbf{x}(t))}{\partial \mathbf{x}}, (f_1(\mathbf{x}(t)) + f_2(\mathbf{x}(t))\mathbf{u}) \right\rangle \right] \\ =& -\frac{1}{2}R^{-1}f_2^\top(\mathbf{x}(t))\frac{\partial \mathcal{V}(\mathbf{x}(t))}{\partial \mathbf{x}}. \end{aligned} \tag{39}$$

Substituting the HJB equation (7) into (39) yields the elliptic partial differential equation:

$$\mathcal{V}(\mathbf{x}(t)) = -\frac{1}{4}\frac{\partial \mathcal{V}(\mathbf{x}(t))}{\partial \mathbf{x}}^\top f_2(\mathbf{x}(t))R^{-1}f_2(\mathbf{x}(t))^\top \frac{\partial \mathcal{V}(\mathbf{x}(t))}{\partial \mathbf{x}} + f_1(\mathbf{x}(t))^\top \frac{\partial \mathcal{V}(\mathbf{x}(t))}{\partial \mathbf{x}} + l_1(\mathbf{x}(t)). \tag{40}$$

Thus, solving the HJB equation for the nonlinear control problem (38) is equivalent to solving the elliptic partial differential equation (40).

In general, the elliptic partial differential equation (40) does not admit a unique solution. However, the discussion of general solutions to nonlinear elliptic partial differential equations is beyond the scope of this paper, and we refer readers to Han & Lin (2011); Evans (2022) for further details.

4 A Canonical Failure Mode in Learning Value Functions

A generic neural network may converge to any solution of the Bellman equation by minimizing the TD error. Moreover, it is likely for the network to converge to undesired solution given the imbalance ratio of solutions. Here, we present an example to show such failure mode do exist in practice.

Consider the time-continuous LQR (12) with the parameters $A = B = Q = R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$.

We obtain the analytical solution by using the subspace invariant method discussed previously:

$$H = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ -1 & 0 & -1 & 0 \\ 0 & -1 & 0 & -1 \end{bmatrix} = V\Lambda V^{-1}, \tag{41}$$

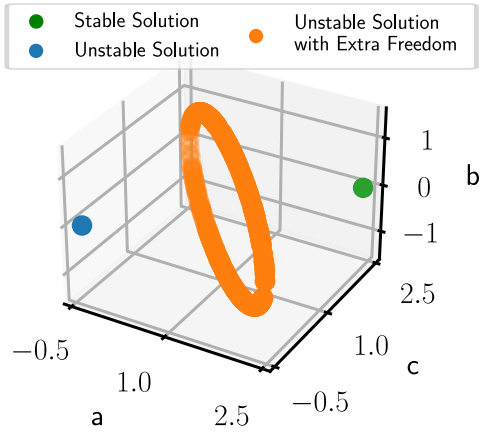

Figure 1: Solution to LQR: $\mathcal{V}(\mathbf{x}) = \mathbf{x}^\top P\mathbf{x}$, where P is given by (43). The green dot indicates the stable solution, the blue dot marks the unstable solution, and the orange ring represents additional solutions arising from the noninvertibility of P_1.

where

$$V = \begin{bmatrix} 1+\sqrt{2} & 0 & 1-\sqrt{2} & 0 \\ 0 & 1+\sqrt{2} & 0 & 1-\sqrt{2} \\ -1 & 0 & -1 & 0 \\ 0 & -1 & 0 & -1 \end{bmatrix}, \Lambda = \begin{bmatrix} \sqrt{2} & 0 & 0 & 0 \\ 0 & \sqrt{2} & 0 & 0 \\ 0 & 0 & -\sqrt{2} & 0 \\ 0 & 0 & 0 & -\sqrt{2} \end{bmatrix}. \tag{42}$$

By selecting columns from the matrix V, we can obtain different solution matrices P. Note that when the first and third or second and fourth columns are selected, i.e., $P_1 = \begin{bmatrix} 1+\sqrt{2} & 1-\sqrt{2} \\ 0 & 0 \end{bmatrix}, P_2 = \begin{bmatrix} -1 & -1 \\ 0 & 0 \end{bmatrix}$ or $P_1 = \begin{bmatrix} 0 & 0 \\ 1-\sqrt{2} & 1-\sqrt{2} \end{bmatrix}, P_2 = \begin{bmatrix} 0 & 0 \\ -1 & -1 \end{bmatrix}$, the matrix P_1 is not invertible, introducing additional degrees of freedom. This leads to an infinite number of unstable solutions.In summary, the analytical solution can be expressed as $P = \begin{bmatrix} a & b \\ b & c \end{bmatrix}$, where

$$\begin{cases} a = 1+\sqrt{2}, b = 0, c = 1+\sqrt{2} \text{ (Stable Solution)} \\ a = 1-\sqrt{2}, b = 0, c = 1-\sqrt{2} \text{ (Unstable Solution)} \\ a = z, b = \sqrt{2z+1-z^2}, c = 2-z, z \in (1-\sqrt{2}, 1+\sqrt{2}) \text{ (Unstable with Extra Freedom)} \end{cases}. \tag{43}$$

These solutions are visualized in Figure 1.

Now, we solve the LQR problem with a learning approach. Specifically, we use a 3-layer MLP architecture with 128 neurons in each layer and the ELU activation function to approximate the value function. The weights are initialized with Lecun normal (Klambauer et al., 2017) and the biases are initialized to zero. We collect 10^4 samples uniformally from $[-2, 2] \times [-2, 2]$. The neural network is then trained by minimizing the mean square error of TD residuals (10) using Adam optimizer with learning rate 10^{-3}. We train the networks for 1000 epochs and achieve a mean square error below 10^{-4}.

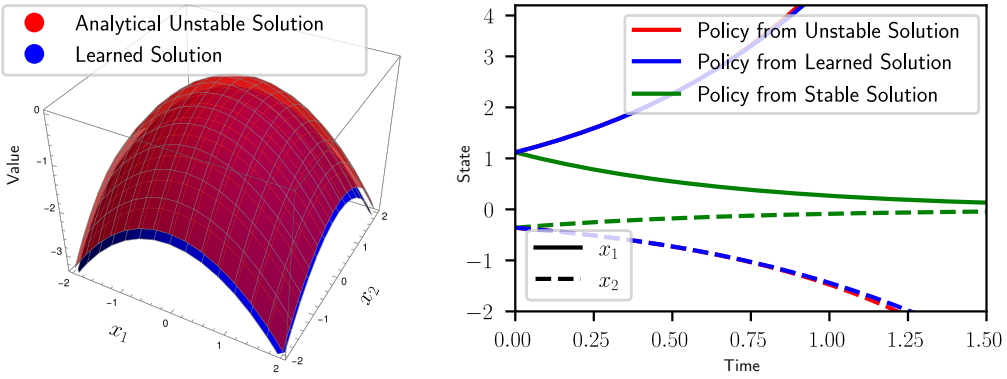

Figure 2: Learned solution to LQR. Left figure: the learned value function converges to the unstable solution, up to an additive constant. Right figure: both the learned and analytical unstable solutions yield identical diverging trajectories, whereas the stable solution converges to the origin.

Figure 2 compares the learned and analytical solutions. In this case, the neural network converges to the unstable solution, and the associated policy causes trajectories to diverge. The issue of non-unique solutions also arises in **nonlinear** dynamics, as we will demonstrate in Section 5. Moreover, we find the solution to which the network converges is highly sensitive to weight initialization (see Appendix C), suggesting that multiple solutions may partly explain the sensitivity of value-based approaches to hyperparameters and random seeds (Ceron et al., 2024).

5 What Can We Do

In this section, we discuss potential solutions to address the failure of the value-based approach in finding the optimal control policy due to the existence of general solutions to the Bellman equation.

5.1 Approach 1: Adding Boundary Conditions

Setting boundary conditions is one of the most common techniques for isolating the stable solution of the Bellman equation (Mnih et al., 2015; Lillicrap, 2015). However, selecting appropriate boundary conditions can be challenging in practice. Insufficient boundary conditions may fail to guarantee uniqueness, while inconsistent ones can result in no solution at all.

Insufficient Boundary Conditions Considering the LQR example in Section 4, we can define a unit circle boundary $\mathcal{B} = \{\mathbf{x} : \mathbf{x}^\top \mathbf{x} = 1\}$ and set the Dirichlet boundary condition: $\mathcal{V}(\mathbf{x}) = 1 + \sqrt{2}$, for $\mathbf{x} \in \mathcal{B}$.

Under the specified boundary conditions, the stable solution, the stable solution $\mathcal{V}(\mathbf{x}) = \mathbf{x}^\top \begin{bmatrix} 1+\sqrt{2} & 0 \\ 0 & 1+\sqrt{2} \end{bmatrix} \mathbf{x}$ still solves the associated Bellman equation. However, there is another solution $\mathcal{V}(\mathbf{x}) = \mathbf{x}^\top \begin{bmatrix} 1-\sqrt{2} & 0 \\ 0 & 1-\sqrt{2} \end{bmatrix} \mathbf{x} + 2\sqrt{2}$, which meets both the Bellman equation and the boundary conditions. This solution is obtained by adding an offset of $2\sqrt{2}$ to the unstable solution $\mathcal{V}(\mathbf{x}) = \mathbf{x}^\top \begin{bmatrix} 1-\sqrt{2} & 0 \\ 0 & 1-\sqrt{2} \end{bmatrix} \mathbf{x}$ by $2\sqrt{2}$. This phenomenon occurs because the Bellman equation and the resulting policy depend only on the relative values of the value function rather than its absolute magnitude.

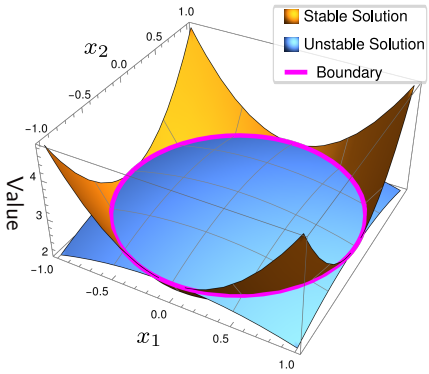

Figure 3: Example of insufficient boundary conditions: both solutions share the same boundary value, yet one yields a stable closed-loop system while the other results in instability.

Inconsistent Boundary Conditions Here, we retain the Dirichlet boundary conditions from the previous paragraph i.e., $\mathcal{V}(\mathbf{x}) = 1 + \sqrt{2}$, if $\mathbf{x} \in \mathcal{B}_1$, where $\mathcal{B}_1 = \{\mathbf{x} : \mathbf{x}^\top \mathbf{x} = 1\}$. Additionally, we introduce a new boundary: $\mathcal{B}_2 = \{\mathbf{x} : \mathbf{x}^\top \mathbf{x} = 0.5\}$, and set the Dirichlet boundary condition for this new boundary as: $\mathcal{V}(\mathbf{x}) = 0$ if $\mathbf{x} \in \mathcal{B}_2$. In this case, no general solution to Bellman equation can satisfy both boundary conditions.

5.2 Approach 2: Special Neural Architectures

For many optimal control problems, the goal is to stabilize the system to an equilibrium point $(\mathbf{x}_{\text{eq}}, \mathbf{u}_{\text{eq}})$ or track a reference trajectory. In this scenario, we can design specialized architectures to exclude the unstable general solutions that the Bellman equation can admit.

Theorem 10. *If $\mathcal{V}(\mathbf{x})$ is a solution to HJB equation (7) without discounted running cost, where $\mathcal{V}(\mathbf{x}_{eq}) = 0$ and $\mathcal{V}(\mathbf{x}) > 0$ for any $\mathbf{x} \neq \mathbf{x}_{eq}$, then the closed-loop system under the policy given by (9) is stable. Furthermore, if $l(\mathbf{x}, \mathbf{u}) > 0$ for any state-control pair $(\mathbf{x}, \mathbf{u})$ other than $(\mathbf{x}_{eq}, \mathbf{u}_{eq})$, then $\lim_{s\to\infty} \|\mathbf{x}(s) - \mathbf{x}_{eq}\| = 0$ for any initial condition $\mathbf{x}(t)$.*

Proof. See Appendix A. □

The key insight here is the value function $\mathcal{V}(\mathbf{x})$ can naturally serve as a Lyapunov function for the closed-loop system with the policy given by (9). Particularly, Bellman equation guarantees $\dot{\mathcal{V}}(\mathbf{x}) < 0$ under the policy (9). Additionally, the definition of the optimal control problem and value function ensure the positive-definiteness conditions for the Lyapunov function, i.e., $\mathcal{V}(\mathbf{x}_{\text{eq}}) = 0$ and $\mathcal{V}(\mathbf{x}) > 0$ for any $\mathbf{x} \neq \mathbf{x}_{\text{eq}}$.

Theorem 10 inspires us to design a neural architecture that preserves the positive-definiteness, i.e., ensuring, regardless of how weight θ are set, we always have $\hat{\mathcal{V}}_\theta(\mathbf{x}_{\text{eq}}) = 0$ and $\hat{\mathcal{V}}_\theta(\mathbf{x}) > 0$ for any

$\mathbf{x} \neq \mathbf{x}_{\text{eq}}$. We provide a positive-definite architecture below

$$\hat{\mathcal{V}}_\theta(\mathbf{x}) = h_{N_h}^\top h_{N_h} + \epsilon \|\mathbf{x} - \mathbf{x}_{eq}\|_2^2, \tag{44}$$

where

$$h_{k+1} = \sigma(\theta_k h_k) \text{ for } k = 1, 2, \ldots, N_h \text{ , and } h_1 = \mathbf{x} - \mathbf{x}_{eq}. \tag{45}$$

Here, σ represents nonlinear activation function that satisfies $\sigma(0) = 0$ (e.g., ReLU, tanh), and ϵ is a small constant (e.g., 10^{-3}). For any $\mathbf{x} \neq \mathbf{x}_{eq}$, we have $h_{N_h}^\top h_{N_h} \geq 0$ and $\|\mathbf{x} - \mathbf{x}_{eq}\|_2^2 > 0$, which implies that $\hat{\mathcal{V}}_\theta(\mathbf{x}) > 0$. For $\mathbf{x}_{eq}$, we have $h_1 = 0$, and by property of σ, $h_{k+1} = 0$ for $k \geq 1$; consequently, $\hat{\mathcal{V}}_\theta(\mathbf{x}_{eq}) = 0$.

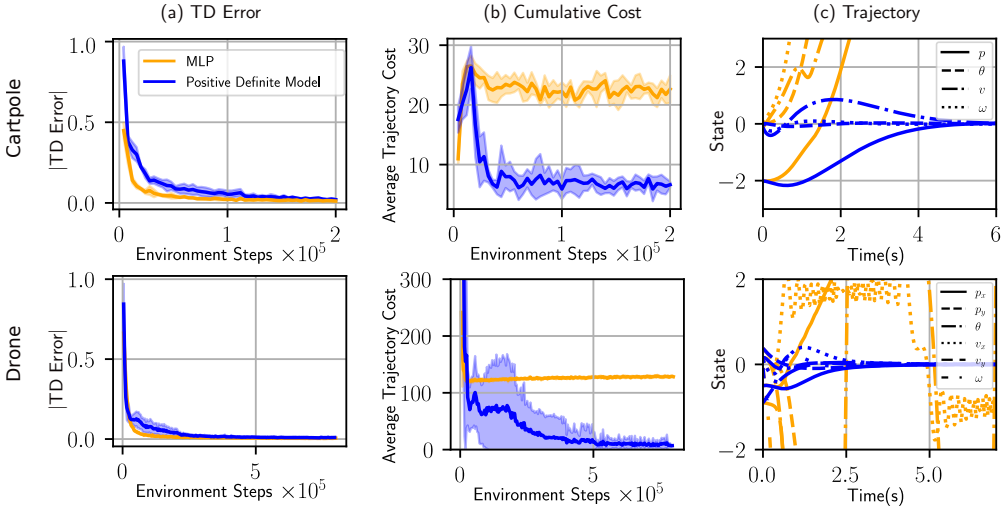

Figure 4: Value learning with MLP and positive-definiteness architecture. For each case, we run the experiments with 5 different seeds, reporting the average performance. The shaded area represents the standard deviation across the runs. The TD error is minimized for both architecture, indicating a solution to Bellman equation is found. However, generic MLP architecture cannot distinguish the stable solution from the others leading to high cumulative cost and diverging behavoir

We apply the proposed architecture to two **nonlinear** dynamical systems: cartpole and drone. The drone scenario poses additional challenges due to its intricate dynamics, the need for rapid response, and limited control inputs. Further details regarding the tasks, learning algorithm, and hyperparameters are provided in Appendix D.

Figure 4 illustrates the results for both our proposed architecture and a generic MLP. The absolute TD error is minimized for both architectures, implying that the learning algorithm successfully finds a solution to the Bellman equation in both cases. However, the generic MLP architecture cannot distinguish the stable solution from the others, the resulting controller leads to an unstable closed-loop system, causing higher cumulative costs. These results highlight the presence of non-unique solutions in systems with **nonlinear** dynamics. In contrast, by using an architecture that preserves positive definiteness, the learning algorithm consistently converges to the stable solution and achieves lower cumulative costs.

We compare our method to Lutter et al. (2020; 2021b)'s continuous-time RL approach, which employs discount factor i.e., $\frac{s}{\tau}$, scheduling and Lagrangian networks (Lutter et al., 2019) for learning the stable solution. Evaluations on three tasks (including their original and the more challenging drone system) show our method outperforms theirs, particularly on the drone task (Table 1). We find that a large discount factor $\frac{s}{\tau}$ complicates

Table 1: Trajectory Cost for Final Policy

Methods	Linear	Cartpole	Drone
Ours	0.87	6.98	8.64
Lutter et al. (2020)	4.38	10.28	362.31

optimization, leading the problem essentially ill-defined (Tassa & Erez, 2007). Moreover, the Lagrangian network does not guarantee that the equilibrium point is the unique minimizer (Lutter et al., 2023)—a key property for learning the stable solution. We hypothesize that these factors account for the performance gap between our method and that of Lutter et al. (2020; 2021b).

6 Related Work

Uniqueness of Solution to Bellman Equation The uniqueness of Bellman equations has been studied in both reinforcement learning and control. The first proof for the uniqueness of the Bellman equation in finite state and action spaces was provided by Richard Bellman (Bellman, 1957). A well-known contraction mapping proof was given by Blackwell (1965), also included in Appendix B. This idea was extended by Sutton (1988). The study in continuous state and action spaces was pioneered by Bertsekas & Shreve (1996), who showed the value function is the unique fixed point of the Bellman equation under regularity conditions.

Recent works have explored non-uniqueness in Bellman equations. For example, Misztela (2018) shows two distinct semi-continuous solutions to an HJB equation, and Hosoya (2024) finds infinitely many solutions in a special class of HJB equations. Additionally, Xuan et al. (2024) shows that a discount factor of one leads to multiple solutions. In this work, we demonstrate that non-uniqueness is common in practice, and that the solution space can grows exponentially with state dimension. We further study the behavior of the closed-loop system and show that only one solution leads to stability, which presents challenges for value-based methods. Several key theorems in this work are inspired by solutions to algebraic Riccati equations in control (Willems, 1971; Lancaster, 1995). We provide an alternative, self-contained, linear-algebraic proof for our main results, without relying on functional analysis tools, contributing to the theoretical foundation.

Neural Architectures for Dynamics and Control The need for stable dynamics and control has led to the development of structured architectures. Kolter & Manek (2019) propose input-convex networks for stable dynamical systems. Meanwhile Lutter et al. (2019) introduce Lagrangian networks for physically plausible dynamics. These architectures have been applied to control tasks (Lutter, 2023). Positive-definiteness architectures are often explored in safety-critical reinforcement learning (Chang & Gao, 2021; Brunke et al., 2022; Dawson et al., 2022; 2023), particularly due to their use of Lyapunov or barrier functions. In this work, we show a close relationship between value functions and Lyapunov functions, demonstrating how structured architectures can facilitate training and enhance policy performance for value-based methods.

7 Conclusion

In this work, we show that solutions to the Bellman equation are generally nonunique in continuous state and action spaces, with the solution space potentially growing exponentially with dimension, a phenomenon we term the "curse of dimensionality in solution space". While one might mitigate this by setting boundary conditions or designing initialization strategies, we demonstrate that this is often challenging in practice. Instead, we propose using structured architectures to constrain the search space to only desirable solutions, leveraging the close relationship between value functions and Lyapunov functions.

The primary limitation of this architecture is its applicability mainly to control problems that demand system stability. In problems such as locomotion, the objective is often to achieve a limit cycle (Tedrake, 2023) rather than converging to an equilibrium point. In these scenarios, a more effective initialization scheme may be necessary to find the desired solution, as we discussed in Appendix C. Another alternative is to switch to policy-based approaches (Schulman et al., 2017; Xu et al., 2021). These methods do not face the challenges of distinguishing solutions that value-based approaches encounter, as they directly optimize the policy with respect to the performance index.

References

Martino Bardi, Italo Capuzzo Dolcetta, et al. *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations*, volume 12. Springer, 1997.

Richard E. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, 1957.

Dimitri Bertsekas and Steven E Shreve. *Stochastic optimal control: the discrete-time case*, volume 5. Athena Scientific, 1996.

David Blackwell. Discounted dynamic programming. *Annals of Mathematical Statistics*, 36(4): 226–235, 1965.

Lukas Brunke, Melissa Greeff, Adam W Hall, Zhaocong Yuan, Siqi Zhou, Jacopo Panerati, and Angela P Schoellig. Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 5(1):411–444, 2022.

Johan Samir Obando Ceron, João Guilherme Madeira Araújo, Aaron Courville, and Pablo Samuel Castro. On the consistency of hyper-parameter selection in value-based deep reinforcement learning. *Reinforcement Learning Journal*, 3:1037–1059, 2024.

Ya-Chien Chang and Sicun Gao. Stabilizing neural control using self-learned almost lyapunov critics. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1803–1809. IEEE, 2021.

Francis H Clarke, Yuri S Ledyaev, Ronald J Stern, and Peter R Wolenski. *Nonsmooth analysis and control theory*, volume 178. Springer Science & Business Media, 2008.

Michael G Crandall and Pierre-Louis Lions. Viscosity solutions of hamilton-jacobi equations. *Transactions of the American mathematical society*, 277(1):1–42, 1983.

Charles Dawson, Sicun Gao, and Chuchu Fan. Safe control with learned certificates: A survey of neural lyapunov, barrier, and contraction methods. *arXiv preprint arXiv:2202.11762*, 2022.

Charles Dawson, Sicun Gao, and Chuchu Fan. Safe control with learned certificates: A survey of neural lyapunov, barrier, and contraction methods for robotics and control. *IEEE Transactions on Robotics*, 39(3):1749–1767, 2023.

Kenji Doya. Reinforcement learning in continuous time and space. *Neural computation*, 12(1): 219–245, 2000.

Lawrence C Evans. *Partial differential equations*, volume 19. American Mathematical Society, 2022.

Wendell H Fleming and Halil Mete Soner. *Controlled Markov processes and viscosity solutions*, volume 25. Springer Science & Business Media, 2006.

Tanmay Gangwani. Value iteration, policy iteration and policy gradient, 2019. URL https://yuanz.web.illinois.edu/teaching/IE498fa19/lec_16.pdf.

Shixiang Gu, Timothy Lillicrap, Ilya Sutskever, and Sergey Levine. Continuous deep q-learning with model-based acceleration. In *International conference on machine learning*, pp. 2829–2838. PMLR, 2016.

Qing Han and Fanghua Lin. *Elliptic partial differential equations*, volume 1. American Mathematical Soc., 2011.

Joao P Hespanha. *Linear systems theory*. Princeton university press, 2018.

Yuhki Hosoya. On the fragility of the basis on the hamilton–jacobi–bellman equation in economic dynamics. *Journal of Mathematical Economics*, 111:102940, 2024. ISSN 0304-4068. DOI: https://doi.org/10.1016/j.jmateco.2024.102940.

Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. *Advances in neural information processing systems*, 30, 2017.

J Zico Kolter and Gaurav Manek. Learning stable deep dynamics models. *Advances in neural information processing systems*, 32, 2019.

P Lancaster. *Algebraic Riccati Equations*. Oxford Science Publications/The Clarendon Press, Oxford University Press, 1995.

TP Lillicrap. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Michael Lutter. Continuous-time fitted value iteration for robust policies. In *Inductive Biases in Machine Learning for Robotics and Control*, pp. 71–111. Springer, 2023.

Michael Lutter, Christian Ritter, and Jan Peters. Deep lagrangian networks: Using physics as model prior for deep learning. *arXiv preprint arXiv:1907.04490*, 2019.

Michael Lutter, Boris Belousov, Kim Listmann, Debora Clever, and Jan Peters. Hjb optimal feedback control with deep differential value functions and action constraints. In *Conference on Robot Learning*, pp. 640–650. PMLR, 2020.

Michael Lutter, Shie Mannor, Jan Peters, Dieter Fox, and Animesh Garg. Robust value iteration for continuous control tasks. 2021a.

Michael Lutter, Shie Mannor, Jan Peters, Dieter Fox, and Animesh Garg. Value iteration in continuous actions, states and time. In *International Conference on Machine Learning (ICML)*, 2021b.

Michael Lutter, Boris Belousov, Shie Mannor, Dieter Fox, Animesh Garg, and Jan Peters. Continuous-time fitted value iteration for robust policies. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI*, 2023.

Aleksandr Mikhailovich Lyapunov. The general problem of the stability of motion. *International journal of control*, 55(3):531–534, 1992.

Arkadiusz Misztela. On nonuniqueness of solutions of hamilton–jacobi–bellman equations. *Applied Mathematics & Optimization*, 77:599–611, 2018.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Lekan Molu, Nicholas Gans, and Tyler Summers. Minimax iterative dynamic game: Application to nonlinear robot control tasks. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6919–6925. IEEE, 2018.

Tenavi Nakamura-Zimmerer, Qi Gong, and Wei Kang. Adaptive deep learning for high-dimensional hamilton–jacobi–bellman equations. *SIAM Journal on Scientific Computing*, 43(2):A1221–A1247, 2021.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1889–1897, Lille, France, 07–09 Jul 2015. PMLR.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Alena Shilova, Thomas Delliaux, Philippe Preux, and Bruno Raffin. *Learning HJB Viscosity Solutions with PINNs for Continuous-Time Reinforcement Learning*. PhD thesis, Inria Lille-Nord Europe, CRIStAL-Centre de Recherche en Informatique, Signal . . . , 2024.

Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3:9–44, 1988.

Richard S Sutton. Reinforcement learning: An introduction. *A Bradford Book*, 2018.

Yuval Tassa and Tom Erez. Least squares solutions of the hjb equation with neural network value-function approximators. *IEEE transactions on neural networks*, 18(4):1031–1041, 2007.

Russ Tedrake. *Underactuated Robotics*. 2023. URL https://underactuated.csail.mit.edu.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8:279–292, 1992.

Jan Willems. Least squares stationary optimal control and the algebraic riccati equation. *IEEE Transactions on automatic control*, 16(6):621–634, 1971.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

Jie Xu, Viktor Makoviychuk, Yashraj Narang, Fabio Ramos, Wojciech Matusik, Animesh Garg, and Miles Macklin. Accelerated policy learning with parallel differentiable simulation. In *International Conference on Learning Representations*, 2021.

Zetong Xuan, Alper Bozkurt, Miroslav Pajic, and Yu Wang. On the uniqueness of solution for the bellman equation of ltl objectives. In *Proceedings of the 6th Annual Learning for Dynamics & Control Conference*. PMLR, 2024.

Supplementary Materials

The following content was not necessarily subject to peer review.

A Additional Proof

Proof of Lemma 3

Proof. We proof Lemma 3 via contradiction.

Assume $\lambda + \bar{\lambda} = 0$ are pure imaginary numbers or zeros and

$$H \begin{bmatrix} x \\ y \end{bmatrix} = \lambda \begin{bmatrix} x \\ y \end{bmatrix}, \tag{46}$$

where x and y are not both zero. We use upper bar to denote the conjugate of the vector.

We have

$$Ax - BR^{-1}B^\top y = \lambda x \tag{47}$$

$$-Qx - A^\top y = \lambda y, \tag{48}$$

By replacing the second equation into first and multiplying both side with $\bar{y}$ yields

$$\bar{y}BR^{-1}B^\top y = -(\bar{x}Q + \bar{\lambda}\bar{y})x - \lambda\bar{y}x = -\bar{x}Qx. \tag{49}$$

Since $BR^{-1}B^\top \succeq 0$ and $Q \succeq 0$, we conclude Qx and $BR^{-1}B^\top y = 0$. Thus

$$\begin{bmatrix} A - \lambda I \\ Q \end{bmatrix} x = 0. \tag{50}$$

If $x \neq 0$, then the above equation contradict the observability of (Q, A) by Popov-Belevitch-Hautus test (Hespanha, 2018). Similarly, if we have $y \neq 0$, we have

$$\bar{x}[BR^{-1}B^\top, A + \bar{\lambda}I] = 0, \tag{51}$$

which contradict the controllability of (A, B). □

Extension of Corollary 6 to the Discounted Setting:

Theorem 11. *For the discounted LQR problem (18), **at most** one solution to the HJB equation (20) results in a stable closed-loop system.*

Proof. We prove Theorem 11 by relating the eigenvalues of the closed-loop system matrix $A_{cl} = A - BR^{-1}BP$ to its undiscounted counterpart (19).

Suppose P is a solution to the discounted LQR problem (18). By Theorem 1, it is also a solution to its undiscounted counterpart (19). Then, we have a following relationship between the closed-loop system matrix A_{cl} and its undiscounted counterpart $\hat{A}_{cl}$:

$$A_{cl} = A - BR^{-1}B^\top P = A - \frac{1}{2\tau}I - BR^{-1}B^\top P + \frac{1}{2\tau}I = \hat{A}_{cl} + \frac{1}{2\tau}I. \tag{52}$$

Therefore, $A_{cl} \succ \hat{A}_{cl}$.

By Corollary 6 of undiscounted version, there exists at most one P that makes $\hat{A}_{cl} \prec 0$. Therefore, at most one P makes $A_{cl} \prec 0$. □

Note, the stable solution only exists when τ is sufficient large. In other words, if the future cost is discounted too quickly, the policy becomes myopic and fails to provide any guarantees for long-term behavior.

Proof of Theorem 7

Proof. Since $\begin{bmatrix} P_1 \\ P_2 \end{bmatrix}$ are invariant subspace of H, following the proof of Theorem 2, we have

$$\begin{bmatrix} A + BR^{-1}B^\top A^{-\top}Q & -BR^{-1}B^\top A^{-\top} \\ -A^{-\top}Q & A^{-\top} \end{bmatrix} \begin{bmatrix} I \\ P_2P_1^{-1} \end{bmatrix} = \begin{bmatrix} \hat{T} \\ P_2P_1^{-1}\hat{T} \end{bmatrix}, \tag{53}$$

which gives two set of matrix equations:

$$\hat{T} = A + BR^{-1}B^\top A^{-\top}Q - BR^{-1}B^\top A^{-T}P_2P_1^{-1}, \tag{54}$$

$$P_2P_1^{-1}\hat{T} = -A^{-\top}Q + A^{-\top}P_2P_1^{-1}. \tag{55}$$

Substitute the first set of equations into the second gives

$$P_2P_1^{-1}(A + BR^{-1}B^\top A^{-\top}Q - BR^{-1}PB^\top A^{-\top}P_2P_1^{-1}) = -A^{-\top}Q + A^{-\top}P_2P_1^{-1}. \tag{56}$$

Rewriting equation (56) and substituting $P_2P_1^{-1} = P$, we have

$$PA + (I + PBR^{-1}B^\top)A^{-\top}(Q - P) = 0. \tag{57}$$

Next, we are going to show $I + PBR^{-1}B^\top$ is invertible.

Suppose there exists non-zero row vector y, such that $y(I + PBR^{-1}B^\top) = 0$. Premultiplying (57) by y yields $yX = 0$. However, then $y = -y(XBR^{-1}B^\top) = 0$, which contradict the assumption $y \neq 0$. So $I + PBR^{-1}B^\top$ is invertible. Thus, we have

$$A^{-\top}(Q - P) = -(I + PBR^{-1}B^\top)^{-1}PA. \tag{58}$$

Multiplying (57) by $A^\top$, we have

$$A^\top PA + A^\top PBR^{-1}B^\top A^{-\top}(Q - P) + Q - P = 0 \tag{59}$$

$$Q + A^\top PA - A^\top PBR^{-1}B^\top(I + PBR^{-1}B^\top)^{-1}PA = P \tag{60}$$

Noticing

$$(R + B^\top PB)(R^{-1}B^\top)(I + PBR^{-1}B^\top)^{-1} = (B^\top + B^\top PBR^{-1}B)(I + PBR^{-1}B^\top)^{-1} = B^\top, \tag{61}$$

we have

$$(R^{-1}B^\top)(I + PBR^{-1}B^\top)^{-1} = (R + B^\top PB)^{-1}B^\top. \tag{62}$$

Substituting (62) back to (60) yields

$$P = Q + A^\top PA - A^\top PB(R + B^\top PB)^{-1}B^\top PA, \tag{63}$$

which completes the proof. □

Proof of Theorem 8 We first introduce the discrete-time counterpart of Lemma 3

Lemma 12. *If the pair (A, B) is controllable and the pair (Q, A) is observable, the Hamiltonian matrix H (36) has no eigenvalue such that $|\lambda| = 1$.*

Proof. Let

$$H \begin{bmatrix} x \\ y \end{bmatrix} = \lambda \begin{bmatrix} x \\ y \end{bmatrix}, \tag{64}$$

and $|\lambda| = 1$. Then,

$$Ax + BR^{-1}B^\top A^{-\top}Qx - BR^{-1}B = \lambda x \tag{65}$$
$$-A^{-\top}Qx + A^{-\top}y = \lambda y. \tag{66}$$

Multiplying the second equation by $BR^{-1}B^\top$ and adding it to the first give

$$Ax - \lambda BR^{-1}By = \lambda x \tag{67}$$
$$-Qx + y = \lambda A^\top y. \tag{68}$$

Multiplying the first by $\bar{\lambda}\bar{y}$ and second by $\bar{x}$ yields

$$\bar{\lambda}\bar{y}Ax = \bar{\lambda}\lambda\bar{y}BR^{-1}By + \bar{\lambda}\lambda\bar{y}x = -\bar{x}Qx - \bar{y}x. \tag{69}$$

Thus

$$-\bar{\lambda}\lambda\bar{y}BR^{-1}B^\top y - \bar{x}Qx = (\bar{y}y - 1)\bar{y}x = 0. \tag{70}$$

Therefore,

$$y^\top B = 0 \tag{71}$$
$$Qx = 0 \tag{72}$$

The equation $Cx = 0$ implies $y^T A = \lambda^{-1} y^T$, which, together with $y^T B = 0$, identifies λ^{-1} as an uncontrollable eigenvalue of the pair (A, B). On the other hand, $y^T B = 0$ implies $Ax = \lambda x$, and combined with $Cx = 0$, this characterizes λ as an unobservable eigenvalue of the pair (C, A). □

We now proof the whole Theorem 8:

Proof. Let $J = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}$, we can easily verify $H^\top JH = J$.

So $H\mathbf{x} = \lambda\mathbf{x}$ implies

$$JH\mathbf{x} = \lambda J\mathbf{x} \tag{73}$$
$$JH\mathbf{x} = \lambda H^\top (JH\mathbf{x}) \tag{74}$$
$$\frac{1}{\lambda}\mathbf{v} = H^\top \mathbf{v}, \tag{75}$$

where $\mathbf{v} = JH\mathbf{x}$. Thus, $\frac{1}{\lambda}$ is also a eigenvalue of H.

Given Lemma 12, there must be n eigenvalues within the unit cycle and n outside. □

Proof of Theorem 9

Proof. By invariant subspace, we have

$$\begin{bmatrix} A + BR^{-1}B^\top Q & -BR^{-1}B^\top A^{-\top} \\ -A^{-\top}Q & A^{-\top} \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} = \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} \Lambda_1, \tag{76}$$

which yields to two matrix equations:

$$AP_1 + BR^{-1}B^\top QP_1 - BRB^\top A^{-\top}P_2 = P_1\Lambda_1 \tag{77}$$
$$-A^{-\top}QP_1 + A^{-\top}P_2 = P_2\Lambda_1. \tag{78}$$

Substituting $P_2 = PP_1$ into the second equation

$$A^{-\top}(P - Q)P_1 = PP_1\Lambda_2. \tag{79}$$

Noticing

$$PA_{cl} = PA - PB(R + B^\top PB)^{-1}B^\top PA = A^{-\top}(P - Q). \tag{80}$$

So,

$$PA_{cl}P_1 = A^{-\top}(P - Q)P_1 = PP_1\Lambda_2. \tag{81}$$

Substituting $P = P_2P_1^{-1}$ into (81), yields

$$P_2P_1^{-1}A_{cl}P_1 = P_2\Lambda_1. \tag{82}$$

Therefore, the diagonal entries of Λ_1 are eigenvalues of $P_1^{-1}A_{cl}P_1$. Since similarity transformations preserve the eigenvalues of a matrix, A_{cl} also have same set of eigenvalues Λ_1 □

Proof of Theorem 10 Here, we introduce the Lyapunov Theorem and use it to finalize the proof.

Lemma 13. ***(Lyapunov Theorem)*** *Consider a dynamical system described by the differential equation:*

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t)), \quad \mathbf{x}(t) \in \mathbb{R}^n, \tag{83}$$

where $f : \mathbb{R}^n \to \mathbb{R}^n$ is continuously differentiable.

Let $\mathcal{V}(\mathbf{x}(t))$ be a continuously differentiable scalar function, called a Lyapunov function, with the following properties: $\mathcal{V}(\mathbf{x}(t)) > 0$ for $\mathbf{x} \neq \mathbf{x}_{eq}$ and $\mathcal{V}(\mathbf{x}_{eq}) = 0$.

Then, if $\dot{\mathcal{V}}(\mathbf{x}(t)) \leq 0$ for all $\mathbf{x}(t) \neq \mathbf{x}_{eq}$, then the equilibrium point at $\mathbf{x}_{eq}$ is stable, where the derivative of $\mathcal{V}(\mathbf{x}(t))$ along the trajectories of the system is given by:

$$\dot{\mathcal{V}}(\mathbf{x}(t)) = \frac{\partial \mathcal{V}(\mathbf{x}(t))^\top}{\partial \mathbf{x}} f(\mathbf{x}(t)). \tag{84}$$

Furthermore, if $\dot{\mathcal{V}}(x) < 0$ for all $\mathbf{x} \neq \mathbf{x}_{eq}$, the equilibrium point at $\mathbf{x}_{eq}$ is asymptotically stable and we have $\lim_{s\to\infty} \|\mathbf{x}(s) - \mathbf{x}_{eq}\| = 0$ for any initial condition $\mathbf{x}(t)$.

Proof. See any control textbook or the English translation of Lyapunov's original thesis (Lyapunov, 1992). □

We now prove Theorem 10 by showing the solution to Bellman equation with positive definite model is a Lyapunov function.

Proof. Since $\mathcal{V}(\mathbf{x}(t))$ is the solution to HJB equation, with the policy $\mathbf{u}^*(t) = \pi^*(\mathbf{x}(t))$, we have

$$\dot{\mathcal{V}}(\mathbf{x}(t)) = \frac{\partial \mathcal{V}(\mathbf{x})^\top}{\partial \mathbf{x}} f(\mathbf{x}(t), \pi^*(\mathbf{x}(t))) = -l(\mathbf{x}(t), \pi^*(\mathbf{x}(t))) \leq 0. \tag{85}$$

Additionally, we have $\mathcal{V}(\mathbf{x}) > 0$ for all $\mathbf{x} \neq \mathbf{x}_{\text{eq}}$ and $\mathcal{V}(\mathbf{x}_{\text{eq}}) = 0$. Therefore $\mathcal{V}(\mathbf{x}(t))$ is also a Lyapunov function for closed-loop system $\dot{\mathbf{x}} = f(\mathbf{x}, \pi^*(\mathbf{x}))$. By Lemma (13), the closed-system is stable. Furthermore, if we have $l(\mathbf{x}(t), \pi^*(\mathbf{x}(t))) > 0$ for all $\mathbf{x} \neq \mathbf{x}_{\text{eq}}$, the closed-loop is asymptotically stable. □

Remark 1. *We assume no discount on the running cost in the proof of Theorem 10. This assumption allows the policy to equally weigh all future consequences of the current control, which is crucial for proving the long-term behavior of the system such as stability. Additionally, the value function $\mathcal{V}(\mathbf{x}) < \infty$ is well-defined, given the presence of equilibrium points.*

However, Theorem 10 may not hold if the discount factor $e^{-\frac{s}{\tau}} \ll 1$ is small, due to myopic policy behavior. In such cases, local stability may still be proven under additional assumptions on the running cost and system dynamics.

B Bellman Equation in Tabular Setting

Here, we include the proof showing that the value function is the unique solution to the Bellman equation in the tabular setting for comparison. The proof is mainly adapted from Blackwell (1965) and the lecture note from Gangwani (2019).

We assume both $\mathcal{X}$ and $\mathcal{U}$ is finite sets with size $|\mathcal{X}|$ and $|\mathcal{U}|$, respectively.

For simplicity, we reframe the problem from minimizing the cost-to-go to maximizing the reward-to-go, where the reward is defined as $r(\mathbf{x}, \mathbf{u}) = -l(\mathbf{x}, \mathbf{u})$.

We define Bellman operator $\mathcal{T} : \mathbb{R}^{|\mathcal{X}|} \to \mathbb{R}^{|\mathcal{X}|}$ for any $\mathcal{W} \in \mathbb{R}^{|\mathcal{X}|}$ as

$$(\mathcal{T}\mathcal{W})(\mathbf{x}) = \max_{\mathbf{u}} [r(\mathbf{x}, \mathbf{u}) + \gamma \mathcal{W}(f(\mathbf{x}, \mathbf{u}))]. \tag{86}$$

Theorem 14. *Bellman operator $\mathcal{T}$ is a contraction mapping under sup-norm $\|\cdot\|_\infty$, i.e.,*

$$\|\mathcal{T}\mathcal{W} - \mathcal{T}\mathcal{Z}\|_\infty \leq \gamma \|\mathcal{W} - \mathcal{Z}\|_\infty, \forall \mathcal{W}, \mathcal{Z} \in \mathbb{R}^{|\mathcal{X}|}. \tag{87}$$

Proof. For any particular state $\mathbf{x} \in \mathcal{X}$, we have

$$|(\mathcal{T}\mathcal{W})(\mathbf{x}) - (\mathcal{T}\mathcal{Z})(\mathbf{x})| = |\max_{\mathbf{u}} [r(\mathbf{x}, \mathbf{u}) + \gamma \mathcal{W}(f(\mathbf{x}, \mathbf{u}))] - \max_{\mathbf{u}} [r(\mathbf{x}, \mathbf{u}) + \gamma \mathcal{Z}(f(\mathbf{x}, \mathbf{u}))]| \tag{88}$$

$$\leq \gamma \max_{\mathbf{u}} |\mathcal{W}(f(\mathbf{x}, \mathbf{u})) - \mathcal{Z}(f(\mathbf{x}, \mathbf{u}))| \tag{89}$$

$$\leq \gamma \|\mathcal{W} - \mathcal{Z}\|_\infty. \tag{90}$$

Since, the above holds for any state $\mathbf{x}$, we can conclude

$$\|\mathcal{T}\mathcal{W} - \mathcal{T}\mathcal{Z}\|_\infty = \max_{\mathbf{x}} |(\mathcal{T}\mathcal{W})(\mathbf{x}) - (\mathcal{T}\mathcal{Z})(\mathbf{x})| \leq \gamma \|\mathcal{W} - \mathcal{Z}\|_\infty. \tag{91}$$

□

We can easily verified value function $\mathcal{V}$ is a fixed point to Bellman operator. Furthermore, value iteration defined as

$$\mathcal{W}_{k+1} = \mathcal{T}\mathcal{W}_k, \tag{92}$$

converge to the value function $\mathcal{V}$.

Theorem 15. *Value iteration (92) converges to value function $\mathcal{V}$, i.e.,*

$$\lim_{k \to \infty} \mathcal{W}_k = \mathcal{V}, \tag{93}$$

where $\mathcal{W}_k = \mathcal{T}^{k-1}\mathcal{W}_0$

Proof. Since $\mathcal{V}$ is a fixed point of $\mathcal{T}$ and according to Theorem 87, we have

$$\|\mathcal{W}_k - \mathcal{V}\|_\infty = \|\mathcal{T}\mathcal{W}_{k-1} - \mathcal{T}\mathcal{V}\|_\infty \leq \gamma \|\mathcal{W}_{k-1} - \mathcal{V}\|_\infty \leq \cdots \leq \gamma^k \|\mathcal{W}_0 - \mathcal{V}\|_\infty. \tag{94}$$

Let $k \to \infty$ completes the proof. □

Corollary 16. *Value function $\mathcal{V}$ is the unique solution to Bellman equation.*

Proof. Suppose there are other $\mathcal{V}'$ that also satisfying Bellman equation. By Theorem 15, we have

$$\|\mathcal{T}^k \mathcal{V}' - \mathcal{V}\|_\infty \leq \gamma^k \|\mathcal{V}' - \mathcal{V}\|_\infty. \tag{95}$$

However, since $\mathcal{V}'$ also satisfying Bellman equation, we have

$$\|\mathcal{T}^k \mathcal{V}' - \mathcal{V}\|_\infty = \|\mathcal{V}' - \mathcal{V}\|_\infty, \tag{96}$$

which contradicts the equation (95). □

C Effect of Initialization

Despite the existence of infinitely many solutions for the LQR problem in Section 4, we observe that the neural network tends to favor certain solutions over others. For example, when using self-normalized initialization methods such as LeCun normal or Kaiming normal (Klambauer et al., 2017), the network tends to converge to an unstable negative definite solution. We hypothesize that this may be due to the negative definite solution having the smallest eigenvalues among the possible solutions, which are closer to the initial weights generated by these self-normalized methods.

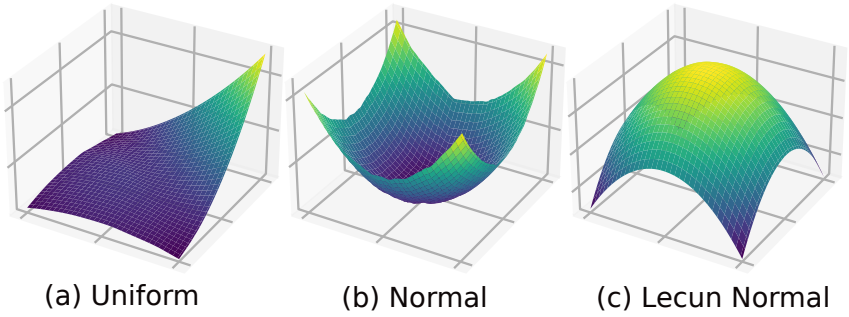

Figure 5: Solutions found with different initialization method

We further tested different initialization methods, including uniform initialization within ± 0.5 and initialization using a normal distribution with a standard deviation of 1. Figure 5 shows the solutions found by the neural network with different initialization methods.

D Experiment Setup

D.1 Tasks

Cartpole Cartpole is a classical control task for reinforcement learning. In this task, we consider bring the cart to the origin while maintain the posture of pole to upright position. The state is denote as $[p, \theta, v, \omega]^\top$, where p denotes the position, θ is the angle measured from the upright position, and v and ω denotes velocity and angular velocity, respectively. The dynamics is given by

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} v \\ \omega \\ \frac{u + m_p \sin\theta (l\dot{\theta}^2 - g\cos\theta)}{m_c + m_p \sin^2\theta} \\ \frac{u\cos\theta + m_p l\dot{\theta}^2 \cos\theta \sin\theta - (m_c + m_p) g \sin\theta}{l(m_c + m_p \sin^2\theta)} \end{bmatrix}, \tag{97}$$

where $m_c = 1kg$, $m_p = 0.1kg$ are mass of cart and pole respectively, and $l = 1m$ is the length of rod, $g = 9.81m/s^2$ is the gravity constant.

Each time, the cartpole is initialized uniformly within the $\pm[4, 0.5, 2, 2]^\top$ and will reset the environment when the state goes beyond $\pm[10, 10, 1000, 1000]$ The control input is the force applied to cartpole, with maximum value equal to the weight of the pole. The running cost is defined as

$$l(\mathbf{x}(t), \mathbf{u}(t)) = \|\mathbf{x}(t)\|_2^2 + \|\mathbf{u}(t)\|_2^2 \tag{98}$$

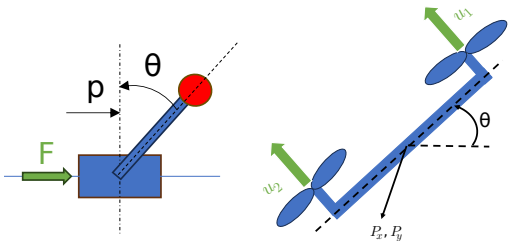

Figure 6: System Diagram

Drone The 2D drone is a canonical underactuated system in the control community. The state is define as $[p_x, p_y, \theta, v_x, v_y, \omega]$, where p_x, p_y represent the position in the XY plane, θ denotes the orientation, v_x, v_y are linear velocity, and ω is the angular velocity. The dynamics is given by

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} v_x \\ v_y \\ \omega \\ -\frac{(u_1+u_2)\sin\theta}{m} \\ \frac{(u_1+u_2)\cos\theta}{m} - g \\ \frac{r(u_1-u_2)}{I} \end{bmatrix}, \tag{99}$$

where $m = 1kg$ is the mass, $r = 0.25m$ is the length of the wing, $I = 0.0625kg \cdot m^2$ is the inertia and $g = 9.81m/s^2$ is gravity constant.

The task is to control the drone to reach a desired position and maintain hovering. Particularly, the drone is initialized uniformly within $\pm[2, 2, 2, 2, 2, 2]^\top$ around the target state and is reset if the state exceeds $\pm[4, 4, 4, 5, 5, 5]$ from the target. The control $[u_1, u_2]^\top$ represents the thrust generated by the propellers, with a maximum value of twice the total weight of the drone. The running cost is defined as

$$l(\mathbf{x}(t), \mathbf{u}(t)) = \|\mathbf{x}(t)\|_2^2 + \|\mathbf{u}(t)\|_2^2. \tag{100}$$

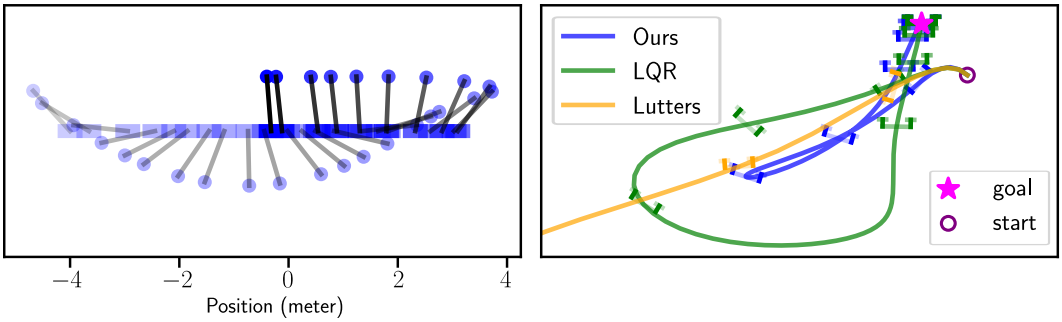

Figure 7: Example trajectories generated using our method. Left: The cartpole starts with high angular and linear velocity, and our method successfully stabilizes it to the origin. Right: Drone tracking problem. We compare trajectories with Lutter et al. (2020) and a linearized LQR controller. Our method finds the optimal trajectory by effectively leveraging the nonlinear dynamics.

D.2 Algorithm

We minimize the following weighted TD loss:

$$\mathcal{L}_\theta = \frac{1}{N}\sum_{i=1}^{N} \left|1 + \frac{\frac{\partial \hat{\mathcal{V}}_\theta(\mathbf{x}^i)}{\partial \mathbf{x}}^\top f(\mathbf{x}^i, \mathbf{u}^i) - \frac{1}{\tau}\hat{\mathcal{V}}_\theta(\mathbf{x}^i)}{l(\mathbf{x}^i, \mathbf{u}^i)}\right| \tag{101}$$

where N is the number of total states collected via Monte Carlo sampling. Compared to the original TD error (10), the loss (101) weights the error based on the running cost, providing a well-scaled measure that keeps the total loss within a reasonable range.

In both tasks, we control an affine system with box-constrained control limits. Therefore, we compute the control $\mathbf{u}^i$ analytically:

$$\mathbf{u}^i = \text{clip}(-\frac{1}{2}R^{-1}f_2^\top(\mathbf{x}^i)\frac{\partial \mathcal{V}(\mathbf{x}^i)}{\partial \mathbf{x}}, \mathbf{u}_{\min}, \mathbf{u}_{\max}), \tag{102}$$

where $f(\mathbf{x}^i)$ is the control Jacobian matrix from (38).

The Algorithm 1 summarize the main algorithm.

Algorithm 1 HJB Equation Learning

Initialize the parameters θ for $\mathcal{V}_\theta$, empty dataset $\mathcal{D} = \{\}$
for $i = 1 : n_{\text{epoch}}$ **do**
  Rollout n_{rollout} trajectories and append states visited to $\mathcal{D}$
  for batch of $\mathbf{x}^i$ in $\mathcal{D}$ **do**
    Compute the optimal control $\mathbf{u}^i$ based on (102)
    Compute the weighed TD error $\mathcal{L}_\theta$ based on (101)
    Update the value function: $\theta \leftarrow \theta - \alpha \frac{d\mathcal{L}_\theta}{d\theta}$
  end for
end for

D.3 Hyperparameters

We use the same hyperparameters for both tasks and summarized them in Table 2.

Table 2: Hyperparameters

Parameter names	**Value**
Network size	[128, 128, 64]
Learning rate	10^{-3}
Batch size	256
Number of rollout	20
Maximum trajectory length	200
ϵ	10^{-3}
Activation function	ELU
Kernel initialization	Lecun normal
Optimizer	Adam