Hamilton–Jacobi Deep Q-Learning for Deterministic Continuous-Time Systems with Lipschitz Continuous Controls*

Jeongho Kim† Jaeuk Shin‡ Insoon Yang‡

Abstract

In this paper, we propose Q-learning algorithms for continuous-time deterministic optimal control problems with Lipschitz continuous controls. Our method is based on a new class of Hamilton–Jacobi–Bellman (HJB) equations derived from applying the dynamic programming principle to continuous-time Q-functions. A novel semi-discrete version of the HJB equation is proposed to design a Q-learning algorithm that uses data collected in discrete time without discretizing or approximating the system dynamics. We identify the condition under which the Q-function estimated by this algorithm converges to the optimal Q-function. For practical implementation, we propose the *Hamilton–Jacobi DQN*, which extends the idea of deep Q-networks (DQN) to our continuous control setting. This approach does not require actor networks or numerical solutions to optimization problems for greedy actions since the HJB equation provides a simple characterization of optimal controls via ordinary differential equations. We empirically demonstrate the performance of our method through benchmark tasks and high-dimensional linear-quadratic problems.

1 Introduction

Model-free reinforcement learning (RL) algorithms provide an effective data-driven solution to sequential decision-making problems, in particular, in the discrete-time setting [1–3]. Recently, there has been a growing interest in and demand for applying these techniques to complex physical control tasks, motivated by robotic and autonomous systems. However, many physical processes evolve in continuous time, requiring the need for RL methods that can systematically handle continuous-time dynamical systems. These systems are often described by deterministic ordinary differential equations (ODEs). Classical approaches first estimate the model parameters by using system identification techniques and then design a suitable model-based controller (e.g., [4]). However, we do not often have such a luxury of having a separate training period for parameter identification, which often requires large-scale high-resolution data. Furthermore, when the model parameters change over time, the classical techniques have fundamental limitations in terms of adaptivity. The focus of this work is to study a control-theoretic model-free RL method that extends the popular Q-learning [5] and deep Q-networks (DQN) [6] to the continuous-time deterministic optimal control setting.

*This work was supported in part by the Creative-Pioneering Researchers Program through SNU, the National Research Foundation of Korea funded by the MSIT(2020R1C1C1009766), the Information and Communications Technology Planning and Evaluation (IITP) grant funded by MSIT(2020-0-00857), and Samsung Electronics.

†Institute of New Media and Communications, Seoul National University, Seoul 08826, South Korea, (jhkim206@snu.ac.kr).

‡Department of Electrical and Computer Engineering, Automation and Systems Research Institute, Seoul National University, Seoul 08826, South Korea, ({sju5379, insoonyang}@snu.ac.kr).

One of the most straightforward ways to tackle such continuous-time control problems is to discretize time, state, and action, and then employ an RL algorithm for discrete Markov decision processes (MDPs). However, this approach could easily be rendered ineffective when a fine discretization is used [7]. To avoid the explicit discretization of state and action, several methods have been proposed using function approximators [8]. Among those, algorithms that use deep neural networks as function approximators provide strong empirical evidence for learning high-performance policies, on a range of benchmark tasks [9–12]. To deal with continuous action spaces, such discrete-time model-free deep RL methods numerically solve optimization problems for greedy actions [13] or use parameterized policies and learn the network parameters via policy gradient [14, 15], actor-critic methods [16–20], or normalized advantage functions [21]. However, in these methods it is unclear how to choose the size of discretized time steps or how the algorithms should be systematically modified to take into account the efficiency and the stability of learning processes according to the characteristics of the continuous-time systems.

The literature regarding continuous-time RL is relatively limited; most of them have tried to avoid explicit discretization using the structural properties of limited classes of system dynamics (for example, see [22–29] for linear or control-affine systems, and see [30] for semi-MDPs with finite state and action spaces). We also refer to [31], where the policy gradient method in continuous-time setting is introduced. However, the reward function does not depend on the control signal in their framework.

In general continuous-time cases, the dynamic programming equation is expressed as a Hamilton–Jacobi–Bellman (HJB) equation that provides a sound theoretical framework. Previous methods use HJB equations for learning the optimal *state-value function* or its gradient via convergent discretization [32], barycentric interpolation [33], advantage functions [34], temporal difference algorithms [7], kernel-based approximations [35], adaptive dynamic programming [36], path integrals [37, 38] and neural network approximation [39, 40].

However, to our knowledge, HJB equations have not been studied for admitting Q-functions as a solution (i.e., state-action value functions) in the previous methods although there have been a few attempts to construct variants of Q-functions for continuous-time dynamical systems. In [41], the Q-function for linear time-invariant systems is defined as the sum of the optimal state-value function and the Hamiltonian. Another variant of Q-functions is introduced as the sum of the running cost and the directional derivative of the state-value function [42], which is then approximated by a parameterized family of functions. However, in our opinion, the definitions of the Q-function in these works are different from the standard state-action value function that is defined as the maximum expected cumulative reward incurred after starting from a particular state with a specific action. Moreover, they have only used HJB equations for the state-value function without introducing or using HJB equations for the constructed Q-functions. The practical performances of these methods have only been demonstrated through low-dimensional tasks. More recently, [43] devises a new method combining advantage updating [44] and existing off-policy RL algorithms to propose continuous-time RL algorithms that are robust to time discretization. However, to tackle problems with continuous action spaces, this method uses off-policy actor-critic methods rather than relying only on the state-value functions.

In this work, we consider continuous-time deterministic optimal control problems with Lipschitz continuous controls in the infinite-horizon discounted setting. We show that the standard Q-function is well defined in continuous-time under Lipschitz constraints on controls. Applying the dynamic programming principle to the Q-function, we derive a novel class of HJB equations. The HJB equation is shown to admit a unique viscosity solution, which corresponds to the optimal Q-function. To the best of our knowledge, this is the first attempt to rigorously characterize the HJB equations for Q-functions in continuous-time control. The HJB equations provide a simple model-free characterization of optimal controls via ODEs and a theoretical basis for our Q-learning method. We propose a new semi-discrete version of the HJB equation to obtain a Q-learning algorithm that uses sample data collected in discrete time without discretizing or approximating the continuous-time dynamics. By design, it attains the flexibility to choose the sampling interval to take into account the features of continuous-time systems, but without the need for sophisticated ODE discretization methods. We provide a convergence analysis that suggests a limit for the sampling interval for the convergence guarantee. This study may open a new exciting avenue of research that connects HJB equations and Q-learning domain.

For a practical implementation of our HJB-based Q-learning, we combine it with the idea of DQN. This new model-free off-policy deep RL algorithm, which we call the *Hamilton-Jacobi DQN* (HJ DQN), is as simple as DQN but capable of solving continuous-time problems without discretizing the system dynamics or the action space. Instead of using any parameterized policy or numerically optimizing the estimated Q-functions to compute greedy actions, HJ DQN benefits from the simple ODE characterization of optimal controls, which are obtained in our theoretical analysis of the HJB equations. Thus, our algorithm is computationally light and easy to implement, thereby requiring less hyperparameter tuning compared to actor-critic methods for continuous control. We evaluate our algorithm on OpenAI benchmark tasks and high-dimensional linear-quadratic (LQ) control problems. The result of our experiments suggests that actor networks in actor-critic methods may be replaced by the optimal control obtained via our HJB equation.

This paper is significantly expanded from a preliminary conference version [45]. A Q-learning algorithm and its DQN variant are newly designed in a principled manner to use transition data collected in discrete time. Furthermore, convergence properties of our Q-learning method are carefully studied in this paper. It contains the results of more thorough numerical experiments on several benchmark tasks and LQ problems, as well as ablation studies.

The remainder of this paper is organized as follows. In Section 2, we define the Q-functions for continuous-time optimal control problems with Lipschitz continuous controls and derive the associated HJB equations. We also characterize optimal control dynamics via an ODE. In Section 3, we propose a Q-learning algorithm based on the semi-discrete HJB equation and identify its convergence properties. In Section 4, we introduce the HJ DQN algorithm and discuss its features. Section 5 provides the results of our experiments on benchmark problems as well as LQ control problems. All the mathematical proofs are contained in Appendix B.

2 Hamilton–Jacobi–Bellman Equations for Q-Functions

Consider a continuous-time dynamical system of the form[1]

$$\dot{x}(t) = f(x(t), a(t)), \quad t > 0, \tag{2.1}$$

where $x(t) \in \mathbb{R}^n$ and $a(t) \in \mathbb{R}^m$ are the system state and the control action, respectively. Here, the vector field $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is an unknown function. The standard infinite-horizon discounted optimal control problem can be formulated as[2]

$$\sup_{a \in \mathcal{A}} J_{\boldsymbol{x}}(a) := \int_0^\infty e^{-\gamma t} r(x(t), a(t))\, \mathrm{d}t, \tag{2.2}$$

[1]Here, $\dot{x}$ denotes $\mathrm{d}x/\mathrm{d}t$.

[2]Although the focus of this work is deterministic control, one may also consider its stochastic counterpart. We briefly discuss the extension of our method to the stochastic control setting in Appendix C.

with $x(0) = \boldsymbol{x}$, where $r : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ is an unknown reward function of interest and $\gamma > 0$ is a discount factor. We follow the convention in continuous-time deterministic optimal control that considers control trajectory, instead of control policy, as the optimization variable [46].

The (continuous-time) Q-function of (2.2) is defined as

$$Q(\boldsymbol{x}, \boldsymbol{a}) := \sup_{a \in \mathcal{A}} \left\{ \int_0^\infty e^{-\gamma t} r(x(t), a(t)) \, \mathrm{d}t \mid x(0) = \boldsymbol{x}, a(0) = \boldsymbol{a} \right\}, \tag{2.3}$$

which represents the maximal reward incurred from time 0 when starting from $x(0) = \boldsymbol{x}$ with $a(0) = \boldsymbol{a}$. Suppose for a moment that the set of admissible controls $\mathcal{A}$ has no particular constraints, i.e., $\mathcal{A} := \{a : \mathbb{R}_{\geq 0} \to \mathbb{R}^m \mid a \text{ measurable}\}$. Then, $Q(\boldsymbol{x}, \boldsymbol{a})$ reduces to the standard optimal value function $v(\boldsymbol{x}) := \sup_{a \in \mathcal{A}} \{ \int_0^\infty e^{-\gamma t} r(x(t), a(t)) \, \mathrm{d}t \mid x(0) = \boldsymbol{x} \}$ for all $\boldsymbol{a} \in \mathbb{R}^m$ since the action can be switched immediately from $\boldsymbol{a}$ to an optimal control and in this case $\boldsymbol{a}$ does not affect the total cost or the system trajectory in the continuous-time setting.

Proposition 1. *Suppose that $\mathcal{A} := \{a : \mathbb{R}_{\geq 0} \to \mathbb{R}^m \mid a \text{ measurable}\}$. Then, the optimal Q-function (2.3) corresponds to the optimal value function v for each $\boldsymbol{a} \in \mathbb{R}^m$, i.e., $Q(\boldsymbol{x}, \boldsymbol{a}, t) = v(\boldsymbol{x}, t)$ for all $(\boldsymbol{x}, \boldsymbol{a}, t) \in \mathbb{R}^n \times \mathbb{R}^m \times [0, T]$.*

Thus, if $\mathcal{A}$ is chosen as above, the Q-function has no additional interesting property under the standard choice of $\mathcal{A}$.[3] Motivated by the observation, we restrict the control $a(t)$ to be a Lipschitz continuous function in t. Since any Lipschitz continuous function is differentiable almost everywhere, we choose the set of admissible controls as

$$\mathcal{A} := \{a : \mathbb{R}_{\geq 0} \to \mathbb{R}^m \mid a \text{ measurable}, \ |\dot{a}(t)| \leq L \text{ a.e.}\},$$

where $|\cdot|$ denotes the standard Euclidean norm, and L is a fixed constant. From now on, we will focus on the optimal control problem (2.2) with Lipschitz continuous controls, i.e., $|\dot{a}(t)| \leq L$ a.e., and the corresponding Q-function (2.3).

Our first step is to study the structural properties of the optimality equation and the optimal control via dynamic programming. Using the discovered structural properties, a DQN-like algorithm is then designed to solve the optimal control problem (2.2) in a model-free manner.

2.1 Dynamic Programming and HJB Equations

By the dynamic programming principle, we have

$$Q(\boldsymbol{x}, \boldsymbol{a}) = \sup_{a \in \mathcal{A}} \left\{ \int_t^{t+h} e^{-\gamma(s-t)} r(x(s), a(s)) \, \mathrm{d}s + e^{-\gamma h} Q(x(t+h), a(t+h)) \mid x(t) = \boldsymbol{x}, a(t) = \boldsymbol{a} \right\}$$

for any $h > 0$. Rearranging this equality, we obtain

$$\begin{aligned} 0 = \sup_{a \in \mathcal{A}} \bigg\{ \frac{1}{h} \int_t^{t+h} e^{-\gamma(s-t)} r(x(s), a(s)) \, \mathrm{d}s + \frac{1}{h} [Q(x(t+h), a(t+h)) - Q(\boldsymbol{x}, \boldsymbol{a})] \\ + \frac{e^{-\gamma h} - 1}{h} Q(x(t+h), a(t+h)) \mid x(t) = \boldsymbol{x}, a(t) = \boldsymbol{a} \bigg\}. \end{aligned}$$

Letting h tend to zero and assuming for a moment that the Q-function is continuously differentiable, its Taylor expansion yields

$$\gamma Q(\boldsymbol{x}, \boldsymbol{a}) - \nabla_{\boldsymbol{x}} Q \cdot f(\boldsymbol{x}, \boldsymbol{a}) - \sup_{\boldsymbol{b} \in \mathbb{R}^m, |\boldsymbol{b}| \leq L} \nabla_{\boldsymbol{a}} Q \cdot \boldsymbol{b} - r(\boldsymbol{x}, \boldsymbol{a}) = 0,$$

[3]This observation is consistent with the previously reported result on the continuous limit of Q-functions [43, 44].

where the optimization variable $\boldsymbol{b}$ represents $\dot{a}(t)$. Note that the supremum is attained at $\boldsymbol{b}^\star = L\frac{\nabla_{\boldsymbol{a}} Q}{|\nabla_{\boldsymbol{a}} Q|}$. Thus, we obtain

$$\gamma Q(\boldsymbol{x}, \boldsymbol{a}) - \nabla_{\boldsymbol{x}} Q \cdot f(\boldsymbol{x}, \boldsymbol{a}) - L|\nabla_{\boldsymbol{a}} Q| - r(\boldsymbol{x}, \boldsymbol{a}) = 0, \tag{2.4}$$

which is the *HJB equation for the Q-function.* However, the Q-function is not continuously differentiable in general. This motivates us to consider a weak solution of the HJB equation. Among several types of weak solutions, it is shown in Appendix A that the Q-function corresponds to the unique *viscosity solution* [47] of the HJB equation under the following assumption:

Assumption 1. *The functions f and r are bounded and Lipschitz continuous, i.e., there exists a constant C such that $\|f\|_{L^\infty} + \|r\|_{L^\infty} < C$ and $\|f\|_{\mathrm{Lip}} + \|r\|_{\mathrm{Lip}} < C$, where $\|\cdot\|_{\mathrm{Lip}}$ denotes a Lipschitz constant of argument.*

2.2 Optimal Controls

In the derivation of the HJB equation above, we deduce that an optimal control a must satisfies $\dot{a} = L\frac{\nabla_{\boldsymbol{a}} Q}{|\nabla_{\boldsymbol{a}} Q|}$ when Q is differentiable. The viscosity solution framework [46] can be used to obtain the following more rigorous characterization of optimal controls when the Q-function is not differentiable.

Theorem 1. *Suppose that Assumption 1 holds. Consider a control trajectory $a^\star(s)$, $s \geq t$, defined by*

$$\dot{a}^\star(s) = L\frac{p_2}{|p_2|} \quad \forall p = (p_1, p_2) \in D^\pm Q(x^\star(s), a^\star(s)) \tag{2.5}$$

for a.e. $s \geq t$, and $a^\star(t) = \boldsymbol{a}$, where $\dot{x}^\star = f(x^\star, a^\star)$ for $s \geq t$ and $x^\star(t) = \boldsymbol{x}$. Assume that the function Q is locally Lipschitz in a neighborhood of $(x^\star(s), a^\star(s))$ and that $D^+Q(x^\star(s), a^\star(s)) = \partial Q(x^\star(s), a^\star(s))$ for a.e. $s \geq t$.[4] *Then, $a^\star$ is optimal among those in $\mathcal{A}$ such that $a(t) = \boldsymbol{a}$, i.e., it satisfies*

$$a^\star \in \arg\max_{a \in \mathcal{A}} \left\{ \int_t^\infty e^{-\gamma(s-t)} r(x(s), a(s)) \, \mathrm{d}s, \;\middle|\; x(t) = \boldsymbol{x}, a(t) = \boldsymbol{a} \right\}. \tag{2.6}$$

If, in addition,

$$\boldsymbol{a} \in \arg\max_{\boldsymbol{a}' \in \mathbb{R}^m} Q(\boldsymbol{x}, \boldsymbol{a}'),$$

then $a^\star$ is an optimal control, i.e., it satisfies

$$a^\star \in \arg\max_{a \in \mathcal{A}} \left\{ \int_t^\infty e^{-\gamma(s-t)} r(x(s), a(s)) \, \mathrm{d}s \;\middle|\; x(t) = \boldsymbol{x} \right\}.$$

Note that at a point $(\boldsymbol{x}, \boldsymbol{a})$ where Q is differentiable, the ODE (2.5) is simplified to $\dot{a}^\star = L\frac{\nabla_{\boldsymbol{a}} Q(x^\star, a^\star)}{|\nabla_{\boldsymbol{a}} Q(x^\star, a^\star)|}$. A useful implication of this theorem is that for any $\boldsymbol{a} \in \mathbb{R}^m$, an optimal control in $\mathcal{A}$ such that $a(t) = \boldsymbol{a}$ can be obtained using the ODE (2.5) with the initial condition $a^\star(t) = \boldsymbol{a}$. Thus, when the control is initialized as an arbitrary value $\boldsymbol{a}$ at arbitrary time t in Q-learning, we can still use the ODE (2.5) to obtain an optimal control. Another important implication of Theorem 1 is that an optimal control can be constructed without numerically solving any optimization problem. This salient feature assists in the design of a computationally efficient DQN algorithm for continuous control without involving any explicit optimization nor any actor network.

[4]Here, D^+Q and D^-Q denote the super- and sub-differentials of Q, respectively, and $D^\pm Q := D^+Q \cup D^-Q$. At a point $(\boldsymbol{x}, \boldsymbol{a})$ where Q is differentiable, the super- and sub-differentials are identical to the singleton of the classical derivative of Q. Moreover, ∂Q denotes the Clarke's generalized gradient of Q (see, e.g., p. 63 of [46]). Note that the right-hand side of ODE (2.5) can be arbitrarily chosen when $p_2 = 0$.

3 Hamilton–Jacobi Q-Learning

3.1 Semi-Discrete HJB Equations and Asymptotic Consistency

In practice, even though the underlying physical process evolves in continuous time, the observed data, such as sensor measurements, are collected in discrete (sample) time. To design a concrete algorithm for learning the Q-function using such discrete-time data, we propose a novel semi-discrete version of the HJB equation (2.4) *without discretizing or approximating the continuous-time system.* Let $h > 0$ be a fixed *sampling interval*, and let $\mathcal{B} := \{b := \{b_k\}_{k=0}^{\infty} \mid b_k \in \mathbb{R}^m, |b_k| \leq L\}$, where b_k is analogous to $\dot{a}(t)$ in the continuous-time case. Given $(\boldsymbol{x}, \boldsymbol{a}) \in \mathbb{R}^n \times \mathbb{R}^m$ and a sequence $b \in \mathcal{B}$, we let

$$Q^{h,b}(\boldsymbol{x}, \boldsymbol{a}) := h \sum_{k=0}^{\infty} r(x_k, a_k)(1 - \gamma h)^k,$$

where $\{(x_k, a_k)\}_{k=0}^{\infty}$ is defined by $x_{k+1} = \xi(x_k, a_k; h)$ and $a_{k+1} = a_k + hb_k$ with $(x_0, a_0) = (\boldsymbol{x}, \boldsymbol{a})$. Here, $\xi(x_k, a_k; h)$ denotes the state of (2.1) at time $t = h$ with initial state $x(0) = x_k$ and constant action $a(t) \equiv a_k$, $t \in [0, h)$. It is worth emphasizing that our semi-discrete approximation does *not* approximate the system dynamics and thus is more accurate than the standard semi-discrete method. The optimal semi-discrete Q-function $Q^{h,\star} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ is then defined by

$$Q^{h,\star}(\boldsymbol{x}, \boldsymbol{a}) := \sup_{b \in \mathcal{B}} Q^{h,b}(\boldsymbol{x}, \boldsymbol{a}). \tag{3.1}$$

Then, $Q^{h,\star}$ satisfies a semi-discrete version of the HJB equation (2.4).

Proposition 2. *Suppose that $0 < h < \frac{1}{\gamma}$. Then, the function $Q^{h,\star}$ is a solution to the following semi-discrete HJB equation:*

$$Q^{h,\star}(\boldsymbol{x}, \boldsymbol{a}) = hr(\boldsymbol{x}, \boldsymbol{a}) + (1 - \gamma h) \sup_{|\boldsymbol{b}| \leq L} Q^{h,\star}(\xi(\boldsymbol{x}, \boldsymbol{a}; h), \boldsymbol{a} + h\boldsymbol{b}). \tag{3.2}$$

Under Assumption 1, $Q^{h,\star}$ coincides with the unique solution of the semi-discrete HJB equation (3.2). Moreover, the optimal semi-discrete Q-function converges uniformly to its original counterpart in every compact subset of $\mathbb{R}^n \times \mathbb{R}^m$.

Proposition 3. *Suppose that $0 < h < \frac{1}{\gamma}$ and that Assumption 1 holds. Then, the function $Q^{h,\star}$ is the unique solution to the semi-discrete HJB equation* (3.2)*. Furthermore, we have*

$$\lim_{h \to 0} \sup_{(\boldsymbol{x}, \boldsymbol{a}) \in K, K \text{compact}} |Q^{h,\star}(\boldsymbol{x}, \boldsymbol{a}) - Q(\boldsymbol{x}, \boldsymbol{a})| = 0.$$

This proposition justifies the use of the semi-discrete HJB equation for small h. We aim to estimate the optimal Q-function using sample data collected in discrete time, enjoying the benefits of both the semi-discrete HJB equation (3.2) and the original HJB equation (2.4). Namely, the semi-discrete version yields to naturally make use of Q-learning and DQN, and the original version provides an optimal control via (2.5) without requiring a numerical solution for any optimization problems or actor networks as we will see in Section 4.

3.2 Convergence Properties

Consider the following model-free update of Q-function using the semi-discrete HJB equation (3.2): In the kth iteration, for each $(\boldsymbol{x}, \boldsymbol{a})$ we collect data $(x_k := \boldsymbol{x}, a_k := \boldsymbol{a}, r_k, x_{k+1})$ and update the Q-function, with learning rate α_k, by

$$Q_{k+1}^h(\boldsymbol{x}, \boldsymbol{a}) := (1 - \alpha_k) Q_k^h(\boldsymbol{x}, \boldsymbol{a}) \ = \alpha_k \Big[hr_k + (1 - \gamma h) \sup_{|\boldsymbol{b}| \leq L} Q_k^h(x_{k+1}, \boldsymbol{a} + h\boldsymbol{b}) \Big], \tag{3.3}$$

where x_{k+1} is obtained by running (or simulating) the continuous-time system from x_k with action a_k fixed for h period without any approximation, i.e., $x_{k+1} = \xi(x_k, a_k; h)$, and $r_k = r(x_k, a_k)$. We refer to this synchronous Q-learning as *Hamilton–Jacobi Q-learning.* Note that this method is not practically useful because the update must be performed for all state-action pairs in the continuous space. In the following section, we propose a DQN-like algorithm to approximately perform HJ Q-learning employing deep neural networks as function approximators. Before doing so, we identify conditions under which the Q-function updated by (3.3) converges to the optimal semi-discrete Q-function (3.1) in L^∞.

Theorem 2. *Suppose that $0 < h < \frac{1}{\gamma}$, $0 \leq \alpha_k \leq 1$ and that Assumption 1 holds. If the sequence $\{\alpha_k\}_{k=0}^\infty$ of learning rates satisfies $\sum_{k=0}^\infty \alpha_k = \infty$, then*

$$\lim_{k\to\infty} \|Q_k^h - Q^{h,\star}\|_{L^\infty} = 0.$$

Finally, by Propositions 3 and Theorem 2, we establish the following convergence result associating HJ Q-learning (3.3) and the optimal Q-function in the original continuous-time setting.

Corollary 1. *Suppose that $0 \leq \alpha_k \leq 1$ and that Assumption 1 holds. If the sequence $\{\alpha_k\}_{k=0}^\infty$ of learning rates satisfies $\sum_{k=0}^\infty \alpha_k = \infty$ then, for each $0 < h < \frac{1}{\gamma}$, there exists k_h such that $h\sum_{\tau=0}^{k_h-1} \alpha_\tau \to \infty$ as $h \to 0$. Moreover, for such a choice of k_h, we have*

$$\lim_{h\to 0} \sup_{k\geq k_h} \sup_{(\boldsymbol{x},\boldsymbol{a})\in K, K\text{compact}} |Q_k^h(\boldsymbol{x},\boldsymbol{a}) - Q(\boldsymbol{x},\boldsymbol{a})| = 0.$$

4 Hamilton–Jacobi DQN

The convergence result in the previous section suggests that the optimal Q-function can be estimated in a model-free manner through the use of the semi-discrete HJB equation. However, as mentioned, it is intractable to directly implement HJ Q-learning (3.3) over a continuous state-action space. As a practical function approximator, we employ deep neural networks. We then propose the *Hamilton–Jacobi DQN* that approximately performs the update (3.3) *without discretizing or approximating the continuous-time system.* Since our algorithm has no actor, we only consider a parameterized Q-function $Q_\theta(\boldsymbol{x},\boldsymbol{a})$, where θ is the parameter vector of the network.

As with DQN, we use a separate target function Q_{θ^-}, where the network parameter vector θ^- is updated more slowly than θ. This allows us to update θ by solving a regression problem with an almost fixed target, resulting in consistent and stable learning [6]. We also use experience replay by storing transition data (x_k, a_k, r_k, x_{k+1}) in a buffer with fixed capacity and by randomly sampling a mini-batch of transition data $\{(x_j, a_j, r_j, x_{j+1})\}$ to update the target value. This reduces bias by breaking the correlation between sample data that are sequential states [6].

When setting the target value in DQN, the target Q-function needs to be maximized over all admissible actions, i.e., $y_j^- := hr_j + \gamma' \max_{\boldsymbol{a}} Q_{\theta^-}(x_{j+1}, \boldsymbol{a})$. Evaluating the maximum is tractable in the case of discrete action spaces. However, in our case of continuous action spaces, it is computationally challenging to maximize the target Q-function with respect to the action variable. To resolve this issue, we go back to the original HJB equation and use the corresponding optimal action in Theorem 1. Specifically, we consider the action dynamics (2.5) with $b_j := L\frac{\nabla_{\boldsymbol{a}} Q_{\theta^-}(x_j,a_j)}{|\nabla_{\boldsymbol{a}} Q_{\theta^-}(x_j,a_j)|}$ fixed over sampling interval h to obtain

$$a_{j+1} = a_j + hb_j := a_j + hL\frac{\nabla_{\boldsymbol{a}} Q_{\theta^-}(x_j,a_j)}{|\nabla_{\boldsymbol{a}} Q_{\theta^-}(x_j,a_j)|}. \tag{4.1}$$

Algorithm 1: Hamilton–Jacobi DQN

```
Initialize Q-function $Q_\theta$ with random weights $\theta$, and target Q-function $Q_{\theta^-}$ with weights
$\theta^- = \theta$;
Initialize replay buffer with fixed capacity;
for episode = 1 to $M$ do
  Randomly sample initial state-action pair $(x_0, a_0)$;
  for $k = 0$ to $K$ do
    Execute action $a_k$ and observe reward $r_k$ and the next state $x_{k+1}$;
    Store $(x_k, a_k, r_k, x_{k+1})$ in buffer;
    Sample the random mini-batch $\{(x_j, a_j, r_j, x_{j+1})\}$ from buffer;
    Set $y_j^- := hr_j + (1-\gamma h)Q_{\theta^-}(x_{j+1}, a_j') \ \forall j$ where $a_j' := a_j + hL\frac{\nabla_{\boldsymbol{a}} Q_\theta(x_j,a_j)}{|\nabla_{\boldsymbol{a}} Q_\theta(x_j,a_j)|}$;
    Update $\theta$ by minimizing $\sum_j (y_j^- - Q_\theta(x_j, a_j))^2$;
    Update $\theta^- \leftarrow (1-\alpha)\theta^- + \alpha\theta$ for $\alpha \ll 1$;
    Set the next action as $a_{k+1} := a_k + hL\frac{\nabla_{\boldsymbol{a}} Q_\theta(x_k,a_k)}{|\nabla_{\boldsymbol{a}} Q_\theta(x_k,a_k)|} + \varepsilon$, where $\varepsilon \sim N(0, \sigma^2 I_m)$;
  end for
end for
```

Using this optimal control action, we can approximate the maximal target Q-function value as $\max_{|\boldsymbol{a}-a_j|\leq hL} Q_{\theta^-}(x_{j+1}, \boldsymbol{a}) \approx Q_{\theta^-}(x_{j+1}, a_j + hb_j)$. This approximation becomes more accurate as h decreases.

Proposition 4. *Suppose that Q_{θ^-} is twice continuously differentiable with bounded first and second derivatives. If $\nabla_{\boldsymbol{a}} Q_{\theta^-}(x_j, a_j) \neq 0$, we have*

$$\lim_{h\to 0} \left| \max_{|\boldsymbol{a}-a_j|\leq hL} Q_{\theta^-}(x_{j+1}, \boldsymbol{a}) - Q_{\theta^-}(x_{j+1}, a_j + hb_j) \right| = 0.$$

Moreover, the difference above is $O(h^2)$ as $h \to 0$.

The major advantage of using the optimal action obtained in the continuous-time case is to avoid explicitly solving the nonlinear optimization problem $\max_{|\boldsymbol{a}-a_j|\leq hL} Q_{\theta^-}(x_{j+1}, \boldsymbol{a})$, which is computationally demanding. With this choice of target Q-function value and the semi-discrete HJB equation (3.2), we set the target value as $y_j^- := hr_j + (1-\gamma h)Q_{\theta^-}(x_{j+1}, a_j + hb_j)$. To mitigate the overestimation of Q-functions, we can employ double Q-learning [48] by simply modifying b_j as $b_j := L\frac{\nabla_{\boldsymbol{a}} Q_\theta(x_j,a_j)}{|\nabla_{\boldsymbol{a}} Q_\theta(x_j,a_j)|}$ to use a greedy action with respect to Q_θ instead of Q_{θ^-}. In this double Q-learning version, Proposition 4 remains valid except for the $O(h^2)$ convergence rate. The network parameter θ can then be trained to minimize the loss function $\sum_j (y_j^- - Q_\theta(x_j, a_j))^2$. For exploration, we add the additional Gaussian noise $\varepsilon \sim N(0, \sigma^2 I_m)$ to generate the next action as $a_{k+1} := a_k + hL\frac{\nabla_{\boldsymbol{a}} Q_\theta(x_k,a_k)}{|\nabla_{\boldsymbol{a}} Q_\theta(x_k,a_k)|} + \varepsilon$. The overall algorithm is presented in Algorithm 1.[5]

4.1 Discussion

We now discuss a few notable features of HJ DQN with regard to existing works:

No use of parameterized policies. Most of model-free deep RL algorithms for continuous control use actor-critic methods [16, 18–20] or policy gradient methods [14, 21] to deal with continuous action spaces. In these methods, by parametrizing policies, the policy improvement step is

[5]When $\nabla_{\boldsymbol{a}} Q_\theta(x_j, a_j) = 0$, $\frac{\nabla_{\boldsymbol{a}} Q_\theta(x_j,a_j)}{|\nabla_{\boldsymbol{a}} Q_\theta(x_j,a_j)|}$ is replaced by an arbitrary vector with norm 1 of the same size.

performed in the space of network weights. By doing so, they avoid solving possibly complicated optimization problems over the policy or action spaces. However, these methods are subject to the issue of being stuck at local optima in the policy (parameter) space due to the use of gradient-based algorithms, as pointed out in the literature regarding policy gradient/search [49–51] and actor-critic methods [52]. Moreover, it is reported that the policy-based methods are sensitive to hyperparameters [53]. Departing from these algorithms, HJ DQN is a value-based method for continuous control without requiring the use of an actor or a parameterized policy. Previous value-based methods for continuous control (e.g., [13]) have a computational challenge in finding a greedy action, which requires a solution to a nonlinear program. Our method avoids numerically optimizing Q-functions over the continuous action space through the use of the optimal control (2.5). This is a notable benefit of the proposed HJB framework.

Continuous-time control. Many existing RL methods for continuous-time dynamical systems have been designed for linear systems [22–24] or control-affine systems [25, 27–29], in which value functions and optimal policies can be represented in a simple form. For general nonlinear systems, Hamilton–Jacobi–Bellman equations have been considered as the optimality equations for state-value functions $v(\boldsymbol{x})$ [7,32,34,35]. Unlike these methods, our method uses variant of Q-function and thus benefits from modern deep RL techniques developed in the literature on DQN. Moreover, as opposed to discrete-time RL methods, it does not discretize or approximate the system dynamics and has the flexibility of choosing the sampling interval h in its algorithm design, without needing a sophisticated ODE discretization method.

4.2 Smoothing

A potential defect of our Lipschitz constrained control setting is that the rate of change in action has a constant norm $L\frac{\nabla_{\boldsymbol{a}} Q(x^\star,a^\star)}{|\nabla_{\boldsymbol{a}} Q(x^\star,a^\star)|}$. This is also observed in Algorithm 1, where the action is updated by $hL\frac{\nabla_{\boldsymbol{a}} Q_\theta(x_j,a_j)}{|\nabla_{\boldsymbol{a}} Q_\theta(x_j,a_j)|}$. Therefore, the magnitude of fluctuations in action is always fixed as hL, which may lead to the oscillatory behavior of action. Such oscillatory behaviors are not uncommon in optimal control (e.g., bang-bang solutions). To alleviate this potential issue, one may introduce an additional smoothing process when updating action. Inspired by [54], we modify the term $\frac{\nabla_{\boldsymbol{a}} Q_\theta(x_j,a_j)}{|\nabla_{\boldsymbol{a}} Q_\theta(x_j,a_j)|}$ by multiplying a smoothing function. Instead of using $hL\frac{\nabla_{\boldsymbol{a}} Q_\theta(x_j,a_j)}{|\nabla_{\boldsymbol{a}} Q_\theta(x_j,a_j)|}$ in the update of action, we suggest to use

$$hL\frac{\phi(|\nabla_{\boldsymbol{a}} Q_\theta(x_j,a_j)|)\nabla_{\boldsymbol{a}} Q_\theta(x_j,a_j)}{|\nabla_{\boldsymbol{a}} Q_\theta(x_j,a_j)|},$$

where $\phi : [0,+\infty) \to [0,1]$ is an increasing function with $\phi(0) = 0$ and $\lim_{r\to\infty} \phi(r) = 1$. A typical example of such a function ϕ is $\phi(r) = \tanh\left(\frac{r}{L}\right)$ or $\phi(r) = \frac{r}{L+r}$. This action update rule is expected to remove the undesirable oscillatory behavior of action, as confirmed in Section 5.

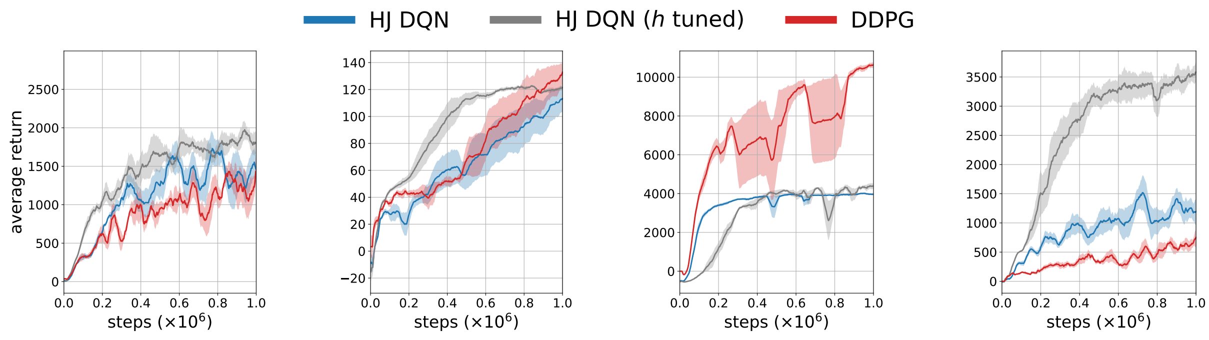
5 Experiments

In this section, we present the empirical performance of our method on benchmark tasks as well as LQ problems. The source code of our HJ DQN implementation is available online [6].

5.1 Actor Networks vs. Optimal Control ODE

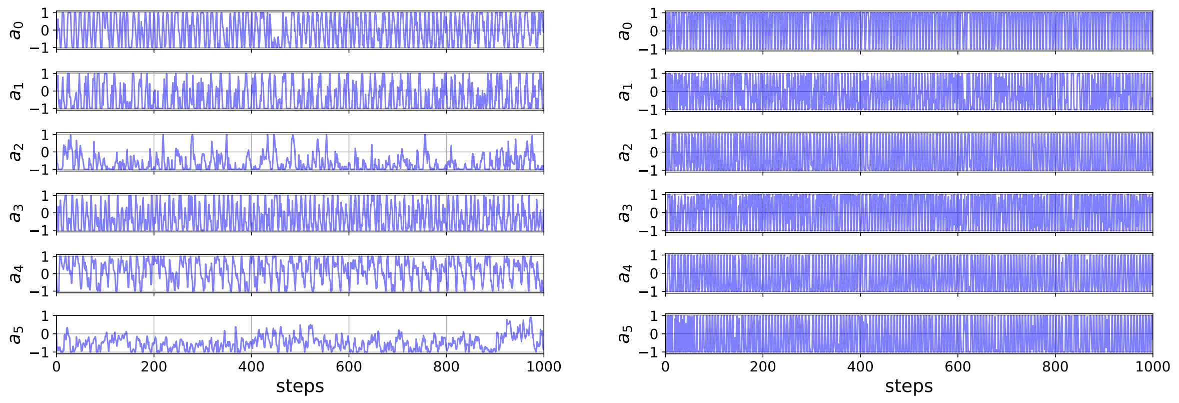
We choose deep deterministic policy gradient (DDPG) [16] as a baseline to compare since it is another variant of DQN for continuous control. DDPG is an actor-critic method using separate

[6]`https://github.com/HJDQN/HJQ`



(a) Hopper-v2 (b) Swimmer-v2 (c) HalfCheetah-v2 (d) Walker2d-v2

Figure 1: Learning curves for the OpenAI gym continuous control tasks.

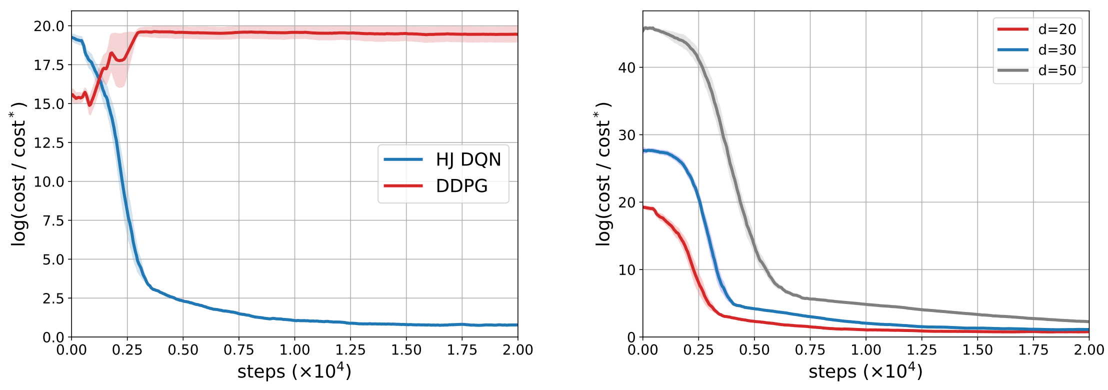


(a) HJ DQN (b) DDPG

Figure 2: Action trajectories obtained by HJ DQN and DDPG for HalfCheetah-v2.

actor networks while ours is a valued-based method that does not use a parameterized policy. Although there are state-of-the-art methods built upon DDPG, such as TD3 [19] and SAC [18], we focus on the comparison between ours and DDPG to examine whether the role of actor networks can be replaced by the optimal control characterized through our HJB equation. The hyperparameters used in the experiments are reported in Appendix D.

We consider continuous control benchmark tasks in OpenAI gym [10] simulated by MuJoCo engine [9]. Figure 1 shows the learning curves for both methods, each of which is tested with five different random seeds for 1 million steps. The solid curve represents the average of returns over 20 consecutive evaluations, while the shaded regions represent half a standard deviation of the average evaluation over five trials. As shown in Figure 1, the performance of our method is comparable to that of DDPG when the default sampling interval is used. Ours outperforms DDPG on Walker2d-v2 while the opposite result is observed in the case of HalfCheetah-v2. As sampling interval h is a hyperparameter of Algorithm 1, we also identify an optimal h for each task, other than the default sampling interval. When we test the different sampling interval, we also tune the learning rate α, as suggested in [43]. Precisely, when the sampling interval is multiplied by a constant from the default interval, the learning rate is also multiplied by the same constant. Except the HalfCheetah-v2, the final performances or learning rate are improved, compared to the default sampling interval. Overall, the results indicate that actor networks may be replaced by the ODE



(a) (b)

Figure 3: Learning curves for the LQ problem: (a) the comparison between HJ DQN and DDPG, and (b) the effect of problem sizes.

characterization (2.5) of optimal control obtained using our HJB framework. Without using actor networks, our method has clear advantages over DDPG in terms of hyperparameter tuning and computational burden.

In Figure 2, we report the action trajectories obtained by HJ DQN and DDPG for HalfCheetah-v2. The action trajectories obtained by HJ DQN is less oscillating compared to DDPG. This confirms the fact that oscillations in action are not uncommon in optimal control. In this particular case of HalfCheetah-v2, where DDPG outperforms HJ DQN, we suspect that fast changes in action may be needed for good performance. Oscillatory actions may be beneficial to some control tasks.

5.2 Linear-Quadratic Problems

We now consider a classical LQ problem with system dynamics

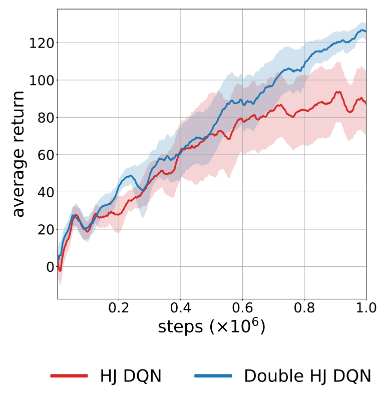
$$\dot{x}(t) = Ax(t) + Ba(t), \quad t > 0, \quad x(t),\, a(t) \in \mathbb{R}^d,$$

and reward function

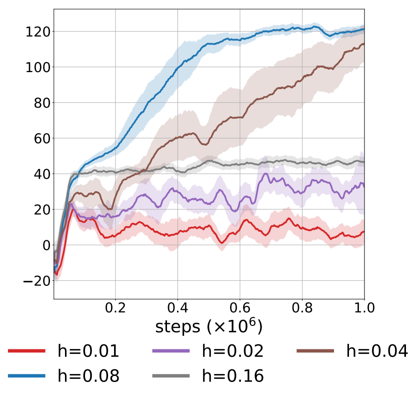
$$r(\boldsymbol{x}, \boldsymbol{a}) = -(\boldsymbol{x}^\top Q\boldsymbol{x} + \boldsymbol{a}^\top R\boldsymbol{a}),$$

where $Q = Q^\top \succeq 0$ and $R = R^\top \succ 0$. Note that our method solves a slight different problem due to the Lipschitz constraint on controls. Thus, the control learned by our method must be suboptimal.

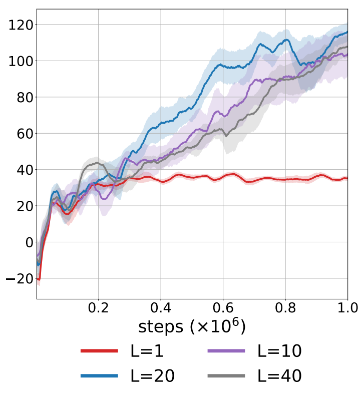
Each component of the system matrices $A \in \mathbb{R}^{d\times d}$ and $B \in \mathbb{R}^{d\times d}$ was generated uniformly from $[-0.1, 0.1]$ and $[-0.5, 0.5]$, respectively. The produced matrix A has an eigenvalue with positive real part, and therefore the system is unstable. The discount factor γ and Lipschitz constant L are set to be $e^{-\gamma h} = 0.99999$ and $L = 10$. We first compare the performance of HJ DQN with DDPG for the case of $d = 20$ and report the results in Figure 3a. The learning curves are plotted in the same manner as the ones in Section 5.1. The y-axis of each figure is the log of the ratio between the actual cost and the optimal cost. Therefore, the curve approaches the x-axis as the performance improves. The result implies that the DDPG cannot reduce the cost at all, whereas HJ DQN successfully learns an effective (suboptimal) policy. In Figure 3b, we present the learning curves for HJ DQN with different system sizes. Although learning speed is affected by the problem size, HJ DQN can successfully solve the LQ problem with high-dimensional systems ($d = 50$). Moreover,



(a) Double Q-learning



(b) Sampling interval



(c) Control constraint

Figure 4: Results of the ablation study using the Swimmer-v2 with respect to (a) double Q-learning, (b) sampling interval h, and (c) control constraint.

it is observed that the standard deviations over trials are relatively small, and the learning curves have almost no variation over trials after approximately 10^4 steps.

5.3 Ablation Study

We make ablations and modifications to HJ DQN to understand the contribution of each component. Figure 4 presents the results for the following design evaluation experiments.

Double Q-learning. We first modify our algorithm to test whether double Q-learning contributes to the performance of our algorithm, as in DQN. Specifically, when selecting actions to update the target value, we instead use $b_j := L\frac{\nabla_{\boldsymbol{a}} Q_{\theta^-}(x_j, a_j)}{|\nabla_{\boldsymbol{a}} Q_{\theta^-}(x_j, a_j)|}$ to remove the effects of double Q-learning. Figure 4 (a) shows that double Q-learning improves the final performance. This observation is consistent with the effect of double Q-learning in DQN. Moreover, double Q-learning reduces the variance of the average return, indicating its contribution to the stability of our algorithm.

Sampling interval. To understand the effect of sampling interval h, we run our algorithm with multiple values of h. As we mentioned before, we also adjust the learning rate α according to the sampling interval. As shown in Figure 4 (b), the final performance and learning speed increase as h varies from 0.01 to 0.08 and the final performance decreases as h varies from 0.08 to 0.16. When h is too small, each episode has too many sampling steps; thus, the network is trained in a small number of episodes given fixed total steps. This limits exploration, thereby decreasing the performance of our algorithm. On the other hand, as Proposition 4 implies, the target error increases with sampling interval h. This error is dominant in the case of large h. Therefore, there exists an optimal sampling interval ($h = 0.08$ in this task) that presents the best performance.

Control constraint. Recall that admissible controls satisfy the constraint $|\dot{a}(t)| \leq L$. The parameter L can be derived from specific control problems or considered as a design choice. We consider the latter case and display the effect of L on the learning curves in Figure 4 (c). The final reward is the lowest, compared to others, in the case of $L = 1$ because the set of admissible controls is too small to allow rapid changes in control signals. HJ DQN, with large enough L (≥ 10), presents a similar learning speed and performance. The final performance and learning speed slightly decrease as L varies from 20 to 40. This is due to too large variation and frequent switching in action values, prohibiting a consistent improvement of Q-functions.

Smoothing. Finally, we present the effect of the smoothing process introduced in Section 4.2.

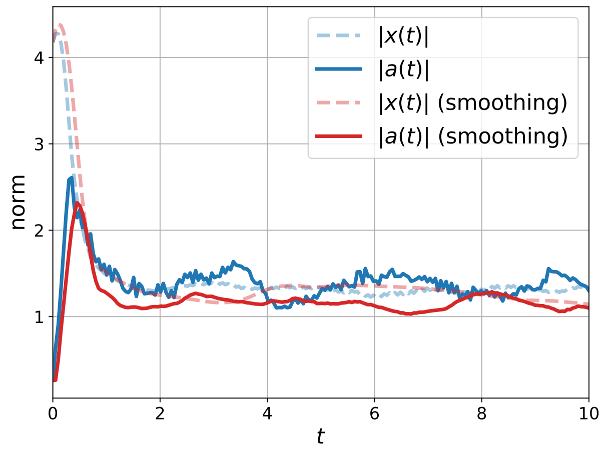


Figure 5: Effect of smoothing on the 20-dimensional LQ problem.

Figure 5 shows $|x(t)|$ and $|a(t)|$ generated by the control learned with and without smoothing on the 20-dimensional LQ problem. Here, $\phi(r) = \tanh\left(\frac{r}{L}\right)$ is chosen as the smoothing function. As expected, with no smoothing process, the action trajectory shows wobbling oscillations (solid blue line). However, when the smoothing process is applied, the action trajectory has no such undesirable oscillations and presents a smooth behavior (solid red line). Regarding $|x(t)|$, the smoothing process has only a small effect. Therefore, the smoothing process can eliminate oscillations in action without significantly affecting the state trajectory.

6 Conclusions

We have presented a new theoretical and algorithmic framework that extends DQN to continuous-time deterministic optimal control for continuous action space. A novel class of HJB equations for Q-functions has been derived and used to construct a Q-learning method for continuous-time control. We have shown the theoretical convergence properties of this method. For practical implementation, we have combined the HJB-based method with DQN, resulting in a simple algorithm that solves continuous-time control problems without an actor network. Benefiting from our theoretical analysis of the HJB equations, this model-free off-policy algorithm does not require any numerical optimization for selecting greedy actions. The result of our experiments indicates that actor networks in DDPG may be replaced by our optimal control simply characterized via an ODE, while reducing computational effort. Our HJB framework may provide an exciting avenue for future research in continuous-time RL in terms of improving the exploration capability with maximum entropy methods, and exploiting the benefits of models with theoretical guarantees.

A Viscosity Solution of the Hamilton–Jacobi Equations

The Hamilton–Jacobi equation is a partial differential equation of the form

$$F(\boldsymbol{z}, u(\boldsymbol{z}), \nabla_{\boldsymbol{z}} u(\boldsymbol{z})) = 0, \quad \boldsymbol{z} \in \mathbb{R}^k, \tag{A.1}$$

where $F : \mathbb{R}^k \times \mathbb{R} \times \mathbb{R}^k \to \mathbb{R}$. A function $u : \mathbb{R}^k \to \mathbb{R}$ that solves the HJ equation is called a (strong) solution. However, such a strong solution exists only in limited cases. To consider a broad class of HJ equations, it is typical to adopt the concept of weak solutions. Among these, the *viscosity solution* is the most relevant to dynamic programming and optimal control problems [46, 47]. Specifically, under a technical condition, the viscosity solution is unique and corresponds to the value function of a continuous-time optimal control problem. In the following definition, $C(\mathbb{R}^k)$ and $C^1(\mathbb{R}^k)$ denote the set of continuous functions and the set of continuously differentiable functions respectively.

Definition 1. *A function $u \in C(\mathbb{R}^k)$ is called the* viscosity solution *of* (A.1) *if it satisfies the following conditions:*

1. *For any $\phi \in C^1(\mathbb{R}^k)$ such that $u - \phi$ attains a local maximum at $\boldsymbol{z}_0$,*
$$F(\boldsymbol{z}_0, u(\boldsymbol{z}_0), \nabla_{\boldsymbol{z}}\phi(\boldsymbol{z}_0)) \leq 0;$$
2. *For any $\phi \in C^1(\mathbb{R}^k)$ such that $u - \phi$ attains a local minimum at $\boldsymbol{z}_0$,*
$$F(\boldsymbol{z}_0, u(\boldsymbol{z}_0), \nabla_{\boldsymbol{z}}\phi(\boldsymbol{z}_0)) \geq 0.$$

Note that the viscosity solution does not need to be differentiable. In our case, the HJB equation (2.4)
$$\gamma Q(\boldsymbol{x}, \boldsymbol{a}) - \nabla_{\boldsymbol{x}} Q(\boldsymbol{x}, \boldsymbol{a}) \cdot f(\boldsymbol{x}, \boldsymbol{a}) - L|\nabla_{\boldsymbol{a}} Q(\boldsymbol{x}, \boldsymbol{a})| - r(\boldsymbol{x}, \boldsymbol{a}) = 0$$
can be expressed as (A.1) with
$$F(\boldsymbol{z}, q, \boldsymbol{p}) = \gamma q - \boldsymbol{p}_1 \cdot f(\boldsymbol{z}) - L|\boldsymbol{p}_2| - r(\boldsymbol{z}),$$
where $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{a}) \in \mathbb{R}^n \times \mathbb{R}^m$ and $\boldsymbol{p} = (\boldsymbol{p}_1, \boldsymbol{p}_2) \in \mathbb{R}^n \times \mathbb{R}^m$. We can show that the HJB equation admits a unique viscosity solution, which coincides with the optimal Q-function.

Theorem 3. *Suppose that Assumption 1 holds.*[7] *Then, the optimal continuous-time Q-function is the unique viscosity solution to the HJB equation* (2.4).

Proof. First, recall that our control trajectory satisfies the constraint $|\dot{a}| \leq L$. Therefore, our dynamical system can be written in the following extended form:
$$\dot{x}(t) = f(x(t), a(t)), \quad \dot{a}(t) = b(t), \quad t > 0, \quad |b(t)| \leq L,$$
by viewing $x(t)$ and $a(t)$ as state variables. More precisely, the dynamics of the extended state variable $z(t) = (x(t), a(t))$ can be written as
$$\dot{z}(t) = G(z(t), b(t)), \quad t > 0, \quad |b(t)| \leq L, \tag{A.2}$$
where $G(\boldsymbol{z}, \boldsymbol{b}) = (f(\boldsymbol{z}), \boldsymbol{b})$. Applying the dynamic programming principle to the Q-function, we have
$$Q(\boldsymbol{z}) = \sup_{|b(t)| \leq L} \left\{ \int_t^{t+h} e^{-\gamma(s-t)} r(z(s))\,\mathrm{d}s + e^{-\gamma h} Q(z(t+h)) \mid z(t) = \boldsymbol{z} \right\}.$$
The remaining proof is almost the same as the proof of Proposition 2.8, Chapter 3 in [46]. However, for the self-completeness of the paper, we provide a detailed proof. In the following, we show that the Q-function satisfies the two conditions in Definition 1.

First, let $\phi \in C^1(\mathbb{R}^{n+m})$ such that $Q - \phi$ attains a local maximum at $\boldsymbol{z}$. Then, there exists $\delta > 0$ such that $Q(\boldsymbol{z}) - Q(\boldsymbol{z}') \geq \phi(\boldsymbol{z}) - \phi(\boldsymbol{z}')$ for $|\boldsymbol{z}' - \boldsymbol{z}| < \delta$. Since f and r are bounded

[7]Assumption 1 can be relaxed by using a modulus associated with each function as in Chapter III.1–3 in [46].

Lipschitz continuous, there exists $h_0 > 0$, which is independent of $b(s)$, such that $|z(s) - \boldsymbol{z}| \leq \delta$, $|r(z(s)) - r(\boldsymbol{z})| \leq C(s-t)$ and $|f(z(s)) - f(\boldsymbol{z})| \leq C(s-t)$ for $t \leq s \leq t + h_0$, where $z(s)$ is a solution to (A.2) for $s \geq t$ with $z(t) = \boldsymbol{z}$. Now, the dynamic programming principle for the Q-function implies that, for any $0 < h < h_0$ and $\varepsilon > 0$, there exists $b(s)$ with $|b(s)| \leq L$ such that

$$Q(\boldsymbol{z}) \leq \int_t^{t+h} e^{-\gamma(s-t)} r(z(s)) \, \mathrm{d}s + e^{-\gamma h} Q(z(t+h)) + h\varepsilon,$$

where $z(s)$ is now a solution to (A.2) with $z(t) = \boldsymbol{z}$ under the particular choice of b. On the other hand, it follows from our choice of h that

$$\int_t^{t+h} e^{-\gamma(s-t)} r(z(s)) \, \mathrm{d}s = \int_t^{t+h} e^{-\gamma(s-t)} r(\boldsymbol{z}) \, \mathrm{d}s + o(h),$$

which implies that

$$Q(\boldsymbol{z}) \leq \int_t^{t+h} e^{-\gamma(s-t)} r(\boldsymbol{z}) \, \mathrm{d}s + e^{-\gamma h} Q(z(t+h)) + h\varepsilon + o(h).$$

Therefore, we have

$$\begin{aligned} \phi(\boldsymbol{z}) - \phi(z(t+h)) &\leq Q(\boldsymbol{z}) - Q(z(t+h)) \\ &\leq \int_t^{t+h} e^{-\gamma(s-t)} r(\boldsymbol{z}) \, \mathrm{d}s + (e^{-\gamma h} - 1) Q(z(t+h)) + h\varepsilon + o(h). \end{aligned}$$

Since the left-hand side of the inequality above is equal to $-\int_t^{t+h} \frac{\mathrm{d}}{\mathrm{d}s} \phi(z(s)) \, \mathrm{d}s = -\int_t^{t+h} \nabla_{\boldsymbol{z}} \phi(z(s)) \cdot G(z(s), b(s)) \, \mathrm{d}s$, we obtain that

$$\begin{aligned} 0 &\leq \int_t^{t+h} \nabla_{\boldsymbol{z}} \phi(z(s)) \cdot G(z(s), b(s)) \, \mathrm{d}s \\ &\quad + \int_t^{t+h} e^{-\gamma(s-t)} r(\boldsymbol{z}) \, \mathrm{d}s + \left(e^{-\gamma h} - 1\right) Q(z(t+h)) + h\varepsilon + o(h) \\ &\leq \int_t^{t+h} \left(\nabla_{\boldsymbol{x}} \phi(z(s)) \cdot f(\boldsymbol{z}) + L |\nabla_{\boldsymbol{a}} \phi(z(s))|\right) \mathrm{d}s \\ &\quad + \int_t^{t+h} e^{-\gamma(s-t)} r(\boldsymbol{z}) \, \mathrm{d}s + \left(e^{-\gamma h} - 1\right) Q(z(t+h)) + h\varepsilon + o(h). \end{aligned}$$

By dividing both sides by h and letting $h \to 0$, we conclude that

$$\nabla_{\boldsymbol{x}} \phi(\boldsymbol{z}) \cdot f(\boldsymbol{z}) + L |\nabla_{\boldsymbol{a}} \phi(\boldsymbol{z})| + r(\boldsymbol{z}) - \gamma Q(\boldsymbol{z}) + \varepsilon \geq 0.$$

Since ε was arbitrarily chosen, we confirm that the Q-function satisfies the first condition in Definition 1, i.e.,

$$\gamma Q(\boldsymbol{z}) - \nabla_{\boldsymbol{x}} \phi(\boldsymbol{z}) \cdot f(\boldsymbol{z}) - L |\nabla_{\boldsymbol{a}} \phi(\boldsymbol{z})| - r(\boldsymbol{z}) \leq 0.$$

We now consider the second condition. Let $\phi \in C^1(\mathbb{R}^{n+m})$ such that $Q - \phi$ attains a local minimum at $\boldsymbol{z}$, i.e., there exists δ such that $Q(\boldsymbol{z}) - Q(\boldsymbol{z}') \leq \phi(\boldsymbol{z}) - \phi(\boldsymbol{z}')$ for $|\boldsymbol{z}' - \boldsymbol{z}| < \delta$. Fix an arbitrary $\boldsymbol{b} \in \mathbb{R}^m$ such that $|\boldsymbol{b}| \leq L$ and let $b(s) \equiv \boldsymbol{b}$ be a constant function. Let $z(s)$ be a solution to (A.2) for $s \geq t$ with $z(t) = \boldsymbol{z}$ under the particular choice of $b(s) \equiv \boldsymbol{b}$. Then, for sufficiently small h, $|z(t+h) - \boldsymbol{z}| \leq \delta$, and therefore we have

$$\begin{aligned} Q(\boldsymbol{z}) - Q(z(t+h)) \leq \phi(\boldsymbol{z}) - \phi(z(t+h)) &= -\int_t^{t+h} \frac{\mathrm{d}}{\mathrm{d}s}\phi(z(s))\,\mathrm{d}s \\ &= -\int_t^{t+h} \nabla_{\boldsymbol{z}}\phi(z(s)) \cdot G(z(s), \boldsymbol{b})\,\mathrm{d}s. \end{aligned} \tag{A.3}$$

On the other hand, the dynamic programming principle yields

$$Q(\boldsymbol{z}) - Q(z(t+h)) \geq \int_t^{t+h} e^{-\gamma(s-t)} r(z(s))\,\mathrm{d}s + (e^{-\gamma h} - 1)Q(z(t+h)). \tag{A.4}$$

By (A.3) and (A.4), we have

$$(e^{-\gamma h} - 1)Q(z(t+h)) + \int_t^{t+h} e^{-\gamma(s-t)} r(z(s))\,\mathrm{d}s \leq -\int_t^{t+h} \nabla_{\boldsymbol{z}}\phi(z(s)) \cdot G(z(s), \boldsymbol{b})\,\mathrm{d}s.$$

Dividing both sides by h and letting $h \to 0$, we obtain that

$$-\gamma Q(\boldsymbol{z}) + r(\boldsymbol{z}) \leq -\nabla_{\boldsymbol{z}}\phi(\boldsymbol{z}) \cdot (f(\boldsymbol{z}), \boldsymbol{b}),$$

or equivalently

$$\gamma Q(\boldsymbol{z}) - \nabla_{\boldsymbol{x}}\phi(\boldsymbol{z}) \cdot f(\boldsymbol{z}) - \nabla_{\boldsymbol{a}}\phi(\boldsymbol{z}) \cdot \boldsymbol{b} - r(\boldsymbol{z}) \geq 0.$$

Since $\boldsymbol{b}$ was arbitrarily chosen from $\{\boldsymbol{b} \in \mathbb{R}^m : |\boldsymbol{b}| \leq L\}$, we have

$$\gamma Q(\boldsymbol{z}) - \nabla_{\boldsymbol{x}}\phi(\boldsymbol{z}) \cdot f(\boldsymbol{z}) - L|\nabla_{\boldsymbol{a}}\phi(\boldsymbol{z})| - r(\boldsymbol{z}) \geq 0,$$

which confirms that the Q-function satisfies the second condition in Definition 1. Therefore, we conclude that the Q-function is a viscosity solution of the HJB equation (2.4).

Lastly, the uniqueness of the viscosity solution can be proved by using Theorem 2.12, Chapter 3 in [46]. □

B Proofs

B.1 Proposition 1

Proof. Fix $(\boldsymbol{x}, \boldsymbol{a}, t) \in \mathbb{R}^n \times \mathbb{R}^m \times [0, T]$. Let ε be an arbitrary positive constant. Then, there exists $a \in \mathcal{A}$ such that $\int_t^T r(x(s), a(s))\,\mathrm{d}s + q(x(T)) < v(\boldsymbol{x}, t) + \varepsilon$, where $x(s)$ satisfies (2.1) with $x(t) = \boldsymbol{x}$ in the Carathéodory sense: $x(s) = \boldsymbol{x} + \int_t^s f(x(\tau), a(\tau))\,\mathrm{d}\tau$. We now construct a new control $\tilde{a} \in \mathcal{A}$ as $\tilde{a}(s) := \boldsymbol{a}$ if $s = t$; $\tilde{a}(s) := a(s)$ if $s > t$. Such a modification of controls at a single point does not affect the trajectory or the total cost. Therefore, we have

$$v(\boldsymbol{x}, t) \leq Q(\boldsymbol{x}, \boldsymbol{a}, t) \leq \int_t^T r(x(s), \tilde{a}(s))\,\mathrm{d}s + q(x(T)) < v(\boldsymbol{x}, t) + \varepsilon. \tag{B.1}$$

Since ε was arbitrarily chosen, we conclude that $v(\boldsymbol{x}, t) = Q(\boldsymbol{x}, \boldsymbol{a}, t)$ for any $\boldsymbol{u} \in \mathbb{R}^m$. □

B.2 Theorem 1

Proof. The classical theorem for the necessary and sufficient condition of optimality (e.g. Theorem 2.54, Chapter III in [46]) implies that $a^\star$ is optimal among those in $\mathcal{A}$ such that $a(t) = \boldsymbol{a}$ if and only if

$$\begin{aligned}&p_1 \cdot f(x^\star(s), a^\star(s)) + p_2 \cdot \dot{a}^\star(s) + r(x^\star(s), a^\star(s))\\ &= \max_{|\boldsymbol{b}|\le L}\{p_1 \cdot f(x^\star(s), a^\star(s)) + p_2 \cdot \boldsymbol{b} + r(x^\star(s), a^\star(s))\}\end{aligned}$$

for all $p = (p_1, p_2) \in D^{\pm}Q(x^\star(s), a^\star(s))$. This optimality condition can be expressed as the desired ODE (2.5). Thus, its solution $a^\star$ with $a^\star(t) = \boldsymbol{a}$ satisfies (2.6).

Suppose now that $\boldsymbol{a} \in \arg\max_{\boldsymbol{a}'\in\mathbb{R}^m} Q(\boldsymbol{x}, \boldsymbol{a}')$. It follows from the definition of Q that

$$\begin{aligned}&\max_{a\in\mathcal{A}}\left\{\int_t^\infty e^{-\gamma(s-t)} r(x(s), a(s))\,\mathrm{d}s \mid x(t) = \boldsymbol{x}\right\}\\ &= \max_{\boldsymbol{a}'\in\mathbb{R}^m}\max_{a\in\mathcal{A}}\left\{\int_t^\infty e^{-\gamma(s-t)} r(x(s), a(s))\,\mathrm{d}s \mid x(t) = \boldsymbol{x}, a(t) = \boldsymbol{a}'\right\}\\ &= \max_{\boldsymbol{a}'\in\mathbb{R}^m} Q(\boldsymbol{x}, \boldsymbol{a}') = Q(\boldsymbol{x}, \boldsymbol{a}) = \max_{a\in\mathcal{A}}\left\{\int_t^\infty e^{-\gamma(s-t)} r(x(s), a(s))\mathrm{d}s \mid x(t) = \boldsymbol{x}, a(t) = \boldsymbol{a}\right\}\\ &= \int_t^\infty e^{-\gamma(s-t)} r(x^\star(s), a^\star(s))\,\mathrm{d}s.\end{aligned}$$

Therefore, $a^\star$ is an optimal control. □

B.3 Proposition 2

Proof. We first show that $Q^{h,\star}$ satisfies (3.2). Fix an arbitrary sequence $b := \{b_n\}_{n=0}^\infty \in \mathcal{B}$. It follows from the definition of $Q^{h,b}$ that

$$Q^{h,b}(\boldsymbol{x}, \boldsymbol{a}) = hr(\boldsymbol{x}, \boldsymbol{a}) + (1 - \gamma h)Q^{h,\tilde{b}}(\xi(\boldsymbol{x}, \boldsymbol{a}; h), \boldsymbol{a} + hb_0).$$

where $\tilde{b} := \{b_1, b_2, \ldots\} \in \mathcal{B}$. Since $Q^{h,\tilde{b}}(\xi(\boldsymbol{x}, \boldsymbol{a}; h), \boldsymbol{a} + hb_0) \le Q^{h,\star}(\xi(\boldsymbol{x}, \boldsymbol{a}; h), \boldsymbol{a} + hb_0)$, we have

$$\begin{aligned}Q^{h,b}(\boldsymbol{x}, \boldsymbol{a}) &\le hr(\boldsymbol{x}, \boldsymbol{a}) + (1 - \gamma h)Q^{h,\star}(\xi(\boldsymbol{x}, \boldsymbol{a}; h), \boldsymbol{a} + hb_0)\\ &\le hr(\boldsymbol{x}, \boldsymbol{a}) + (1 - \gamma h)\sup_{|\boldsymbol{b}|\le L}\left\{Q^{h,\star}(\xi(\boldsymbol{x}, \boldsymbol{a}; h), \boldsymbol{a} + h\boldsymbol{b})\right\}.\end{aligned}$$

Taking supremum of both sides with respect to $b \in \mathcal{B}$ yields

$$Q^{h,\star}(\boldsymbol{x}, \boldsymbol{a}) \le hr(\boldsymbol{x}, \boldsymbol{a}) + (1 - \gamma h)\sup_{|\boldsymbol{b}|\le L}\left\{Q^{h,\star}(\xi(\boldsymbol{x}, \boldsymbol{a}; h), \boldsymbol{a} + h\boldsymbol{b})\right\}. \tag{B.2}$$

To obtain the other direction of inequality, we fix an arbitrary $\boldsymbol{b} \in \mathbb{R}^m$ such that $|\boldsymbol{b}| \le L$. Let $\boldsymbol{x}' := \xi(\boldsymbol{x}, \boldsymbol{a}; h)$ and $\boldsymbol{a}' := \boldsymbol{a} + h\boldsymbol{b}$. Fix an arbitrary $\varepsilon > 0$ and choose a sequence $c := \{c_n\}_{n=0}^\infty \in \mathcal{B}$ such that

$$Q^{h,\star}(\boldsymbol{x}', \boldsymbol{a}') \le Q^{h,c}(\boldsymbol{x}', \boldsymbol{a}') + \varepsilon.$$

We now construct a new sequence $\tilde{c} := \{\boldsymbol{b}, c_0, c_1, \ldots\} \in \mathcal{B}$. Then,

$$Q^{h,\tilde{c}}(\boldsymbol{x}, \boldsymbol{a}) = hr(\boldsymbol{x}, \boldsymbol{a}) + (1 - \gamma h)Q^{h,c}(\boldsymbol{x}', \boldsymbol{a}') \ge hr(\boldsymbol{x}, \boldsymbol{a}) + (1 - \gamma h)(Q^{h,\star}(\boldsymbol{x}', \boldsymbol{a}') - \varepsilon),$$

which implies that

$$Q^{h,\star}(\boldsymbol{x},\boldsymbol{a}) \geq Q^{h,\tilde{c}}(\boldsymbol{x},\boldsymbol{a}) \geq hr(\boldsymbol{x},\boldsymbol{a}) + (1-\gamma h)(Q^{h,\star}(\boldsymbol{x}',\boldsymbol{a}') - \varepsilon).$$

Taking the supremum of both sides with respect to $\boldsymbol{b} \in \mathbb{R}^m$ such that $|\boldsymbol{b}| \leq L$ yields

$$Q^{h,\star}(\boldsymbol{x},\boldsymbol{a}) \geq hr(\boldsymbol{x},\boldsymbol{a}) - (1-\gamma h)\varepsilon + (1-\gamma h)\sup_{|\boldsymbol{b}|\leq L}\left\{Q^{h,\star}(\xi(\boldsymbol{x},\boldsymbol{a};h),\boldsymbol{a}+h\boldsymbol{b})\right\}.$$

Since ε was arbitrarily chosen, we finally obtain that

$$Q^{h,\star}(\boldsymbol{x},\boldsymbol{a}) \geq hr(\boldsymbol{x},\boldsymbol{a}) + (1-\gamma h)\sup_{|\boldsymbol{b}|\leq L}\left\{Q^{h,\star}(\xi(\boldsymbol{x},\boldsymbol{a};h),\boldsymbol{a}+h\boldsymbol{b})\right\}. \tag{B.3}$$

Combining two estimates (B.2) and (B.3), we conclude that $Q^{h,\star}$ satisfies the semi-discrete HJB equation (3.2). Since the proof for the uniqueness of the solution is almost the same as the proof of Theorem 4.2, Chapter VI in [46], we have omitted the detailed proof. □

B.4 Proposition 3

Proof. For the completeness of the paper, we provide a sketch of the proof although it is similar to the proof of Theorem 1.1, Chapter VI in [46]. We begin by defining two functions $\underline{Q}^\star$ and $\overline{Q}^\star$ as

$$\underline{Q}^\star(\boldsymbol{x},\boldsymbol{a}) := \liminf_{(\boldsymbol{x}',\boldsymbol{a}',h)\to(\boldsymbol{x},\boldsymbol{a},0+)} Q^{h,\star}(\boldsymbol{x}',\boldsymbol{a}'),$$

$$\overline{Q}^\star(\boldsymbol{x},\boldsymbol{a}) := \limsup_{(\boldsymbol{x}',\boldsymbol{a}',h)\to(\boldsymbol{x},\boldsymbol{a},0+)} Q^{h,\star}(\boldsymbol{x}',\boldsymbol{a}').$$

According to the proof of Theorem 1.1, Chapter VI in [46], it suffices to show that $\overline{Q}^\star$ satisfies the first condition of Definition 1 and $\underline{Q}^\star$ satisfies the second condition of Definition 1. To this end, for any $\phi \in C^1$, let $(\boldsymbol{x}_0,\boldsymbol{a}_0)$ be a strict local maximum point of $\overline{Q}^\star - \phi$ and choose a small enough neighborhood $\mathcal{N}$ of $(\boldsymbol{x}_0,\boldsymbol{a}_0)$ such that $(\overline{Q}^\star - \phi)(\boldsymbol{x}_0,\boldsymbol{a}_0) = \max_{\mathcal{N}}(\overline{Q}^\star - \phi)$. Then, there exists a sequence $\{(\boldsymbol{x}_n,\boldsymbol{a}_n,h_n)\}$ with $(\boldsymbol{x}_n,\boldsymbol{a}_n) \to (\boldsymbol{x}_0,\boldsymbol{a}_0)$ and $h_n \to 0+$ such that

$$(Q^{h_n,\star} - \phi)(\boldsymbol{x}_n,\boldsymbol{a}_n) = \max_{\mathcal{N}}(Q^{h_n,\star} - \phi)$$

and

$$Q^{h_n,\star}(\boldsymbol{x}_n,\boldsymbol{a}_n) \to \overline{Q}^\star(\boldsymbol{x}_0,\boldsymbol{a}_0).$$

Recall that $Q^{h,\star}$ satisfies (3.2). Thus, there exists $\boldsymbol{b}_n$ with $|\boldsymbol{b}_n| \leq L$ such that

$$Q^{h_n,\star}(\boldsymbol{x}_n,\boldsymbol{a}_n) - h_n r(\boldsymbol{x}_n,\boldsymbol{a}_n) - (1-\gamma h_n)Q^{h_n,\star}(\xi(\boldsymbol{x}_n,\boldsymbol{a}_n;h_n),\boldsymbol{a}_n + h\boldsymbol{b}_n) = 0.$$

Since $Q^{h_n,\star} - \phi$ attains a local maximum at $(\boldsymbol{x}_n,\boldsymbol{a}_n)$, we have

$$(1-\gamma h_n)(\phi(\boldsymbol{x}_n,\boldsymbol{a}_n) - \phi(\xi(\boldsymbol{x}_n,\boldsymbol{a}_n;h_n),\boldsymbol{a}_n + h\boldsymbol{b}_n) + \gamma h_n Q^{h_n,\star}(\boldsymbol{x}_n,\boldsymbol{a}_n) - h_n r(\boldsymbol{x}_n,\boldsymbol{a}_n) \leq 0 \tag{B.4}$$

for small enough $h_n > 0$. Since $|\boldsymbol{b}_n| \leq L$ for all $n \geq 0$, there exists a subsequence n_k and $\boldsymbol{b}$ with $|\boldsymbol{b}| \leq L$ such that $\boldsymbol{b}_{n_k} \to \boldsymbol{b}$ as $k \to \infty$. Then, we substitute n in (B.4) by n_k, divide both sides by h_{n_k} and let $k \to \infty$ to obtain that at $(\boldsymbol{x}_0,\boldsymbol{a}_0)$

$$-\nabla_{\boldsymbol{x}}\phi \cdot f - \nabla_{\boldsymbol{a}}\phi \cdot \boldsymbol{b} + \gamma\overline{Q}^\star - r \leq 0,$$

where we use the fact that

$$\lim_{h\to 0}\frac{\xi(\boldsymbol{x},\boldsymbol{a};h) - \boldsymbol{x}}{h} = f(\boldsymbol{x},\boldsymbol{a}).$$

This implies that the first condition of Definition 1 is satisfied. Similarly, it can be shown that $\underline{Q}^\star$ satisfies the second condition of Definition 1. □

B.5 Theorem 2

We begin by defining an optimal Bellman operator in the semi-discrete setting, $\mathcal{T}^h : L^\infty \to L^\infty$, by

$$(\mathcal{T}^h Q)(\boldsymbol{x}, \boldsymbol{a}) := hr(\boldsymbol{x}, \boldsymbol{a}) + (1 - \gamma h) \sup_{|\boldsymbol{b}| \le L} Q(\xi(\boldsymbol{x}, \boldsymbol{a}; h), \boldsymbol{a} + h\boldsymbol{b}), \tag{B.5}$$

where $\xi(\boldsymbol{x}, \boldsymbol{a}; h)$ denotes the solution of the ODE (2.1) at time $t = h$ with initial state $x(0) = \boldsymbol{x}$ and constant action $a(t) \equiv \boldsymbol{a}$ for $t \in [0, h)$. Our first observation is that the Bellman operator is a monotone $(1 - \gamma h)$-contraction mapping for a sufficiently small h.

Lemma 1. *Suppose that $0 < h < \frac{1}{\gamma}$. Then, the Bellman operator $\mathcal{T}^h$ is a monotone contraction mapping. More precisely, it satisfies the following properties:*

(i) $\mathcal{T}^h Q \le \mathcal{T}^h Q'$ for all $Q, Q' \in L^\infty$ such that $Q \le Q'$;

(ii) $\|\mathcal{T}^h Q - \mathcal{T}^h Q'\|_{L^\infty} \le (1 - \gamma h)\|Q - Q'\|_{L^\infty}$ for all $Q, Q' \in L^\infty$.

Proof. (i) Since $Q(\boldsymbol{x}, \boldsymbol{a}) \le Q'(\boldsymbol{x}, \boldsymbol{a})$ for all $(\boldsymbol{x}, \boldsymbol{a}) \in \mathbb{R}^n \times \mathbb{R}^m$, we have

$$\sup_{|\boldsymbol{b}| \le L} Q(\xi(\boldsymbol{x}, \boldsymbol{a}; h), \boldsymbol{a} + h\boldsymbol{b}) \le \sup_{|\boldsymbol{b}| \le L} Q'(\xi(\boldsymbol{x}, \boldsymbol{a}; h), \boldsymbol{a} + h\boldsymbol{b}).$$

Multiplying $(1 - \gamma h)$ and then adding $hr(\boldsymbol{x}, \boldsymbol{a})$ to both sides, we confirm the monotonicity of $\mathcal{T}^h$ as desired.

(ii) We first note that for any $\boldsymbol{b} \in \mathbb{R}^m$ with $|\boldsymbol{b}| \le L$,

$$\begin{aligned}
&\big[hr(\boldsymbol{x}, \boldsymbol{a}) + (1 - \gamma h) Q(\xi(\boldsymbol{x}, \boldsymbol{a}; h), \boldsymbol{a} + h\boldsymbol{b})\big] - \big[hr(\boldsymbol{x}, \boldsymbol{a}) + (1 - \gamma h) Q'(\xi(\boldsymbol{x}, \boldsymbol{a}; h), \boldsymbol{a} + h\boldsymbol{b})\big] \\
&= (1 - \gamma h)\big[Q(\xi(\boldsymbol{x}, \boldsymbol{a}; h), \boldsymbol{a} + h\boldsymbol{b}) - Q'(\xi(\boldsymbol{x}, \boldsymbol{a}; h), \boldsymbol{a} + h\boldsymbol{b})\big] \\
&\le (1 - \gamma h)\|Q - Q'\|_{L^\infty}.
\end{aligned}$$

By the definition of $\mathcal{T}^h Q'$, we have

$$\begin{aligned}
&hr(\boldsymbol{x}, \boldsymbol{a}) + (1 - \gamma h) Q(\xi(\boldsymbol{x}, \boldsymbol{a}; h), \boldsymbol{a} + h\boldsymbol{b}) \\
&\le (1 - \gamma h)\|Q - Q'\|_{L^\infty} + hr(\boldsymbol{x}, \boldsymbol{a}) + (1 - \gamma h) Q'(\xi(\boldsymbol{x}, \boldsymbol{a}; h), \boldsymbol{a} + h\boldsymbol{b}) \\
&\le (1 - \gamma h)\|Q - Q'\|_{L^\infty} + \mathcal{T}^h Q'(\boldsymbol{x}, \boldsymbol{a}).
\end{aligned}$$

Taking the supremum of both sides with respect to $\boldsymbol{b} \in \mathbb{R}^m$ such that $|\boldsymbol{b}| \le L$, yields

$$\mathcal{T}^h Q(\boldsymbol{x}, \boldsymbol{a}) \le (1 - \gamma h)\|Q - Q'\|_{L^\infty} + \mathcal{T}^h Q'(\boldsymbol{x}, \boldsymbol{a}),$$

or equivalently

$$\mathcal{T}^h Q(\boldsymbol{x}, \boldsymbol{a}) - \mathcal{T}^h Q'(\boldsymbol{x}, \boldsymbol{a}) \le (1 - \gamma h)\|Q - Q'\|_{L^\infty}.$$

We now change the role of Q and Q' to obtain

$$|\mathcal{T}^h Q(\boldsymbol{x}, \boldsymbol{a}) - \mathcal{T}^h Q'(\boldsymbol{x}, \boldsymbol{a})| \le (1 - \gamma h)\|Q - Q'\|_{L^\infty}.$$

Therefore, the operator $\mathcal{T}^h$ is a $(1 - \gamma h)$-contraction with respect to $\|\cdot\|_{L^\infty}$. □

Using the Bellman operator $\mathcal{T}^h$, HJ Q-learning (3.3) can be expressed as

$$Q_{k+1}^h := (1-\alpha_k)Q_k^h + \alpha_k \mathcal{T}^h Q_k^h.$$

Consider the difference $\Delta_k^h := Q_k^h - Q^{h,\star}$. Note that $\|\Delta_k^h\|_{L^\infty}$ represents the optimality gap at the kth iteration. It satisfies

$$\Delta_{k+1}^h = (1-\alpha_k)\Delta_k^h + \alpha_k[\mathcal{T}^h(\Delta_k^h + Q^{h,\star}) - \mathcal{T}^h Q^{h,\star}], \tag{B.6}$$

where we used the semi-discrete HJB equation $Q^{h,\star} = \mathcal{T}^h Q^{h,\star}$. The contraction property of the Bellman operator $\mathcal{T}^h$ can be used to show that the optimality gap $\|\Delta_k^h\|_{L^\infty}$ decreases geometrically. More precisely, we have the following lemma:

Lemma 2. *Suppose that* $0 < h < \frac{1}{\gamma}$, $0 \le \alpha_k \le 1$ *and that Assumption 1 holds. Then, the following inequality holds:*

$$\|\Delta_k^h\|_{L^\infty} \le \left(\prod_{\tau=0}^{k-1}(1-\alpha_\tau\gamma h)\right)\|\Delta_0^h\|_{L^\infty}.$$

Proof. We use mathematical induction to prove the assertion. When $k=1$, it follows from the Q-function update (3.3) and the contraction property of $\mathcal{T}^h$ that

$$\begin{aligned}
\|\Delta_1^h\|_{L^\infty} &\le (1-\alpha_0)\|\Delta_0^h\|_{L^\infty} + \alpha_0\|\mathcal{T}^h(\Delta_0^h + Q^{h,\star}) - \mathcal{T}^h Q^{h,\star}\|_{L^\infty} \\
&\le (1-\alpha_0)\|\Delta_0^h\|_{L^\infty} + \alpha_0(1-\gamma h)\|\Delta_0^h\|_{L^\infty} \\
&= (1-\alpha_0\gamma h)\|\Delta_0^h\|_{L^\infty}.
\end{aligned}$$

Therefore, the assertion holds for $k=1$. We now assume that the assertion holds for $k=n$:

$$\|\Delta_n^h\|_{L^\infty} \le \left(\prod_{\tau=0}^{n-1}(1-\alpha_\tau\gamma h)\right)\|\Delta_0^h\|_{L^\infty}.$$

We need to show that the inequality holds for $k=n+1$. By using the same estimate as in the case of $k=1$ and the induction hypothesis for $k=n$, we obtain

$$\begin{aligned}
\|\Delta_{n+1}^h\|_{L^\infty} &\le (1-\alpha_n)\|\Delta_n^h\|_{L^\infty} + \alpha_n\|\mathcal{T}^h(\Delta_n^h + Q^{h,\star}) - \mathcal{T}^h Q^{h,\star}\|_{L^\infty} \\
&\le (1-\alpha_n)\|\Delta_n^h\|_{L^\infty} + \alpha_n(1-\gamma h)\|\Delta_n^h\|_{L^\infty} \\
&= (1-\alpha_n\gamma h)\|\Delta_n^h\|_{L^\infty} \\
&\le (1-\alpha_n\gamma h)\left(\prod_{\tau=0}^{n-1}(1-\alpha_\tau\gamma h)\right)\|\Delta_0^h\|_{L^\infty} \\
&= \left(\prod_{\tau=0}^{n}(1-\alpha_\tau\gamma h)\right)\|\Delta_0^h\|_{L^\infty}.
\end{aligned}$$

This completes our mathematical induction, and thus the result follows. □

This lemma yields a condition on the sequence of learning rates under which the Q-function updated by (3.3) converges to the optimal semi-discrete Q-function (3.1) in L^∞.

Proof. It suffices to show that

$$\lim_{k\to\infty} \|\Delta_k^h\|_{L^\infty} = 0.$$

By Lemma 2 and the elementary inequality $1 - x \le e^{-x}$, we have

$$\|\Delta_k^h\|_{L^\infty} \le \left(\prod_{\tau=0}^{k-1}(1-\alpha_\tau\gamma h)\right)\|\Delta_0^h\|_{L^\infty} \le \exp\left(-\gamma h\left(\sum_{\tau=0}^{k-1}\alpha_\tau\right)\right)\|\Delta_0^h\|_{L^\infty}.$$

Therefore, if $\sum_{\tau=0}^{\infty}\alpha_\tau = \infty$, the result follows. $\square$

B.6 Corollary 1

Proof. We first observe that there exists an index k_h, depending on h, such that $\sum_{\tau=0}^{k_h-1}\alpha_\tau > \frac{1}{h^2}$ since $\sum_{\tau=0}^{\infty}\alpha_\tau = \infty$. Then, we have

$$h\left(\sum_{\tau=0}^{k_h-1}\alpha_\tau\right) > \frac{1}{h} \to \infty \quad \text{as} \quad h \to 0.$$

Moreover, by the triangle inequality, we have

$$|Q_k^h(\boldsymbol{x},\boldsymbol{a}) - Q(\boldsymbol{x},\boldsymbol{a})| \le |Q_k^h(\boldsymbol{x},\boldsymbol{a}) - Q^{h,\star}(\boldsymbol{x},\boldsymbol{a})| + |Q^{h,\star}(\boldsymbol{x},\boldsymbol{a}) - Q(\boldsymbol{x},\boldsymbol{a})|$$

for all $(\boldsymbol{x},\boldsymbol{a}) \in \mathbb{R}^n \times \mathbb{R}^m$. By Proposition 2, the second term on the right-hand side uniformly vanishes over any compact subset K of $\mathbb{R}^n \times \mathbb{R}^m$ as $h \to 0$. The first term is nothing but $|\Delta_k^h(\boldsymbol{x},\boldsymbol{a})|$, which is bounded as follows (by Lemma 2):

$$|\Delta_k^h(\boldsymbol{x},\boldsymbol{a})| \le \left(\prod_{\tau=0}^{k-1}(1-\alpha_\tau\gamma h)\right)\|\Delta_0^h\|_{L^\infty} \le \exp\left(-\gamma h\left(\sum_{\tau=0}^{k-1}\alpha_\tau\right)\right)\|\Delta_0^h\|_{L^\infty}, \quad k \ge 1,$$

where the second inequality holds because $1 - x \le e^{-x}$. Our choice of k_h then yields

$$\sup_{k\ge k_h}\|\Delta_k^h\|_{L^\infty} \le \exp\left(-\gamma h\left(\sum_{\tau=0}^{k_h-1}\alpha_\tau\right)\right)\|\Delta_0^h\|_{L^\infty} \to 0$$

as $h \to 0$. Therefore, we conclude that

$$\begin{aligned}
&\sup_{k\ge k_h}\sup_{\substack{(\boldsymbol{x},\boldsymbol{a})\in K\\ K\text{compact}}}|Q_k^h(\boldsymbol{x},\boldsymbol{a}) - Q(\boldsymbol{x},\boldsymbol{a})| \\
&\le \sup_{k\ge k_h}\sup_{\substack{(\boldsymbol{x},\boldsymbol{a})\in K\\ K\text{compact}}}|Q_k^h(\boldsymbol{x},\boldsymbol{a}) - Q^{h,\star}(\boldsymbol{x},\boldsymbol{a})| + \sup_{k\ge k_h}\sup_{\substack{(\boldsymbol{x},\boldsymbol{a})\in K\\ K\text{compact}}}|Q^{h,\star}(\boldsymbol{x},\boldsymbol{a}) - Q(\boldsymbol{x},\boldsymbol{a})| \\
&\le \sup_{k\ge k_h}\|\Delta_k^h\|_{L^\infty} + \sup_{\substack{(\boldsymbol{x},\boldsymbol{a})\in K\\ K\text{compact}}}|Q^{h,\star}(\boldsymbol{x},\boldsymbol{a}) - Q(\boldsymbol{x},\boldsymbol{a})| \to 0
\end{aligned}$$

as $h \to 0$. $\square$

B.7 Proposition 4

Proof. We first notice that by the triangle inequality,

$$\begin{aligned}
&\left|\max_{|\boldsymbol{a}-a_j|\le hL} Q_{\theta^-}(x_{j+1},\boldsymbol{a}) - Q_{\theta^-}(x_{j+1},a_j+hb_j)\right| \\
&\le \left|\max_{|\boldsymbol{a}-a_j|\le hL} Q_{\theta^-}(x_{j+1},\boldsymbol{a}) - Q_{\theta^-}\left(x_{j+1},a_j+hL\frac{\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j)}{|\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j)|}\right)\right| \\
&\quad + \left|Q_{\theta^-}\left(x_{j+1},a_j+hL\frac{\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j)}{|\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j)|}\right) - Q_{\theta^-}(x_{j+1},a_j+hb_j)\right| \\
&=: \Delta_1+\Delta_2.
\end{aligned}$$

We first consider Δ_1. Let $\boldsymbol{a}^\star := \arg\max_{|\boldsymbol{a}-a_j|\le hL} Q_{\theta^-}(x_{j+1},\boldsymbol{a})$. By the Taylor expansion, we have

$$\begin{aligned}
\max_{|\boldsymbol{a}-a_j|\le hL} Q_{\theta^-}(x_{j+1},\boldsymbol{a}) &= Q_{\theta^-}(x_{j+1},\boldsymbol{a}^\star) \\
&= Q_{\theta^-}(x_{j+1},a_j) + \nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j)\cdot(\boldsymbol{a}^\star-a_j) + O(h^2).
\end{aligned}$$

Similarly, we again use the Taylor expansion to obtain that

$$Q_{\theta^-}\left(x_{j+1},a_j+hL\frac{\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j)}{|\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j)|}\right) = Q_{\theta^-}(x_{j+1},a_j) + hL|\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j)| + O(h^2).$$

Subtracting one equality from another yields

$$\begin{aligned}
&Q_{\theta^-}(x_{j+1},\boldsymbol{a}^\star) - Q_{\theta^-}\left(x_{j+1},a_j+hL\frac{\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j)}{|\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j)|}\right) \\
&= \nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j)\cdot(\boldsymbol{a}^\star-a_j) - hL|\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j)| + O(h^2) \le O(h^2),
\end{aligned}$$

where the last inequality holds because $|\boldsymbol{a}^\star - a_j| \le hL$. Since our choice of $\boldsymbol{a}^\star$ implies that the left-hand side of the inequality above is always non-negative, we conclude that $\Delta_1 = O(h^2)$.

Regarding Δ_2, we have

$$\begin{aligned}
\Delta_2 &= \left|Q_{\theta^-}\left(x_{j+1},a_j+hL\frac{\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j)}{|\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j)|}\right) - Q_{\theta^-}\left(x_{j+1},a_j+hL\frac{\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_j,a_j)}{|\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_j,a_j)|}\right)\right| \\
&\le Lh\|\nabla_{\boldsymbol{a}}Q_{\theta^-}\|_{L^\infty}\left|\frac{\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j)}{|\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j)|} - \frac{\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_j,a_j)}{|\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_j,a_j)|}\right|.
\end{aligned}$$

Note that for any two non-zero vectors v, w,

$$\left|\frac{v}{|v|}-\frac{w}{|w|}\right| \le \frac{|v-w|}{|v|} + |w|\left(\frac{1}{|v|}-\frac{1}{|w|}\right) = \frac{|v-w|}{|v|} + \frac{|w|-|v|}{|v|} \le \frac{2|v-w|}{|v|}.$$

On the other hand, we have

$$|\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j) - \nabla_{\boldsymbol{a}}Q_{\theta^-}(x_j,a_j)| \le \|\nabla^2_{\boldsymbol{xa}}Q_{\theta^-}\|_{L^\infty}|x_{j+1}-x_j| = O(h).$$

Since we assume that Q_{θ^-} is twice differentiable and $|\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_j,a_j)| =: C > 0$, we have $|\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j)| > C/2$ for sufficiently small h. Therefore, we obtain that

$$\Delta_2 \le 2Lh\|\nabla_{\boldsymbol{a}}Q_{\theta^-}\|_{L^\infty}\frac{|\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j) - \nabla_{\boldsymbol{a}}Q_{\theta^-}(x_j,a_j)|}{|\nabla_{\boldsymbol{a}}Q_{\theta^-}(x_{j+1},a_j)|} = O(h^2).$$

Combining the estimates of Δ_1 and Δ_2 yields

$$\left| \max_{|\boldsymbol{a}-a_j|\le hL} Q_{\theta^-}(x_{j+1},\boldsymbol{a}) - Q_{\theta^-}(x_{j+1}, a_j + hb_j) \right| = O(h^2)$$

as desired. □

C Brief Discussion on Extension to Stochastic Systems

The Hamilton-Jacobi Q-learning can be extended to the continuous-time stochastic control setting with controlled diffusion processes. Consider the following stochastic counterpart of the system (2.1):

$$\mathrm{d}x_t = f(x_t,a_t)\mathrm{d}t + \sigma(x_t,a_t)\mathrm{d}W_t, \quad t>0, \tag{C.1}$$

where $\sigma : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^{n\times k}$ is the diffusion coefficient and W_t is the k-dimensional standard Bronwian motion. We now define the Q-function as

$$Q(\boldsymbol{x},\boldsymbol{a}) := \sup_{a\in\mathcal{A}} \mathbb{E}\left[\int_0^\infty e^{-\gamma t} r(x_t,a_t)\mathrm{d}t \;\mid\; x_0=\boldsymbol{x}, a_0=\boldsymbol{a}\right].$$

Again, the dynamic programming principle implies

$$\begin{aligned} 0 = \sup_{a\in\mathcal{A}} \mathbb{E}\Bigg[&\frac{1}{h}\int_t^{t+h} e^{-\gamma(s-t)} r(x(s),a(s))\,\mathrm{d}s + \frac{1}{h}[Q(x(t+h),a(t+h)) - Q(\boldsymbol{x},\boldsymbol{a})] \\ &+ \frac{e^{-\gamma h}-1}{h} Q(x(t+h),a(t+h)) \mid x(t)=\boldsymbol{x}, a(t)=\boldsymbol{a}\Bigg]. \end{aligned} \tag{C.2}$$

Then, we use the Itô formula

$$\begin{aligned} \mathrm{d}Q(x_t,a_t) &= \nabla_{\boldsymbol{x}} Q\cdot \mathrm{d}x_t + \nabla_{\boldsymbol{a}} Q \cdot \dot{a}\mathrm{d}t + \frac{1}{2}\mathrm{d}x_t^\top \nabla_{\boldsymbol{x}}^2 Q \mathrm{d}x_t \\ &= \nabla_{\boldsymbol{x}} Q \cdot (f(x_t,a_t)\mathrm{d}t + \sigma(x_t,a_t)\mathrm{d}W_t) + \nabla_{\boldsymbol{a}} Q\cdot \dot{a}\mathrm{d}t + \frac{(\mathrm{d}W_t)^\top \sigma^\top \nabla_{\boldsymbol{x}}^2 Q \sigma \mathrm{d}W_t}{2} \end{aligned}$$

to derive the following Hamilton-Jacobi-Bellman equation for the stochastic system (C.1):

$$\gamma Q - \nabla_{\boldsymbol{x}} Q\cdot f(\boldsymbol{x},\boldsymbol{a}) - L|\nabla_{\boldsymbol{a}} Q| - r(\boldsymbol{x},\boldsymbol{a}) - \frac{\mathrm{tr}(\sigma^\top \nabla_{\boldsymbol{x}}^2 Q \sigma)}{2} = 0. \tag{C.3}$$

Note that, in this case also, the optimal control satisfies $\dot{a} = L\frac{\nabla_{\boldsymbol{a}} Q}{|\nabla_{\boldsymbol{a}} Q|}$ when Q is differentiable.

Since in most practical systems transition samples are collected in discrete time, we also introduce the semi-discrete version of (C.3). We define a stochastic semi-discrete Q-function $Q^{h,\star}$ as

$$Q^{h,\star}(\boldsymbol{x},\boldsymbol{a}) := \sup_{b\in\mathcal{B}} \mathbb{E}\left[h\sum_{k=0}^\infty r(x_k,a_k)(1-\gamma h)^k\right],$$

where $\mathcal{B} := \{b := \{b_k\}_{k=0}^\infty \mid b_k\in\mathbb{R}^m, |b_k|\le L\}$, $x_{k+1} = \xi(x_k,a_k;h)$ and $a_{k+1} = a_k + hb_k$. Here, $\xi(x_k,a_k;h)$ is now a solution to the stochastic differential equation (C.1) at time $t=h$ with initial state $\boldsymbol{x}$ and constant control $a(t)\equiv \boldsymbol{a}$, $t\in[0,h)$. Then, similar to the deterministic semi-discrete HJB equation (3.2), its stochastic counterpart can be written as follows:

$$Q^{h,\star}(\boldsymbol{x},\boldsymbol{a}) = hr(\boldsymbol{x},\boldsymbol{a}) + (1-\gamma h)\sup_{|\boldsymbol{b}|\le L} \mathbb{E}\left[Q^{h,\star}(\xi(\boldsymbol{x},\boldsymbol{a};h),\boldsymbol{a}+h\boldsymbol{b})\right].$$

Table 1: Hyperparameters for HJ DQN.

Hyperparameter	HalfCheetah-v2	Hopper-v2	Walker2d-v2	Swimmer-v2	LQ
optimizer	Adam [57]	Adam [57]	Adam [57]	Adam [57]	Adam [57]
learning rate	5×10^{-4}	10^{-4}	10^{-4}	5×10^{-4}	10^{-3}
Lipschitz constant (L)	30	30	30	15	10
default sampling interval (h)	0.05	0.008	0.008	0.04	0.05
tuned sampling interval (h)	0.01	0.016	0.032	0.08	-
(Continuous) discount (γ)	$-\log(0.99)/h$, where h is the sampling interval	$-\log(0.99)/h$, where h is the sampling interval	$-\log(0.99)/h$, where h is the sampling interval	$-\log(0.99)/h$	$-\log(0.99999)/h$
replay buffer size	10^6	10^6	10^6	10^6	2×10^4
target smoothing coefficient (α)	0.001	0.001	0.001	0.001	0.001
Noise coefficient (σ)	0.1	0.1	0.1	0.1	0.1
number of hidden layers	2 (fully connected)	2 (fully connected)	2 (fully connected)	2 (fully connected)	2 (fully connected)
number of hidden units per layer	256	256	256	256	256
number of samples per minibatch	128	128	128	128	512
nonlinearity	ReLU	ReLU	ReLU	ReLU	ReLU

Using Robbins-Monro stochastic approximation [55,56], we obtain the following model-free update rule: in the kth iteration, we collect data (x_k, a_k, r_k, x_{k+1}) and update the Q-function by

$$Q_{k+1}^h(x_k, a_k) := (1-\alpha_k) Q_k^h(x_k, a_k) + \alpha_k \Big[h r_k + (1-\gamma h) \sup_{|\boldsymbol{b}| \le L} Q_k^h(x_{k+1}, a_k + h\boldsymbol{b}) \Big], \tag{C.4}$$

where x_{k+1} is obtained by simulating the stochastic system from x_k with action a_k fixed for h period, i.e., $x_{k+1} = \xi(x_k, a_k; h)$. The corresponding HJ DQN algorithm for stochastic systems is essentially the same as Algorithm 1 although the transition samples are now collected through the stochastic system.

D Implementation Details

All the simulations in Section 5 were conducted using Python 3.7.4 on a PC with Intel Core i9-9900X @ 3.50GHz, NVIDIA GeForce RTX 2080 Ti and 64GB RAM.

Table 2: Hyperparameters for DDPG.

Hyperparameter	MuJoCo tasks	LQ
optimizer	Adam [57]	
actor learning rate	10^{-4}	
critic learning rate	10^{-3}	
(Discrete) discount (γ')	0.99	0.99999
replay buffer size	10^6	2×10^4
target smoothing coefficient (α)	0.001	
number of hidden layers	2 (fully connected)	
number of hidden units per layer	256	
number of samples per minibatch	128	512
nonlinearity	ReLU	

Table 1 shows the list of hyperparameters that are used in our implementation of HJ DQN for each MuJoCo task and the LQ problem. For DDPG, we list our choice of hyperparameters in Table 2, which are taken from [16] for MuJoCo tasks, except the network architecture which is used in OpenAI's implementation of DDPG[8]. The discount factor in the discrete-time algorithms is chosen as $\gamma' = 0.99$ for MuJoCo tasks and 0.99999 for the LQ problem so that it is equivalent to $e^{-\gamma h} \approx (1 - \gamma h)$ in our algorithm for continuous-time systems.

References

[1] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming.* Belmont, MA: Athena Scientific, 1996.

[2] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction.* Cambridge, MA: MIT Press, 1998.

[3] C. Szepesvari, *Algorithms for Reinforcement Learning.* San Rafael, CA: Morgan and Claypool Publishers, 2010.

[4] L. Ljung, *System Identification: Theory for the User*, 2nd ed. Pearson, 1998.

[5] C. J. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, pp. 279–292, 1992.

[6] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, and S. Petersen, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, 2015.

[7] K. Doya, "Reinforcement learning in continuous time and space," *Neural Computation*, vol. 12, pp. 219–245, 2000.

[8] G. J. Gordon, "Stable function approximation in dynamic programming," in *International Conference on Machine Learning*, 1995, pp. 261–268.

[9] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033.

[8] https://github.com/openai/spinningup

[10] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "OpenAI gym," *arXiv preprint arXiv:1606.01540*, 2016.

[11] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," in *International Conference on Machine Learning*, 2016, pp. 1329–1338.

[12] Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. L. Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq, and T. Lillicrap, "DeepMind control suite," *arXiv preprint arXiv:1801.00690*, 2018.

[13] M. Ryu, Y. Chow, R. Anderson, C. Tjandraatmadja, and C. Boutilier, "CAQL: Continuous action Q-learning," *arXiv preprint arXiv:1909.12397*, 2020.

[14] J. Schulman, S. Levine, P. Moritz, M. Jordan, and P. Abbeel, "Trust region policy optimization," in *International Conference on Machine Learning*, 2015, pp. 1889–1897.

[15] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[16] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[17] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International Conference on Machine Learning*, 2016, pp. 1928–1937.

[18] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International Conference on Machine Learning*, 2018, pp. 1861–1870.

[19] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International Conference on Machine Learning*, 2018, pp. 1587–1596.

[20] C. Tessler, G. Tennenholtz, and S. Mannor, "Distributional policy optimization: An alternative approach for continuous control," in *Advances in Neural Information Processing Systems*, 2019, pp. 1350–1360.

[21] S. Gu, T. Lillicrap, I. Sutskever, and S. Levine, "Continuous deep Q-learning with model-based acceleration," in *International Conference on Machine Learning*, 2016, pp. 2829–2838.

[22] M. Palanisamy, H. Modares, F. L. Lewis, and M. Aurangzeb, "Continuous-time Q-learning for infinite-horizon discounted cost linear quadratic regulator problems," *IEEE Transactions on Cybernetics*, vol. 45, pp. 165–176, 2015.

[23] T. Bian and Z.-P. Jiang, "Value iteration and adaptive dynamic programming for data-driven adaptive optimal control design," *Automatica*, vol. 71, pp. 348–360, 2016.

[24] K. G. Vamvoudakis, "Q-learning for continuous-time linear systems: A model-free infinite horizon optimal control approach," *Systems & Control Letters*, vol. 100, pp. 14–20, 2017.

[25] Y. Jiang and Z.-P. Jiang, "Global adaptive dynamic programming for continuous-time nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 60, pp. 2917–2929, 2015.

[26] J. Kim and I. Yang, "Hamilton–Jacobi–Bellman equations for maximum entropy optimal control," *arXiv preprint arXiv:2009.13097*, 2020.

[27] S. Bhasin, R. Kamalapurkar, M. Johnson, K. G. Vamvoudakis, F. L. Lewis, and W. E. Dixon, "A novel actor-critic-identifier architecture for approximate optimal control of uncertain nonlinear systems," *Automatica*, vol. 49, pp. 82–92, 2013.

[28] H. Modares and F. L. Lewis, "Optimal tracking control of nonlinear partially-unknown constrained-input systems using integral reinforcement learning," *Automatica*, vol. 50, pp. 1780–1792, 2014.

[29] K. G. Vamvoudakis and F. Lewis, "Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem," *Automatica*, vol. 46, pp. 878–888, 2010.

[30] S. J. Bradtke and M. O. Duff, "Reinforcement learning methods for continuous-time Markov decision problems," in *Advances in Neural Information Processing Systems*, 1995, pp. 393–400.

[31] R. Munos, "Policy gradient in continuous time," *Journal of Machine Learning Research*, vol. 7, pp. 771–791, 2006.

[32] ——, "A study of reinforcement learning in the continuous case by the means of viscosity solutions," *Machine Learning*, vol. 40, pp. 265–299, 2000.

[33] R. Munos and A. W. Moore, "Barycentric interpolators for continuous space and time reinforcement learning," in *Advances in Neural Information Processing Systems*, 1999, pp. 1024–1030.

[34] P. Dayan and S. P. Singh, "Improving policies without measuring merits," in *Advances in Neural Information Processing Systems*, 1996, pp. 1059–1065.

[35] M. Ohnishi, M. Yukawa, M. Johansson, and M. Sugiyama, "Continuous-time value function approximation in reproducing kernel Hilbert spaces," in *Advances in Neural Information Processing Systems*, 2018, pp. 2813–2824.

[36] Y. Yang, D. Wunsch, and Y. Yin, "Hamiltonian-driven adaptive dynamic programming for continuous nonlinear dynamical systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, pp. 1929–1940, 2017.

[37] E. Theodorou, J. Buchli, and S. Schaal, "A generalized path integral control approach to reinforcement learning," *Journal of Machine Learning Research*, vol. 11, pp. 3137–3181, 2010.

[38] K. Rajagopal, S. N. Balakrishnan, and J. R. Busemeyer, "Neural network-based solutions for stochastic optimal control using path integrals," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, pp. 534–545, 2017.

[39] Y. Tassa and T. Erez, "Least squares solutions of the HJB equation with neural network value-function approximators," *IEEE Transactions on Neural Networks*, vol. 18, pp. 1031–1041, 2007.

[40] M. Lutter, B. Belousov, K. Listmann, D. Clever, and J. Peters, "HJB optimal feedback control with deep differential value functions and action constraints," in *Conference on Robot Learning*, 2020, pp. 640–650.

[41] G. P. Kontoudis and K. G. Vamvoudakis, "Kinodynamic motion planning with continuous-time Q-learning: An online, model-free, and safe navigation framework," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, pp. 3803–3817, 2019.

[42] P. Mehta and S. Meyn, "Q-learning and pontryagin's minimum principle," in *IEEE Conference on Decision and Control*, 2009, pp. 3598–3605.

[43] C. Tallec, L. Blier, and Y. Ollivier, "Making deep Q-learning methods robust to time discretization," in *International Conference on Machine Learning*, 2019, pp. 6096–6104.

[44] L. C. Baird, "Reinforcement learning in continuous time: advantage updating," in *IEEE International Conference on Neural Networks*, 1994, pp. 2448–2453.

[45] J. Kim and I. Yang, "Hamilton–Jacobi–Bellman for Q-learning in continuous time," in *Learning for Dynamics and Control (L4DC)*, 2020, pp. 739–748.

[46] M. Bardi and I. Capuzzo-Dolcetta, *Optimal Control and Viscosity Solutions of Hamilton–Jacobi–Bellman Equations.* Boston, MA: Birkhäuser, 1997.

[47] M. Crandall and P.-L. Lions, "Viscosity solutions of Hamilton–Jacobi equations," *Transactions of the American Mathematical Society*, vol. 277, pp. 1–42, 1983.

[48] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016, pp. 2094–2100.

[49] N. Kohl and P. Stone, "Policy gradient reinforcement learning for fast quadrupedal locomotion," in *IEEE International Conference on Robotics and Automation*, 2004, pp. 2619–2624.

[50] S. Levine and V. Koltun, "Guided policy search," in *International Conference on Machine Learning*, 2013, pp. 1–9.

[51] M. Fazel, R. Ge, S. M. Kakade, and M. Mesbahi, "Global convergence of policy gradient methods for the linear quadratic regulator," *arXiv preprint arXiv:1801.05039*, 2018.

[52] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *International Conference on Machine Learning*, 2014, pp. 387–395.

[53] D. Quillen, E. Jang, O. Nachum, C. Finn, J. Ibarz, and S. Levine, "Deep reinforcement learning for vision-based robotic grasping: A simulated comparative evaluation of off-policy methods," in *IEEE International Conference on Robotics and Automation*, 2018, pp. 6284–6291.

[54] M. Abu-Khalaf and F. L. Lewis, "Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach," *Automatica*, vol. 41, pp. 779–791, 2005.

[55] H. Robbins and S. Monro, "A stochastic approximation method," *Annals of Mathematical Statistics*, vol. 22, pp. 400–407, 1951.

[56] H. Kushner and G. G. Yin, *Stochastic Approximation and Recursive Algorithms and Applications.* New York: Springer Science & Business Media, 2003.

[57] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representation*, 2015.