Continuous Q-Score Matching: Diffusion Guided Reinforcement Learning for Continuous-Time Control

Chengxiu Hua[1], Jiawen Gu[1*], Yushun Tang[1,2*]

[1]Southern University of Science and Technology, [2]Huawei Technologies Co., Ltd.
12331005@mail.sustech.edu.cn, jwgu.hku@outlook.com, tangys2022@mail.sustech.edu.cn

Abstract

Reinforcement learning (RL) has achieved significant success across a wide range of domains, however, most existing methods are formulated in discrete time. In this work, we introduce a novel RL method for continuous-time control, where stochastic differential equations govern state-action dynamics. Departing from traditional value function-based approaches, our key contribution is the characterization of continuous-time Q-functions via a martingale condition and the linking of diffusion policy scores to the action gradient of a learned continuous Q-function by the dynamic programming principle. This insight motivates Continuous Q-Score Matching (CQSM), a score-based policy improvement algorithm. Notably, our method addresses a long-standing challenge in continuous-time RL: preserving the action-evaluation capability of Q-functions without relying on time discretization. We further provide theoretical closed-form solutions for linear-quadratic (LQ) control problems within our framework. Numerical results in simulated environments demonstrate the effectiveness of our proposed method and compare it to popular baselines.

1 Introduction

RL has achieved substantial success across a wide range of domains over the past decade [44]. Most existing approaches adopt a discrete-time formulation, typically modeled as a Markov Decision Process (MDP) [36, 16], where agents interact with the environment at fixed time intervals. However, many real-world systems—such as autonomous driving in dynamic traffic conditions [47], robotic manipulation [34], and high-frequency algorithmic trading [27]—exhibit continuous, fine-grained dynamics that are inadequately captured by discrete-time models. These applications naturally motivate the need for continuous-time reinforcement learning (CT-RL) frameworks that more faithfully represent the temporal structure of decision-making. Recent works on CT-RL have explored stochastic modeling using stochastic differential equations (SDEs) [11, 21], entropy-regularized exploration techniques [15], and model-free learning methods for diffusion generative models fine-tuning [58, 17] and financial applications [19, 4].

Despite these advances, value-based methods like Q-learning [56]—a cornerstone of discrete-time RL—remain challenging to adapt to the continuous-time setting. Traditional Q-learning algorithms (e.g., SARSA [44], DQN [32]) rely on estimating state-action value functions via temporal-difference (TD) learning and have demonstrated strong performance in discrete action spaces. There is a line of work on discretizing continuous action spaces to apply Q-learning in high-dimensional continuous control settings [49, 40, 20]. Discretizing continuous actions is a common approach to extend Q-learning, but it often struggles with scalability in high-dimensional spaces and relies on

* Corresponding authors.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

discrete-time assumptions. However, extending Q-learning to continuous action spaces introduces major challenges. Early work [13] proposed neural network-based formulations for continuous Q-learning, but later studies [8] reported severe performance drops in high-dimensional settings due to the curse of dimensionality. Moreover, when Q-learning is directly extended to continuous time, the Q-function tends to collapse into an action-independent value function [46], losing its ability to distinguish between actions—a critical property for decision-making. In [25], they show that the discrete Q-learning algorithm is noisier and slower in convergence speed compared with their proposed continuous PG and little q learning algorithms.

To bridge this gap, recent work has explored Q-function-based frameworks that preserve action dependence in continuous time. For example, [10] replaced the Q-function with a generalized Hamiltonian, while [25] proposed little q-learning, a time-discretization-free method based on first-order Q-function approximations, which achieved faster convergence than discrete-time soft Q-learning (SARSA). Score-matching-based methods [35] have also emerged as promising tools for learning diffusion policies, though they still rely on time discretization. Other approaches, such as [26], incorporate the action as a state variable by constraining the action process to be absolutely continuous with bounded growth. However, these methods are limited to deterministic dynamics and require upfront discretization of the continuous-time problem. These constraints underscore a key open challenge: how to design a principled and scalable Q-learning framework for stochastic, continuous-time environments that maintains the action characteristics of the Q-function. We address this by introducing a novel Q-function formulation and a corresponding algorithm, Continuous Q-Score Matching (CQSM), which operates directly in continuous time and supports stochastic dynamics.

Major contributions. This work makes the following key contributions:
(1) **Continuous-Time Q-Function Characterization.** We derive a Bellman equation (also known as the Feynman-Kac formula) for continuous-time Q-functions. This bridges discrete Q-learning with continuous control theory. We rigorously characterize Q-functions for a given score function using a martingale condition defined over an enlarged filtration that incorporates both state and action noise. Based on this foundation, we propose new algorithms for direct Q-function learning, analogous to policy evaluation and policy gradient methods in [23, 24].
(2) **Score Improvement Theorem and CQSM Algorithm.** We establish a score improvement theorem by the dynamic programming principle that enables principled policy updates in continuous time. By coupling the Q-function with the score function of a diffusion policy, we develop a model-free, actor-critic-style algorithm: Continuous Q-Score Matching (CQSM). This algorithm facilitates efficient policy improvement using only the denoising score function.
(3) **Analytical Validation via LQ Control.** We resolve the LQ problem under measurable scores, demonstrating the theoretical soundness of our framework. LQ problems are fundamental in control theory, as they serve as tractable approximations to more complex nonlinear systems, with practical relevance in areas such as algorithmic trading [2] and resource management [12]. Using analytical solutions, we compare CQSM to policy gradient and little q-learning methods, which are the action-independent value-based RL models, highlighting both its theoretical guarantees and numerical advantages.

In Section 2, we review related work relevant to our method. Section 3 introduces the continuous-time reinforcement learning formulation using stochastic differential equations and presents key preliminary results. In Section 4, we develop the Q-learning theory in continuous time, establishing martingale characterizations. We further extend the analysis to the infinite-horizon setting and prove the score improvement theorem. Section 5 presents numerical evaluations on LQ control tasks, comparing CQSM with policy gradient and little q-learning methods. Finally, Section 6 concludes with a summary and discussion of future directions.

2 Related Works

In this section, we review existing work across four areas related to our method: stochastic optimal control, continuous reinforcement learning, diffusion Q-learning, and behavior cloning.

Stochastic Optimal Control. Our approach builds on classical stochastic optimal control theory [57], particularly through the development of the Hamilton–Jacobi–Bellman (HJB) equation for

continuous-time Q-functions. While traditional stochastic control frameworks often assume full knowledge of the system dynamics [3], we instead assume a model for the dynamics of state-action pairs. This shift motivates new theoretical developments that underpin our Q-learning framework.

Continuous Reinforcement Learning. The formulation of CT-RL in stochastic settings—where state evolution follows a stochastic differential equation (SDE)—dates back to [33, 7], though early work lacked data-driven learning mechanisms. Recent advances have introduced more practical formulations. For instance, [52] proposed an exploratory control framework for continuous RL, while [53], [23], and [24] extended this line of work to mean-variance objectives, policy evaluation, and policy gradients, respectively. [25] further introduced the notion of a little q-value, leading to a continuous analogue of Q-learning. Additional developments include mean-field RL with continuous dynamics [28], jump-diffusion extensions [14], and infinite-horizon variants of TRPO and PPO [59]. Building on these foundations, we advance the study of continuous-time Q-learning under diffusion policies.

Diffusion Q-Learning. Diffusion Q-learning [55] integrates diffusion models with Q-learning by using Q-values as training objectives and backpropagates through the diffusion model. More recently, [5] proposed a model-free online RL method based on diffusion policies. [35] further established a connection between diffusion-based policies and the Q-function by relating the policy score to the action gradient of the Q-function. Building on this line of work, [54] employed entropy estimation to balance exploration and exploitation in diffusion policies, improving the performance of the policy. In parallel, [30] generalized diffusion model training by reweighting the conventional denoising score matching loss, leading to two efficient algorithms for training diffusion policies in online RL without requiring samples from optimal policies. However, their approach relies on time discretization and requires injecting noise into the final action of the diffusion chain. In contrast, our work preserves the full action-evaluation capability of the Q-function in continuous time, without relying on any discretization in either time or action space. Our method is grounded in a martingale-based formulation of the HJB equation, which provides a principled theoretical foundation for continuous-time Q-learning.

Behavior Cloning. Behavior cloning focuses on imitating expert trajectories without access to reward signals. Diffusion models are particularly well-suited for this task due to their generative flexibility and natural alignment with score-matching objectives. Recent works [22, 37] have applied diffusion models to behavior cloning by framing policy learning as a distribution-matching problem over expert data. These approaches inspired our incorporation of score-matching terms into the objective. However, our framework goes beyond imitation, enabling policy improvement through Q-function learning in continuous time.

3 Formulation and Preliminaries

In this section, we introduce the continuous-time RL formulation using stochastic differential equations and present key preliminary results.

Notation. We introduce the non-standard notation used throughout the main text and appendix. For a vector x, denote by $\|x\|_2$ the Euclidean norm of x. For a function f on an Euclidean space, ∇f (resp. $\nabla^2 f$) denotes the gradient (resp. the Hessian) of f. The Kullback–Leibler (KL) divergence of two positive density functions f, g is defined as $D_{KL}(f||g) := \int_A \log \frac{f(a)}{g(a)} f(a)\,\mathrm{d}a$. Define an operator $\mathcal{L} : C^{2,2}(\mathbb{R}^n \times \mathbb{R}^d) \cap C(\mathbb{R}^n \times \mathbb{R}^d) \to C(\mathbb{R})$ [57] associated with the diffusion process as:

$$\mathcal{L}\varphi(x,a) := \nabla_x \varphi(x,a)^\top b_X + \nabla_a \varphi(x,a)^\top \Psi + \frac{1}{2}\mathrm{tr}\left(\sigma_X \sigma_X^\top \nabla_x^2 \varphi(x,a)\right) + \frac{1}{2}\mathrm{tr}\left(\sigma_a \sigma_a^\top \nabla_a^2 \varphi(x,a)\right).$$

In both the main text and the proofs, we refer frequently to the *score* of the action distribution, denoted by Ψ. This vector field defines the temporal evolution of actions and serves as a proxy for the true score $\nabla_a \log \pi(a|x)$ where π refers to the action distribution.

Continuous RL. Let d, n be positive integers, $T > 0$. We denote the state as $X_t \in \mathbb{R}^n$ and the action as $a_t \in \mathbb{R}^d$ with $t \in [0, T]$. We consider the following stochastic, continuous-time setting for

state and action dynamics:

$$\mathrm{d}X_t = b_X(t, X_t, a_t)\,\mathrm{d}t + \sigma_X(t, X_t, a_t)\,\mathrm{d}B_t^X, \mathrm{d}a_t = \Psi(t, X_t, a_t)\,\mathrm{d}t + \sigma_a(t, X_t, a_t)\,\mathrm{d}B_t^a, \quad (1)$$

where $\Psi : [0,T] \times \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}^d$ corresponds to the score of our policy, which serves as the primary optimization variable in this setting, $b_X, b_a : [0,T] \times \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}^n$ corresponds to the continuous state and action dynamics, and σ_X, σ_a are functions from $[0,T] \times \mathbb{R}^n \times \mathbb{R}^d$ to positive semidefinite matrices in $\mathbb{R}^{n\times n}$ and $\mathbb{R}^{d\times d}$ respectively. The processes are driven by two independent Brownian motions: $B^X = \{B_s^X, s \geq 0\}$ and $B^a = \{B_s^a, s \geq 0\}$. All processes are defined on a filtered probability space $(\Omega, \mathcal{F}, \mathbb{P}; \{\mathcal{F}_s\}_{s\geq 0})$ where $\{\mathcal{F}_s\}_{s\geq 0}$ is the natural filtration generated by a standard n-dimensional Brownian motion B^X and a standard d-dimensional Brownian motion B^a. The (continuous-time) Q-function under any given Ψ is defined as

$$Q(t,x,a;\Psi) = \mathbb{E}^{\mathbb{P}}\left[\int_t^T \left[r(s, X_s, a_s) - \frac{1}{2}\lambda\|\Psi(s, X_s, a_s)\|_2^2\right]\mathrm{d}s + h(X_T, a_T)\middle| X_t = x, a_t = a\right], \quad (2)$$

where $\mathbb{E}^{\mathbb{P}}$ is the expectation with respect to both Brownian motions B_t^X and B_t^a, $r : [0,T] \times \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}$ and $h : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}$ are running and lump-sum reward function, respectively, and $\lambda > 0$ is the regularization coefficient governing the cost of large score magnitudes. The goal is to find an optimal score function $\Psi^* \in \Pi$ where Π denotes the set of admissible diffusion scores, such that the optimal Q-function

$$Q(t,x,a) = \sup_{\Psi\in\Pi} Q(t,x,a;\Psi). \quad (3)$$

We now give a precise definition of the admissible score set Π.

Definition 1. *A score Ψ is called admissible if*
(i) $\Psi := \{\Psi(t, X_t, a_t) : t \geq 0\}$ *is adapted;*
(ii) $\mathbb{E}^{\mathbb{P}}\left[\int_0^T \|\Psi(s, X_s, a_s)\|_2^2\,\mathrm{d}s\right] < \infty$.

Details regarding the well-posedness of the control problem (1)-(3) are provided in Appendix A.

The score matching term $\frac{1}{2}\lambda\|\Psi(X_s, a_s)\|_2^2$ in the objective can be interpreted from the following two perspectives:

Quadratic Execution Costs. Let a_t denote the investor's portfolio position. The score matching term captures the quadratic costs of execution trades of size $\Psi\,\mathrm{d}t$, where λ quantifies the level of transaction costs. This is consistent with execution cost models in portfolio optimization [1];

Policy Regularization via KL Divergence. Suppose the diffusion coefficient σ_a is deterministic and $\Psi(t, X_t, a_t) = b_a(t, X_t, a_t) + u(t, X_t, a_t)\sigma_a(t)$ where u represents a control. Under this setup, the score-matching cost admits an interpretation as a KL divergence between trajectory distributions. Specifically, let $\pi^{\text{base}}(a_T|x,a)$ denote the distribution over terminal actions when $u = 0$ and let $\pi^u(a_T|x_t,a_t)$ denote the distribution under control u. By the Section 3 in [6]), we have

$$D_{KL}(\pi^u(a_T|x,a)||\pi^{\text{base}}(a_T|x,a)) = \mathbb{E}^{\mathbb{P}}\left[\int_t^T \frac{1}{2}\|u(s, X_s, a_s)\|_2^2\,\mathrm{d}s\middle| X_t = x, a_t = a\right]. \quad (4)$$

Setting $\lambda = \|\sigma_a(t)\|^2$, the original score-matching term aligns exactly with this KL regularization. The corresponding objective can then be interpreted as:

$$\begin{aligned} Q(t,x,a) = &\sup_{u\in\Pi} \mathbb{E}^{\mathbb{P}}\left[\int_t^T r(X_s, a_s)\,\mathrm{d}s + h(X_T, a_T)\middle| X_t = x, a_t = a\right] \\ &- \lambda D_{KL}(\pi^u(a_T|x,a)||\pi^{\text{base}}(a_T|x,a)). \end{aligned} \quad (5)$$

Thus, the KL term encourages the optimal policy to remain close to the base dynamics, introducing a form of regularized policy improvement.

4 Continuous Q-Score Matching Algorithm

In this section, we develop a continuous-time Q-learning framework using a martingale characterization and the HJB equation. This offers an alternative approach to policy improvement.

4.1 Dynamic Programming and HJB Equation for Q-Function

By the dynamic programming principle, the Q-function satisfies the following HJB equation:

$$\sup_{\Psi\in\Pi}\left\{\mathcal{L}Q(t,x,a)+\frac{\partial Q}{\partial t}(t,x,a)+r(t,x,a)-\frac{1}{2}\lambda\|\Psi(t,x,a)\|_2^2\right\}=0. \tag{6}$$

Note that the terms $\nabla_x Q\cdot b_X(t,x,a), \nabla_a Q\cdot b_a(t,x,a), \frac{1}{2}\text{tr}\left(\sigma_X\sigma_X^\top\nabla_x^2 Q\right), \frac{1}{2}\text{tr}\left(\sigma_a\sigma_a^\top\nabla_a^2 Q\right)$ and $r(t,x,a)$ are all independent of Ψ. Hence, the supremum in (6) is attained at $\Psi^*(t,x,a)=\lambda^{-1}\nabla_a Q(t,x,a)$. Substituting this back into the HJB equation gives the following nonlinear partial differential equation characterizing the optimal Q-function:

$$\begin{cases}\dfrac{\partial Q}{\partial t}(t,x,a)+r(t,x,a)+\nabla_x Q(t,x,a)^\top\cdot b_X(t,x,a)+\dfrac{1}{2}\lambda^{-1}\|\nabla_a Q(t,x,a)\|_2^2\\ +\dfrac{1}{2}\text{tr}\left(\sigma_X(t,x,a)\sigma_X(t,x,a)^\top\nabla_x^2 Q(t,x,a)\right)+\dfrac{1}{2}\text{tr}\left(\sigma_a(t,x,a)\sigma_a(t,x,a)^\top\nabla_a^2 Q(t,x,a)\right)=0,\\ Q(T,x,a)=h(x,a).\end{cases} \tag{7}$$

We now focus on the optimal Q-function associated with the optimal score Ψ^*. To avoid unduly technicalities, we assume throughout this paper that the Q-function $Q\in C^{1,2,2}([0,T)\times\mathbb{R}^n\times\mathbb{R}^d)\cap C([0,T)\times\mathbb{R}^n\times\mathbb{R}^d)$ satisfies the polynomial growth condition in the joint state-action variable $z=(x,a)$. The following theorem establishes the key martingale characterization underpinning policy evaluation for diffusion-based policies.

Theorem 1. *If $Q(\cdot,\cdot,\cdot;\Psi)$ is the Q-function associated with the score Ψ if and only if it satisfies terminal condition $Q(T,x,a;\Psi)=h(x,a)$, and for all $(x,a)\in\mathbb{R}^n\times\mathbb{R}^d$, the following process*

$$M_s=Q(s,X_s,a_s;\Psi)+\int_t^s\left[r(u,X_u,a_u)-\frac{1}{2}\lambda\|\Psi(u,X_u,a_u)\|_2^2\right]\mathrm{d}u \tag{8}$$

is a $(\{\mathcal{F}_s\}_{s\geq 0},\mathbb{P})$-martingale on $[t,T]$. Conversely, if there is a continuous Q-function $\tilde{Q}$ such that for all $(x,a)\in\mathbb{R}^n\times\mathbb{R}^d$, $\tilde{M}_s$ is a martingale, where

$$\tilde{M}_s=\tilde{Q}(s,X_s,a_s;\Psi)+\int_t^s\left[r(u,X_u,a_u)-\frac{1}{2}\lambda\|\Psi(u,X_u,a_u)\|_2^2\right]\mathrm{d}u, \tag{9}$$

and $\tilde{Q}(T,x,a;\Psi)=h(x,a)$, then $\tilde{Q}=Q$ on $[0,T]\times\mathbb{R}^n\times\mathbb{R}^d$. Furthermore, the martingale property of $M\in L^2_{\mathcal{F}}([0,T])$is equivalent to the following orthogonality condition:

$$\mathbb{E}^{\mathbb{P}}\int_0^T\xi_t\left[\mathrm{d}Q(t,X_t,a_t;\Psi)+r(t,X_t,a_t)\,\mathrm{d}t-\frac{1}{2}\lambda\|\Psi(t,X_t,a_t)\|_2^2\,\mathrm{d}t\right]=0, \tag{10}$$

for any test process $\xi\in L^2_{\mathcal{F}}([0,T];Q(\cdot,X_\cdot,a_\cdot;\Psi))$.

Proof can be found in Appendix B.1. In summary, the martingality of the process defined in Equation (8) under a given score Ψ is both necessary and sufficient for Q to be the corresponding action value function.

4.2 Score Evaluation of the Q-Function

We now discuss how the HJB equation can be used to design a Q-learning algorithm for estimating $Q(x,a;\Psi)$ using sample trajectories. A number of algorithms can be developed based on two types of objectives: to minimize the martingale loss function or to satisfy the martingale orthogonality conditions. Following [23], we leverage the martingale orthogonality condition, which states that for any $T>0$ and a suitable test process ξ,

$$\mathbb{E}^{\mathbb{P}}\int_0^T\xi_t\left\{\mathrm{d}Q(t,X_t,a_t;\Psi)+\left[r(t,X_t,a_t)-\frac{1}{2}\lambda\|\Psi(t,X_t,a_t)\|_2^2\right]\mathrm{d}t\right\}=0. \tag{11}$$

To approximate the Q-function, we consider a parameterized family $Q^\theta(\cdot,\cdot,\cdot;\Psi)$ where $\theta\in\Theta\subset\mathbb{R}^{L_\theta}$ (in principle, we need at least L_θ equations as our martingale orthogonality conditions in order to fully

determine θ.) and choose the special test function $\xi_t = \frac{\partial Q^\theta}{\partial \theta}(t, X_t, a_t; \Psi)$. Stochastic approximation [38] leads to the online update:

$$\theta \leftarrow \theta + \alpha_\theta \frac{\partial Q^\theta}{\partial \theta}(t, X_t, a_t; \Psi) \left(\mathrm{d}Q(t, X_t, a_t; \Psi) + \left[r(t, X_t, a_t) - \frac{1}{2}\lambda \|\Psi(t, X_t, a_t)\|_2^2 \right] \mathrm{d}t \right) \quad (12)$$

where α_θ is a learning rate. This recovers the mean-squared TD error (MSTDE) method for policy evaluation in the discrete RL [43]. We must, however, stress that testing against this specific function is theoretically not sufficient to guarantee the martingale condition. Additional discussions are provided in Appendix D.

4.3 Policy Optimization via Matching the Score to the Q-function

Now we extend the analysis to the infinite-horizon setting. The dynamics of the state and action processes are given by:

$$\mathrm{d}X_t = b_X(X_t, a_t)\,\mathrm{d}t + \sigma_X(X_t, a_t)\,\mathrm{d}B_t^X, \mathrm{d}a_t = \Psi(X_t, a_t)\,\mathrm{d}t + \sigma_a(X_t, a_t)\,\mathrm{d}B_t^a, \quad (13)$$

and the following discounted Q-function under any given Ψ:

$$Q(x, a; \Psi) = \mathbb{E}^{\mathbb{P}} \left[\int_t^{+\infty} e^{-\beta(s-t)} \left[r(X_s, a_s) - \frac{1}{2}\lambda \|\Psi(X_s, a_s)\|_2^2 \right] \mathrm{d}s \middle| X_t = x, a_t = a \right], \quad (14)$$

where $\beta > 0$ is a discount factor that measures the time-depreciation of the objective value (or the impatience level of the agent) and the optimal Q-function $Q(x, a) = \sup_{\Psi \in \Pi} Q(x, a; \Psi)$.

Note that in this case, the Q-function does not depend on time explicitly. As a result, there is no terminal condition, but instead we have a growth condition $\mathbb{E}^{\mathbb{P}}[e^{-\beta t} Q(X_t, a_t; \Psi)] \to 0$ as $t \to \infty$. Again, using the dynamic programming principle, we have

$$\sup_{\Psi \in \Pi} \left\{ \mathcal{L}Q(x, a) - \beta Q(x, a) + r(x, a) - \frac{\lambda}{2} \|\Psi(x, a)\|_2^2 \right\} = 0. \quad (15)$$

We now introduce an alternative method for updating policies based on Q-function estimates that avoids the use of policy gradients. The following result serves as a score improvement theorem, analogous to the classic policy improvement theorem in reinforcement learning.

Theorem 2. *Let Ψ be any given score function and let the associated Q-function $Q(\cdot, \cdot; \Psi) \in C^{2,2}(\mathbb{R}^n \times \mathbb{R}^d) \cap C^0(\mathbb{R}^n \times \mathbb{R}^d)$. Suppose further that the score function Ψ^1 defined by $\Psi^1 = \lambda^{-1} \nabla_a Q(x, a; \Psi)$ for some $\lambda > 0$ is admissible. Then $Q(x, a; \Psi^1) \geq Q(x, a; \Psi), (x, a) \in \mathbb{R}^n \times \mathbb{R}^d$.*

Proof can be found in Appendix B.2.

[35] similarly constructed a score function Ψ^1, but their approach only determines the direction of the policy update, without specifying the magnitude of the update vector. This result implies that iteratively updating the score function by aligning it with the action gradient of the Q-function leads to monotonic improvement in the Q-values. In other words, setting $\Psi \leftarrow \lambda^{-1} \nabla_a Q(x, a)$ guarantees an improvement in the resulting Q-function globally. Figure 1 shows a visual description of Theorem 2 and the implied policy update direction via CQSM. Building on this theoretical foundation, we now describe how to implement a sample-based update using a parameterized score function Ψ^v with the parameter $v \in \mathbb{R}^{L_v}$. To match the direction of $\nabla_a Q(x, a)$, we define the update as: $v \in \arg\min \frac{1}{2} \|\Psi^v(x, a) - \lambda^{-1} \nabla_a Q(x, a)\|^2$. This yields the CQSM as shown in Algorithm 1.

Policy sampling. Consider the dynamics of (13) for a fixed state and setting $\sigma_a(x, a) = \sqrt{2} I_d$. Given certain conditions, under appropriate regularity conditions, the stationary distribution of the action a_t as $t \to \infty$ for any fixed $x \in \mathbb{R}^n$, denoted $\pi(a|x)$: $\pi(a|x) \sim e^{\frac{1}{\lambda} Q(x,a)}$. That is, the stationary action distribution corresponds to a Boltzmann distribution over actions: $\pi(a|x) = \frac{1}{Z} e^{\frac{1}{\lambda} Q(x,a)}$, where $Z = \int_{\mathbb{R}^d} e^{\frac{1}{\lambda} Q(s,a)}\,\mathrm{d}a$. This gives rise to a soft optimal policy, where the action distribution is shaped by the Q-values, rather than relying on the soft Hamiltonian or auxiliary q-functions used in the soft actor-critic literature (e.g., [24, 25]). However, for general $\sigma_a(x, a)$, the stationary distribution $\pi(a|x)$ may not have a closed-form expression [48]. More details about action sampling can be found in Appendix E.

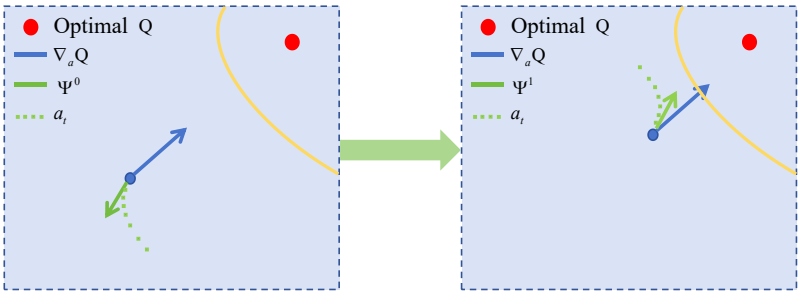


Figure 1: The left image shows a randomly initialized score function Ψ^0, and the right shows the updated score Ψ^1 after one step. If a discrepancy exists between the score Ψ (green vector) and the action gradient $\nabla_a Q(x, a)$ (blue vector), then aligning Ψ with $\nabla_a Q$ yields a strict improvement in Q-value at (x, a).

Here we highlight several hyperparameters: the trajectory truncation parameter (time horizon) T (needs to be sufficiently large); the total sample size N or the sampling interval Δt, with $N \cdot \Delta t = T$. We define the observation times as $t_k := k \cdot \Delta t, k = 0, \ldots, N-1$, at which data is collected from the simulated environment, denoted as Environment$_{\Delta t}$.

Algorithm 1 Continuous Q-Score Matching (CQSM)

Inputs: initial state x_0, time step Δt, initial learning rates α_θ, α_v and learning rate schedule function $l(\cdot)$ (a function of time), the action value function $Q^\theta(\cdot, \cdot)$, the score function $\Psi^v(\cdot, \cdot)$, the test function $\xi(x_{\cdot \wedge t}, a_{\cdot \wedge t})$, the initial parameter θ_0, v_0, and regularization parameter λ.
Required: Environment simulator $(x', r) = \text{Environment}_{\Delta t}(x, a)$.
for $k = 0, \ldots, N-1$ **do**
  Sample action a via iterative denoising: $a^T \sim \mathcal{N}(0, I) \to a^0$, using score Ψ^v: $a \sim \pi(x)$
  Simulate environment step: $(x', r) = \text{Environment}_{\Delta t}(x, a)$. Update state: $x_{t_{k+1}} \leftarrow x'$.
  Sample new action $a' \sim \pi(x')$ via denoising with Ψ^v. Store $a_{t_{k+1}} \leftarrow a'$
  Compute test function: $\xi_{t_k} = \xi(x_{t_0:k}, a_{t_0:k})$
  Compute temporal difference:

$$\delta = Q^\theta(x', a') - Q^\theta(x, a) + r(x, a)\Delta t - \frac{\lambda}{2}\|\Psi^v(x, a)\|^2 \Delta t - \beta Q^\theta(x, a)\Delta t$$

  Compute parameter updates:

$$\Delta\theta = \xi_{t_k} \cdot \delta, \quad \Delta v = \left(\frac{1}{\lambda}\nabla_a Q^\theta(x, a) - \Psi^v(x, a)\right)\frac{\partial \Psi^v}{\partial v}(x, a)$$

  Update parameters:

$$\theta \leftarrow \theta + l(k\Delta t)\alpha_\theta \Delta\theta, \quad v \leftarrow v + l(k\Delta t)\alpha_v \Delta v$$

  Set $x \leftarrow x'$
end for

5 Experiments

In this section, we present numerical evaluations on LQ control tasks. Additional experimental results are provided in Appendix F. We compare the performance of our proposed CQSM algorithm against continuous time policy gradient [24] and continuous time little q-learning methods [25]. Below, we briefly review these baseline methods.

- CT-RL policy gradient: Given an admissible policy, this method first performs policy evaluation to estimate the corresponding value function. It then computes the policy gradient

as: $g(t,x;\phi) = \frac{\partial}{\partial \phi} J(t,x;\pi^\phi)$ (J is a value function). [24] transforms policy gradient into policy evaluation to develop a policy gradient algorithm.

- CT-RL q-learning: The little q-function is defined as the first-order derivative of the Q-function with respect to Δt. Policy improvement is achieved via $\pi^\phi(a|t,x) = \frac{\exp\{\frac{1}{\lambda} q^\phi(t,x,a)\}}{\int \exp\{\frac{1}{\lambda} q^\phi(t,x,a)\}\, da}$, leading to a continuous-time q-learning theory.

Linear-Quadratic Stochastic Control. We now focus on the family of stochastic control problems with linear state dynamics

$$b_X(x,a) = Ax + Ba \quad \text{and} \quad \sigma_X(x,a) = Cx + Da, \sigma_a(x,a) = \sqrt{2}, x, a \in \mathbb{R}, \tag{16}$$

where $A, B, C, D \in \mathbb{R}$ and the quadratic reward

$$r(x,a) = -\left(\frac{M}{2}x^2 + Rxa + \frac{N}{2}a^2 + Px + P'a\right), \tag{17}$$

where $M \geq 0, N > 0, R, P, P' \in \mathbb{R}$. If $D \neq 0$, then one smooth solution to the HJB equation

$$\beta Q(x,a) - Q_x b(x,a) - \frac{1}{2\lambda} Q_a^2 - \frac{1}{2}\sigma_X^2 Q_{xx} - \frac{1}{2}\sigma_a^2 Q_{aa} - r(x,a) = 0, \tag{18}$$

is given by $Q(x,a) = \frac{1}{2}k_0 x^2 + k_1 x + \frac{1}{2}k_2 a^2 + k_3 a + k_4 xa + k_5$ where

$$\begin{cases} k_0 = \frac{1}{\lambda(\beta - 2A - C^2)} k_4^2 - \frac{M}{\beta - 2A - C^2} \\ k_1 = \frac{1}{\lambda(\beta - A)} k_3 k_4 - \frac{P}{\beta - A} \\ k_2 = \frac{\beta}{2}\lambda - \lambda\sqrt{\frac{\beta^2}{4} + \frac{1}{\lambda}(N - 2\varpi)} \\ k_3 = -\frac{BP + P'(\beta - A)}{(\beta - A)\left(\beta - \frac{B}{\lambda(\beta - A)} k_4 - \frac{1}{\lambda} k_2\right)} \\ k_4 = \frac{-\lambda(\beta - 2A - C^2)B \pm \sqrt{\lambda^2(\beta - 2A - C^2)^2 B^2 + D^2(D^2\lambda M + 2\lambda(\beta - 2A - C^2)\varpi)}}{D^2} \\ k_5 = \frac{2\lambda k_2 + k_3^2}{2\lambda\beta} \end{cases}. \tag{19}$$

For the particular solution, we can verify that $k_2 < 0$. To ensure Q is concave, a property essential for verifying that this function indeed corresponds to the action-value function[1], we impose the additional conditions $k_0 < 0, k_0 k_2 - k_4^2 > 0$. Next, we state one of the main results of this paper.

Theorem 3. *Suppose the dynamics and the reward function are given by (16) and (17), respectively. Then, the Q-function is given by* $Q(x,a) = \frac{1}{2}k_0 x^2 + k_1 x + \frac{1}{2}k_2 a^2 + k_3 a + k_4 xa + k_5$ *where* $k_0, k_1, k_2, k_3, k_4, k_5$ *are as in (19). Furthermore, the optimal score function takes the form:*

$$\Psi^*(x,a) = \lambda^{-1}(k_2 a + k_3 + k_4 x). \tag{20}$$

Additional details of Theorem 3 are provided in the Appendix C.

In our simulations, to ensure the stationarity of the controlled state process, we use the following model parameters: $A = -1, B = C = 0, D = 1, M = N = P' = 2, R = P = 1, \beta = 1, \lambda = 0.1$. We parameterize the Q-function as $Q^\theta = \frac{1}{2}\theta_0 x^2 + \theta_1 x + \theta_2 a^2 + \theta_3 a + \theta_4 xa + \theta_5$ and the corresponding score function as $\Psi^v(x,a) = -e^{v_1} a + v_2 x + v_3$. Using these parameterizations and the model setup, the optimal parameter values are computed as:

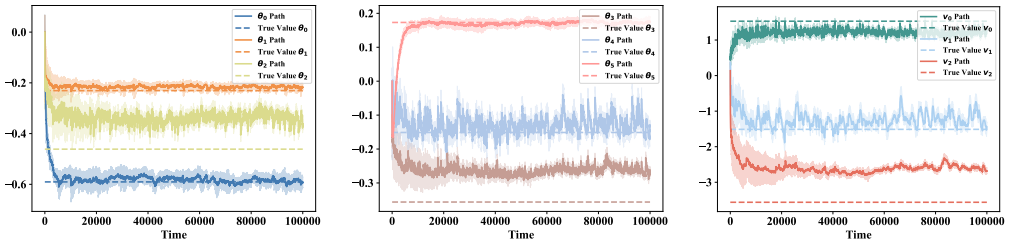
$$\theta^* = [-0.59047134, -0.23069812, -0.46141679, -0.35624157, -0.15119060, 0.17312350],$$
$$v^* = [1.52913155, -1.5119060, -3.5624157].$$

Implementation Details. The learning rate is initialized as $\alpha_\theta = \alpha_v = 0.01$ and decay according to $l(t) = \frac{1}{\max\{1, \sqrt{\log t}\}}$. To evaluate performance and stability, each experiment is repeated five times with different random seeds for sample generation. The parameter vector θ is initialized as zero and v is initialized in the range $[0,1]$. The corresponding optimal values θ^* and v^* are then used as baselines for comparison. The time cost for each experiment is 352 seconds on a computer with Intel Core i5-10500 CPU and 32G Memory.

[1] the HJB equation has an additional quadratic solution, which, however, is convex.

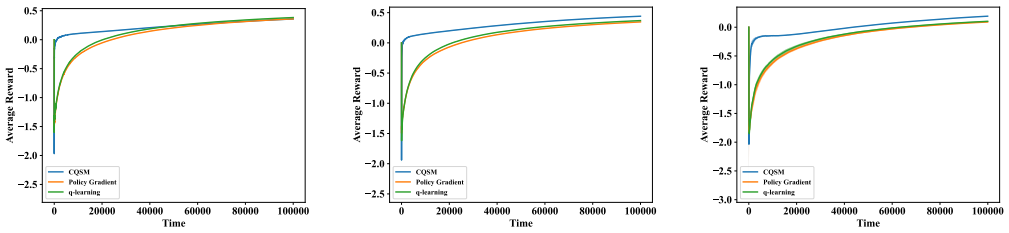
Performance Results. Each experiment is repeated ten times with different random seeds. Figure 2 illustrates the convergence behavior of the proposed CQSM algorithm for one realized trajectory with time step $\Delta t = 0.1$. Both the Q-function and score function parameters gradually approach their theoretical optima except θ_2, θ_3, v_2. Note that these parameters are closely tied to $\nabla_a Q$. Our method requires the estimation of both the Q-function Q and its gradient $\nabla_a Q$. This dual estimation introduces additional variance and bias, potentially leading to inaccurate policy updates.

Furthermore, we compare the performance of our CQSM algorithm against two benchmark methods in terms of the running average reward obtained during the learning process. The other two algorithms are the PG-based algorithm proposed in ([24], Algorithm 3) and the little q-learning algorithm presented in ([25], Algorithm 4). Figure 3 presents the running average rewards and standard deviations for all three methods under three step sizes, $\Delta t = 0.01, 0.1, 1$. Our proposed CQSM consistently outperforms the baselines in the early stages of training, achieving higher rewards more quickly than both PG and little q-learning. After a sufficient amount of time, all methods eventually stabilize to similar average reward levels.



(a) The path of learned $\theta_0, \theta_1, \theta_2$ (b) The path of learned $\theta_3, \theta_4, \theta_5$ (c) The path of learned v_0, v_1, v_2

Figure 2: Paths of learned parameters of the CQSM reinforcement learning algorithm described in Algorithm 1. A single state trajectory of length $T = 10^5$ is generated. The dashed lines indicate the optimal parameter values. The shaded regions represent the standard deviation of the learned parameters across these runs, with the width of each shaded area equal to twice the corresponding standard deviation.



(a) $\Delta t = 0.01$ (b) $\Delta t = 0.1$ (c) $\Delta t = 1$

Figure 3: Running average rewards of three RL algorithms. A single state trajectory of length $T = 10^5$ is generated and discretized using three different step sizes: $\Delta t = 0.01$ in panel (a), $\Delta t = 0.1$ in panel (b), $\Delta t = 1$ in panel (c). For each setting, we apply three online algorithms: Policy Gradient described in Algorithm 3 in [24], q-Learning described in Algorithm 4 in [25] and CQSM described in Algorithm 1. We plot the mean running average reward over time and the shaded areas represent the standard deviation across the runs.

6 Conclusion

In this paper, we introduce a Q-function framework for continuous-time stochastic optimal control problems with diffusion policies. By using the dynamic programming principle, we derive the associated HJB equation for the Q-function. Building on this and utilizing the martingale orthogonality condition, we develop the CQSM algorithm. We further demonstrate the effectiveness of CQSM

in an LQ setting, showing promising results compared to existing continuous-time reinforcement learning algorithms.

Several interesting directions remain for future work. For the finite-horizon case, extending the approach to handle an important portfolio selection with a mean-variance objective poses a challenging problem due to inherent time inconsistency. Another promising direction is the optimization of the diffusion term σ_a, which could lead to improved exploration and performance. Furthermore, a theoretical convergence rate analysis of CQSM could offer deeper insights and guide further enhancements to the algorithm.

Acknowledgements

We gratefully acknowledge financial support from the Key Project of National Natural Science Foundation of China 72432005, Guangdong Basic and Applied Basic Research Foundation 2023A1515030197.

References

[1] Alain Bensoussan, Guiyuan Ma, Chi Chung Siu, and Sheung Chi Phillip Yam. Dynamic mean-variance problem with frictions. *Finance and Stochastics*, 26(2):267–300, 2022.

[2] Alvaro Cartea, Sebastian Jaimungal, and Jason Ricci. Algorithmic trading, stochastic control, and mutually exciting processes. *SIAM review*, 60(3):673–703, 2018.

[3] Shuping Chen, Xunjing Li, and Xunyu Zhou. Stochastic linear quadratic regulators with indefinite control weight costs. *SIAM Journal on Control and Optimization*, 36(5):1685–1702, 1998.

[4] Min Dai, Yuchao Dong, and Yanwei Jia. Learning equilibrium mean-variance strategy. *Mathematical Finance*, 33(4):1166–1212, 2023.

[5] Shutong Ding, Ke Hu, Zhenhao Zhang, Kan Ren, Weinan Zhang, Jingyi Yu, Jingya Wang, and Ye Shi. Diffusion-based reinforcement learning via q-weighted variational policy optimization. *arXiv preprint arXiv:2405.16173*, 2024.

[6] Carles Domingo-Enrich, Michal Drozdzal, Brian Karrer, and Ricky TQ Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. *arXiv preprint arXiv:2409.08861*, 2024.

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

[8] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International conference on machine learning*, pages 1329–1338. PMLR, 2016.

[9] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.

[10] Xuefeng Gao, Zuoquan Xu, and Xunyu Zhou. State-dependent temperature control for langevin diffusions. *SIAM Journal on Control and Optimization*, 60(3):1250–1268, 2022.

[11] Xuefeng Gao, Jiale Zha, and Xunyu Zhou. Reward-directed score-based diffusion models via q-learning. *arXiv preprint arXiv:2409.04832*, 2024.

[12] P Jameson Graber. Linear quadratic mean field type control and mean field games with common noise, with application to production of an exhaustible resource. *Applied Mathematics & Optimization*, 74:459–486, 2016.

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

[14] Xin Guo, Anran Hu, and Yufei Zhang. Reinforcement learning for linear-convex models with jumps via stability analysis of feedback controls. *SIAM Journal on Control and Optimization*, 61(2):755–787, 2023.

[15] Xin Guo, Renyuan Xu, and Thaleia Zariphopoulou. Entropy regularization for mean field games with learning. *Mathematics of Operations research*, 47(4):3239–3260, 2022.

[16] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. Pmlr, 2018.

[17] Yinbin Han, Meisam Razaviyayn, and Renyuan Xu. Stochastic control for fine-tuning diffusion models: Optimality, regularity, and convergence. In *Forty-second International Conference on Machine Learning*, 2024.

[18] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020.

[19] Yilie Huang, Yanwei Jia, and Xunyu Zhou. Achieving mean–variance efficiency by continuous-time reinforcement learning. In *Proceedings of the Third ACM International Conference on AI in Finance*, pages 377–385, 2022.

[20] David Ireland and Giovanni Montana. Revalued: Regularised ensemble value-decomposition for factorisable markov decision processes. In *The Twelfth International Conference on Learning Representations*, 2024.

[21] Haque Ishfaq, Guangyuan Wang, Sami Nur Islam, and Doina Precup. Langevin soft actor-critic: Efficient exploration through uncertainty-driven critic learning. *arXiv preprint arXiv:2501.17827*, 2025.

[22] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

[23] Yanwei Jia and Xunyu Zhou. Policy evaluation and temporal-difference learning in continuous time and space: A martingale approach. *Journal of Machine Learning Research*, 23(154):1–55, 2022.

[24] Yanwei Jia and Xunyu Zhou. Policy gradient and actor-critic learning in continuous time and space: Theory and algorithms. *Journal of Machine Learning Research*, 23(275):1–50, 2022.

[25] Yanwei Jia and Xunyu Zhou. q-learning in continuous time. *Journal of Machine Learning Research*, 24(161):1–61, 2023.

[26] Jeongho Kim, Jaeuk Shin, and Insoon Yang. Hamilton-jacobi deep q-learning for deterministic continuous-time systems with lipschitz continuous controls. *Journal of Machine Learning Research*, 22(206):1–34, 2021.

[27] Pankaj Kumar. Deep reinforcement learning for high-frequency market making. In *Asian Conference on Machine Learning*, pages 531–546. PMLR, 2023.

[28] James-Michael Leahy, Bekzhan Kerimkulov, David Siska, and Lukasz Szpruch. Convergence of policy gradient for entropy regularized mdps with neural network approximation in the mean-field regime. In *International Conference on Machine Learning*, pages 12222–12252. PMLR, 2022.

[29] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[30] Haitong Ma, Tianyi Chen, Kai Wang, Na Li, and Bo Dai. Efficient online reinforcement learning for diffusion policy. In *Forty-second International Conference on Machine Learning*, 2025.

[31] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PmLR, 2016.

[32] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[33] Rémi Munos and Paul Bourgine. Reinforcement learning for continuous stochastic control problems. *Advances in neural information processing systems*, 10, 1997.

[34] Tim Pearce, Tabish Rashid, Anssi Kanervisto, Dave Bignell, Mingfei Sun, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Ida Momennejad, Katja Hofmann, et al. Imitating human behaviour with diffusion models. *arXiv preprint arXiv:2301.10677*, 2023.

[35] Michael Psenka, Alejandro Escontrela, Pieter Abbeel, and Yi Ma. Learning a diffusion model policy from rewards via q-score matching. In *International Conference on Machine Learning*, pages 41163–41182. PMLR, 2024.

[36] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[37] Moritz Reuss, Maximilian Li, Xiaogang Jia, and Rudolf Lioutikov. Goal-conditioned imitation learning using score-based diffusion policies. *arXiv preprint arXiv:2304.02532*, 2023.

[38] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

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

[40] Tim Seyde, Peter Werner, Wilko Schwarting, Igor Gilitschenski, Martin Riedmiller, Daniela Rus, and Markus Wulfmeier. Solving continuous control via q-learning. In *The Eleventh International Conference on Learning Representations*, 2023.

[41] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, pages 387–395. Pmlr, 2014.

[42] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

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

[44] Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[45] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999.

[46] Corentin Tallec, Léonard Blier, and Yann Ollivier. Making deep q-learning methods robust to time discretization. In *International Conference on Machine Learning*, pages 6096–6104. PMLR, 2019.

[47] Akshaj Tammewar, Nikita Chaudhari, Bunny Saini, Divya Venkatesh, Ganpathiraju Dharahas, Deepali Vora, Shruti Patil, Ketan Kotecha, and Sultan Alfarhood. Improving the performance of autonomous driving through deep reinforcement learning. *Sustainability*, 15(18):13799, 2023.

[48] Wenpin Tang, Yuming Paul Zhang, and Xunyu Zhou. Exploratory hjb equations and their convergence. *SIAM Journal on Control and Optimization*, 60(6):3191–3216, 2022.

[49] Arash Tavakoli, Mehdi Fatemi, and Petar Kormushev. Learning to represent action values as a hypergraph on the action vertices. In *International Conference on Learning Representations*, 2021.

[50] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[51] Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020.

[52] Haoran Wang, Thaleia Zariphopoulou, and Xunyu Zhou. Reinforcement learning in continuous time and space: A stochastic control approach. *Journal of Machine Learning Research*, 21(198):1–34, 2020.

[53] Haoran Wang and Xunyu Zhou. Continuous-time mean–variance portfolio selection: A reinforcement learning framework. *Mathematical Finance*, 30(4):1273–1308, 2020.

[54] Yinuo Wang, Likun Wang, Yuxuan Jiang, Wenjun Zou, Tong Liu, Xujie Song, Wenxuan Wang, Liming Xiao, Jiang Wu, Jingliang Duan, et al. Diffusion actor-critic with entropy regulator. *Advances in Neural Information Processing Systems*, 37:54183–54204, 2024.

[55] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023.

[56] Christopher John Cornish Hellaby Watkins. Learning from delayed rewards. *Ph. D. thesis, Cambridge University*, 1989.

[57] Jiongmin Yong and Xunyu Zhou. *Stochastic controls: Hamiltonian systems and HJB equations*, volume 43. Springer Science & Business Media, 1999.

[58] Hanyang Zhao, Haoxian Chen, Ji Zhang, David Yao, and Wenpin Tang. Score as action: Fine-tuning diffusion generative models by continuous-time reinforcement learning. In *ICLR 2025 Workshop on Deep Generative Model in Machine Learning: Theory, Principle and Efficacy*, 2025.

[59] Hanyang Zhao, Wenpin Tang, and David Yao. Policy optimization for continuous reinforcement learning. *Advances in Neural Information Processing Systems*, 36:13637–13663, 2023.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: It is contained in the abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: It is showed in section 4.2: This test function is theoretically not sufficient to guarantee the martingale condition. Be careful when choosing test function, or choose other methods.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Assumptions are in the section 3, 4, 5 and Proofs are provided in the Supplementary Materials.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: All the needed information is in the section 5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: For data, please see the section 5. The code will be released upon paper acceptance.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Experimental details are shown in the section 5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Experiments were repeated 5 times, and the mean and variance of parameters and reward were plotted in section 5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Compute resources are shown in the section 5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We follow the Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: This work is a foundational research.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Both baseline codes for comparison are publicly available under the open-source licenses.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The details of the code will be released upon paper acceptance.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Diffusion RL: Problem Formulation and Well-Posedness

Listed below are the standard assumptions to ensure the well-posedness of the stochastic control problem in (1)-(3).

Assumption 1. *The following conditions for the dynamics and reward functions hold true:*
(1) $b_X, \sigma_X, \sigma_a, r, h, \nabla_a r, \nabla_a h$ are all continuous functions in their respective arguments;
(2) b_X, σ_X, σ_a are globally Lipschitz continuous in (x, a), i.e., for $\varphi \in \{b, \sigma_X\}$, there exists a constant $C > 0$ such that

$$\|\varphi(t, x, a) - \varphi(t, x', a')\|_2 \leq C(\|x - x'\|_2 + \|a - a'\|_2), \forall t \in [0, T], x, x' \in \mathbb{R}^n, a, a' \in \mathbb{R}^d;$$

(3) b_X, σ_X are linear growth continuous in (x, a), i.e., for $\varphi \in \{b, \sigma_X\}$, there exists a constant $C > 0$ such that

$$\|\varphi(t, x, a)\|_2 \leq C(\|x\|_2 + \|a\|_2), \forall t \in [0, T], x \in \mathbb{R}^n, a \in \mathbb{R}^d;$$

(4) σ_a is bounded for any (x, a), i.e., there exist constants $C > 0$ such that $\|\sigma_a(t, x, a)\|_2 \leq C, \forall t \in [0, T], x \in \mathbb{R}^n, a \in \mathbb{R}^d$.
(5) $r, h, \nabla_a r, \nabla_a h$ have polynomial growth in (x, a), i.e., for $\varphi \in \{r, h, \nabla_a r, \nabla_a h\}$ there exist constants $C > 0$ such that

$$|\varphi(t, x, a)| \leq C(1 + \|x\|_2 + \|a\|_2), \forall t \in [0, T], x \in \mathbb{R}^n, a \in \mathbb{R}^d.$$

Lemma 1. *Suppose $\{a_t : t \geq 0\}$ follows*

$$\mathrm{d}a_t = b_t \,\mathrm{d}t + \sigma_t \,\mathrm{d}B_t^a, t \geq 0,$$

with $C_\sigma := \operatorname{ess\,sup}_{t,\omega} |\sigma_t| < \infty$. If $\int_0^t |b_s|^2 \,\mathrm{d}s < \infty$ for all $t \geq 0, a \in \mathbb{R}^d$, then there exists a constant $C > 0$, which is independent of T and a_0, such that

$$\mathbb{E}^{\mathbb{P}}\left[\sup_{0 \leq t \leq T} |a_t|^2\right] \leq C\left(1 + |a_0|^2\right), \forall T \geq 0. \tag{21}$$

Proof. By the elementary inequality

$$(a + b + c)^2 \leq (3 \max\{a, b, c\})^2 \leq 3^2 a^2 + 3^2 b^2 + 3^2 c^2, a, b, c \geq 0,$$

we have

$$\begin{aligned}
\mathbb{E}^{\mathbb{P}}\left[\sup_{0 \leq t \leq T} |a_t|^2\right] \leq & \mathbb{E}^{\mathbb{P}}\left[\left(|a_0| + \sup_{0 \leq t \leq T} \int_0^t |b_s| \,\mathrm{d}s + \sup_{0 \leq t \leq T} \left|\int_0^t \sigma_s \,\mathrm{d}B_s^a\right|\right)^2\right] \\
\leq & \mathbb{E}^{\mathbb{P}}\left[\left(|a_0| + \int_0^T |b_s| \,\mathrm{d}s + \sup_{0 \leq t \leq T} \left|\int_0^t \sigma_s \,\mathrm{d}B_s^a\right|\right)^2\right] \\
\leq & 3^2 |a_0|^2 + 3^2 \int_0^T |b_s|^2 \,\mathrm{d}s + 3^2 \mathbb{E}^{\mathbb{P}}\left[\sup_{0 \leq t \leq T} \left|\int_0^t \sigma_s \,\mathrm{d}B_s^a\right|^2\right] \\
\leq & 3^2 |a_0|^2 + 3^2 \int_0^T |b_s|^2 \,\mathrm{d}s + 3^2 C_2 \left(\mathbb{E}^{\mathbb{P}}\left[\int_0^t |\sigma_s|^2 \,\mathrm{d}s\right]\right) \\
\leq & 3^2 |a_0|^2 + 3^2 \int_0^T |b_s|^2 \,\mathrm{d}s + 3^2 C_2 C_\sigma^2 T \\
\leq & C\left(1 + |a_0|^2\right),
\end{aligned}$$

where the second to last inequality is due to the Burkholder–Davis–Gundy inequality. This proves (21). □

Lemma 2. *Let Assumption 1 hold, the solution of state SDE (1) satisfies the condition*

$$\mathbb{E}^{\mathbb{P}}\left[\sup_{0 \leq t \leq T} |X_t|^2\right] \leq C\left(1 + |x_0|^2\right), \tag{22}$$

for some constant $C > 0$.

Proof. Based on the proved growth condition on b_X, σ_X, Cauchy–Schwarz inequality, and Burkholder-Davis-Gundy inequalities, we obtain

$$\begin{aligned}
&\mathbb{E}^{\mathbb{P}}\left[\sup_{0\le t\le T}|X_t|^2\right]\\
\le& C_1\mathbb{E}^{\mathbb{P}}\left[|x_0|^2+\sup_{0\le t\le T}\left|\int_0^t b_X(s,X_s,a_s)\,\mathrm{d}s\right|^2+\sup_{0\le t\le T}\left|\int_0^t \sigma_X(s,X_s,a_s)\,\mathrm{d}B_s^X\right|^2\right]\\
\le& C_1\mathbb{E}^{\mathbb{P}}\left[|x_0|^2+C_2\int_0^T\left(\sup_{0\le\tau\le s}|X_\tau|^2+|a_s|^2\right)\mathrm{d}s\right]\\
\le& C_3(1+|x_0|^2)+C_4\int_0^T\sup_{0\le\tau\le s}\mathbb{E}^{\mathbb{P}}\left[|X_\tau|^2\right]\mathrm{d}s.
\end{aligned}$$

Applying Gronwall's inequality, we obtain the desired result. □

Theorem 4. *Let Assumption 1 hold, then there exists a constant $C_1>0$ such that the Q-function satisfies*

$$|Q(t,x,a)|\le C_1(1+\|x\|_2+\|a\|_2),$$

for all $t\in[0,T], x\in\mathbb{R}^n, a\in\mathbb{R}^d$. Finally, the Q-function is finite.

Proof. Let $\Psi=C$ where C is a constant. It then follows from Lemma 1 and Lemma 2 that

$$\begin{aligned}
&Q(t,x,a)\\
\ge&\mathbb{E}^{\mathbb{P}}\left[\int_t^T\left[r(s,X_s,a_s)-\frac{\lambda}{2}\|\Psi(s,X_s,a_s)\|_2^2\right]\mathrm{d}s+h(X_T,a_T)\middle|X_t=x,a_t=a\right]\\
\ge&\mathbb{E}\left[\int_t^T\left(-C(1+\|X_s\|_2+\|a_s\|_2)-\frac{\lambda}{2}C^2\right)\mathrm{d}s\middle|X_t=x,a_t=a\right]\\
\ge&-C'(1+\|x\|_2+\|a\|_2)
\end{aligned}$$

for some constant C' independent of x,a. On the other hand, for any $\Psi\in\Pi$, we have

$$\begin{aligned}
Q(t,x,a)=&\mathbb{E}^{\mathbb{P}}\left[\int_t^T\left[r(X_s,a_s^p)-\frac{\lambda}{2}\|\Psi(X_s,a_s^p)\|_2^2\right]\mathrm{d}s+h(X_T,a_T)\middle|X_t=x,a_t=a\right]\\
\le&\mathbb{E}^{\mathbb{P}}\left[\int_t^T\left[C(1+\|X_s\|_2+\|a_s\|_2)\right]\mathrm{d}s\middle|X_t=x,a_t=a\right]\\
\le&C''(1+\|x\|_2+\|a\|_2)
\end{aligned}$$

for some constant C'' independent of x,a. The final result is evident. The proof is complete. □

We have indeed established in the above that

$$\mathbb{E}^{\mathbb{P}}\left[\int_t^T|r(s,X_s,a_s)|\,\mathrm{d}s\right]<\infty. \tag{23}$$

B Proofs of Martingale Characterization and Score Improvement Theorem

B.1 Proof of Theorem 1

To show $M_s=Q(s,X_s,a_s;\Psi)+\int_t^s\left[r(u,X_u,a_u)-\frac{1}{2}\lambda\|\Psi(u,X_u,a_u)\|_2^2\right]\mathrm{d}u$ is a martingale, observe that

$$\begin{aligned}
M_s=&\mathbb{E}^{\mathbb{P}}\left[\int_s^T\left[r(u,X_u,a_u)-\frac{1}{2}\lambda\|\Psi(u,X_u,a_u)\|_2^2\right]\mathrm{d}s+h(X_T,a_T)\middle|X_s,a_s\right]\\
&+\int_t^s\left[r(u,X_u,a_u)-\frac{1}{2}\lambda\|\Psi(u,X_u,a_u)\|_2^2\right]\mathrm{d}u\\
=&\mathbb{E}^{\mathbb{P}}\left[M_T|\mathcal{F}_s\right],
\end{aligned} \tag{24}$$

where we have used the Markov property of the process $\{(X_s, a_s), t \le s \le T\}$. This establishes that M is a martingale.

Conversely, if $\tilde{M}$ is a martingale, then $\tilde{M}_s = \mathbb{E}^{\mathbb{P}}\left[\tilde{M}_T | \mathcal{F}_s\right]$, which is equivalent to

$$\begin{aligned}\tilde{Q}(s, X_s, a_s; \Psi) =& \mathbb{E}^{\mathbb{P}}\left[\int_s^T \left[r(u, X_u, a_u) - \frac{1}{2}\lambda\|\Psi(u, X_u, a_u)\|_2^2\right] \mathrm{d}u + \tilde{Q}(T, X_T, a_T)\middle| \mathcal{F}_s\right] \\ =& \mathbb{E}^{\mathbb{P}}\left[\int_s^T \left[r(u, X_u, a_u) - \frac{1}{2}\lambda\|\Psi(u, X_u, a_u)\|_2^2\right] \mathrm{d}u + h(X_T)\middle| \mathcal{F}_s\right] \\ =& Q(s, X_s, a_s; \Psi), s \in [t, T].\end{aligned} \tag{25}$$

Letting $s = t$, we conclude $\tilde{Q}(t, x, a; \Psi) = Q(t, x, a; \Psi)$.

The "only if" part is evident. To prove the "if" part, assume that $\mathrm{d}M_t = A_t\,\mathrm{d}t + C_t\,\mathrm{d}B_t$. In particular, in our case, $A_t = \mathcal{L}Q(t, x, a; \Psi) + r(t, X_t, a_t) - \frac{1}{2}\lambda\|\Psi(t, X_t, a_t)\|_2^2$ and $C_t = \left(\frac{\partial Q}{\partial Z}\right)^\top G(t, Z_t)$. $A, C \in L^2_{\mathcal{F}}([0, T])$ follows by assumption ($Q \in C^{1,2,2}([0, T) \times \mathbb{R}^n \times \mathbb{R}^d) \cap C([0, T) \times \mathbb{R}^n \times \mathbb{R}^d)$) and Theorem 4. For any $0 \le s < s' \le T$, take $\xi_t = sgn(A_t)$ if $t \in [s, s']$ and $\xi_t = 0$ otherwise. Then

$$0 = \mathbb{E}^{\mathbb{P}}\int_s^{s'} \xi_t\,\mathrm{d}M_t = \mathbb{E}^{\mathbb{P}}\int_s^{s'} (|A_t|\,\mathrm{d}t + \xi_t C_t\,\mathrm{d}B_t) = \mathbb{E}^{\mathbb{P}}\int_s^{s'} |A_t|\,\mathrm{d}t, \tag{26}$$

where the expectation of the second term vanishes because $|\xi C| \le |C| \in L^2_{\mathcal{F}}([0, T])$ and hence $\mathbb{E}^{\mathbb{P}}\int_0^{\cdot} \xi_t C_t\,\mathrm{d}B_t$ is a martingale. This yields $A_t = 0$ almost surely, and thus M is a martingale.

B.2 Proof of Theorem 2

Fix $(x, a) \in \mathbb{R}^n \times \mathbb{R}^d$, applying Ito's formula, we have

$$\begin{aligned}e^{-\beta s}Q(X_s, a_s; \Psi) = e^{-\beta t}Q(x, a; \Psi) + \int_t^s e^{-\beta(\tau - t)} \big\{&-\beta Q(X_\tau, a_\tau; \Psi) + \nabla_x Q^\top \cdot b_X(X_\tau, a_\tau) \\ &+\nabla_a Q^\top \cdot \Psi(X_\tau, a_\tau)) + \frac{1}{2}\mathrm{tr}\left(\sigma_X \sigma_X^\top \nabla_x^2 Q\right) + \frac{1}{2}\mathrm{tr}\left(\sigma_a \sigma_a^\top \nabla_a^2 Q\right)\Big\}\,\mathrm{d}\tau \\ &+ \int_t^s e^{-\beta(\tau - t)}\nabla_x Q^\top \cdot \sigma_X(X_\tau, a_\tau)\,\mathrm{d}B_\tau^X + e^{-\beta\tau}\nabla_a Q^\top \cdot \sigma_a(X_\tau, a_\tau)\,\mathrm{d}B_\tau^a.\end{aligned} \tag{27}$$

Define the stopping times $T_n := \inf\{s \ge t : \|X_s\|_2 \ge n, \|a_s\|_2 \ge n\}$, for $n \ge 1$. Then we have

$$\begin{aligned}\mathbb{E}^{\mathbb{P}}\left[e^{-\beta(s \wedge T_n)}Q(X_{s \wedge T_n}, a_{s \wedge T_n}; \Psi)\middle| X_t = x, a_t = a\right] &= e^{-\beta t}Q(x, a; \Psi) \\ + \mathbb{E}^{\mathbb{P}}\Bigg[\int_t^{s \wedge T_n} e^{-\beta(\tau - t)} \big\{&-\beta Q(X_\tau, a_\tau; \Psi) + \nabla_x Q^{\Psi, \top} \cdot b_X(X_\tau, a_\tau) + \nabla_a Q^\top \cdot \Psi(X_\tau, a_\tau) \\ &+\frac{1}{2}\mathrm{tr}\left(\sigma_X \sigma_X^\top \nabla_x^2 Q\right) + \frac{1}{2}\mathrm{tr}\left(\sigma_a \sigma_a^\top \nabla_a^2 Q\right)\Big\}\,\mathrm{d}\tau\Bigg| X_t = x, a_t = a\Bigg].\end{aligned} \tag{28}$$

On the other hand, by standard arguments and the assumption that $Q(\cdot, \cdot; \Psi)$ is smooth, we have

$$\begin{aligned}\beta Q(x, a; \Psi) - \big\{&\nabla_x Q^\top \cdot b_X(x, a) + \nabla_a Q^\top \cdot \Psi(x, a) \\ &+\frac{1}{2}\mathrm{tr}\left(\sigma_X \sigma_X^\top \nabla_x^2 Q\right) + \frac{1}{2}\mathrm{tr}\left(\sigma_a \sigma_a^\top \nabla_a^2 Q\right) + r(x, a) - \frac{1}{2}\lambda\|\Psi\|_2^2\Big\} = 0,\end{aligned} \tag{29}$$

for any $(x, a) \in \mathbb{R}^n \times \mathbb{R}^d$. It follows that

$$\begin{aligned}\beta Q(x, a; \Psi) - \sup_{\tilde{\Psi}} \Big\{&\nabla_x Q^{\Psi, \top} \cdot b_X(x, a) + \nabla_a Q^\top \cdot \tilde{\Psi}(x, a) \\ &+\frac{1}{2}\mathrm{tr}\left(\sigma_X \sigma_X^\top \nabla_x^2 Q\right) + \frac{1}{2}\mathrm{tr}\left(\sigma_a \sigma_a^\top \nabla_a^2 Q\right) + r(x, a) - \frac{1}{2}\lambda\|\tilde{\Psi}\|_2^2\Big\} \le 0.\end{aligned} \tag{30}$$

Notice that the minimizer of the Hamiltonian in (30) is given by $\Psi^1 = \lambda^{-1}\nabla_a Q(x,a;\Psi)$ for some $\lambda > 0$. It then follows that Equation (28) implies

$$\begin{aligned}&\mathbb{E}^{\mathbb{P}}\left[e^{-\beta(s\wedge T_n)}Q(X_{s\wedge T_n}, a_{s\wedge T_n};\Psi)\middle| X_t = x, a_t = a\right]\\ \geq& e^{-\beta t}Q(x,a;\Psi) - \mathbb{E}^{\mathbb{P}}\left[\int_t^{s\wedge T_n} e^{-\beta(\tau-t)}\left(r(x_\tau,a_\tau) - \frac{\lambda}{2}\|\Psi^1\|_2^2\right)\mathrm{d}\tau\middle| X_t = x, a_t = a\right].\end{aligned} \tag{31}$$

Sending $n \to \infty$, we deduce that

$$\begin{aligned}&\mathbb{E}^{\mathbb{P}}\left[e^{-\beta s}Q(X_s, a_s;\Psi)\middle| X_t = x, a_t = a\right]\\ \geq& e^{-\beta t}Q(x,a;\Psi) - \mathbb{E}^{\mathbb{P}}\left[\int_t^{s} e^{-\beta(\tau-t)}\left(r(x_\tau,a_\tau) - \frac{\lambda}{2}\|\Psi^1\|_2^2\right)\mathrm{d}\tau\middle| X_t = x, a_t = a\right].\end{aligned} \tag{32}$$

Noting Lemma 1, Lemma 2 and Q is polynomial growth, we have

$$\begin{aligned}&\liminf_{s\to\infty}\mathbb{E}^{\mathbb{P}}\left[e^{-\beta s}Q(X_s, a_s;\Psi)\middle| X_t = x, a_t = a\right]\\ \leq& \limsup_{s\to\infty}\mathbb{E}^{\mathbb{P}}\left[e^{-\beta s}Q(X_s, a_s;\Psi)\middle| X_t = x, a_t = a\right] = 0\end{aligned} \tag{33}$$

and applying the dominated convergence theorem yield

$$Q(x,a;\Psi) \leq \mathbb{E}^{\mathbb{P}}\left[\int_t^{\infty} e^{-\beta\tau}\left(r(x_\tau,a_\tau) - \frac{\lambda}{2}\|\Psi^1\|_2^2\right)\mathrm{d}\tau\middle| X_t = x, a_t = a\right] = Q(x,a;\Psi^1). \tag{34}$$

C Derivation of Linear-Quadratic Stochastic Control

The following derivation corresponds to Section 5 of the main text.

Assumption 2. *The discounted rate satisfies* $\beta > 2A + C^2$.

This assumption ensures a sufficiently large discount rate, which guarantees that $\liminf_{T\to\infty} e^{-\beta T}\mathbb{E}^{\mathbb{P}}\left[Q(X_T, a_T;\Psi)\right] = 0$ for any score Ψ, thereby ensuring the corresponding expected reward remains finite.

By HJB equation

$$\beta Q(x,a) - Q_x b(x,a) - \frac{1}{2\lambda}Q_a^2 - \frac{1}{2}\sigma_X^2 Q_{xx} - \frac{1}{2}\sigma_a^2 Q_{aa} - r(x,a) = 0, \tag{35}$$

we have

$$\begin{aligned}x^2:&\quad \frac{\beta}{2}k_0 - Ak_0 - \frac{1}{2\lambda}k_4^2 - \frac{1}{2}C^2k_0 + M/2 = 0,\\ x:&\quad \beta k_1 - Ak_1 - \frac{k_3k_4}{\lambda} + P = 0,\\ a^2:&\quad \frac{\beta}{2}k_2 - k_4B - \frac{1}{2\lambda}k_2^2 - \frac{1}{2}k_0D^2 + N/2 = 0,\\ a:&\quad \beta k_3 - Bk_1 - \frac{k_2k_3}{\lambda} + P' = 0,\\ xa:&\quad \beta k_4 - k_0B - k_4A - \frac{1}{\lambda}k_2k_4 - k_0CD + R = 0,\\ \text{Cons}:&\quad \beta k_5 - k_2 - \frac{1}{2\lambda}k_3^2 = 0.\end{aligned}$$

By x^2 term:

$$k_0 = \frac{1}{\lambda(\beta - 2A - C^2)}k_4^2 - \frac{M}{\beta - 2A - C^2} \tag{36}$$

and substitute k_0 to a^2 term, we obtain

$$\frac{\beta}{2}k_2 - k_4B - \frac{1}{2\lambda}k_2^2 - \frac{1}{2}D^2\left(\frac{1}{\lambda(\beta - 2A - C^2)}k_4^2 - \frac{M}{\beta - 2A - C^2}\right) + N/2 = 0, \tag{37}$$

$$\frac{\beta}{2}k_2 - \frac{1}{2\lambda}k_2^2 + N/2 = k_4B + \frac{1}{2}D^2\left(\frac{1}{\lambda(\beta - 2A - C^2)}k_4^2 - \frac{M}{\beta - 2A - C^2}\right) := \varpi. \tag{38}$$

First, we consider $D \neq 0$. Hence, we have

$$k_2 = \frac{\beta}{2}\lambda - \lambda\sqrt{\frac{\beta^2}{4} + \frac{1}{\lambda}(N - 2\varpi)}, \tag{39}$$

$$k_4 = \frac{-\lambda(\beta - 2A - C^2)B \pm \sqrt{\lambda^2(\beta - 2A - C^2)^2B^2 + D^2(D^2\lambda M + 2\lambda(\beta - 2A - C^2)\varpi)}}{D^2}. \tag{40}$$

By xa term, we can determine the value of ϖ and ϖ satisfies the following bounds

$$\frac{\beta^2}{4} + \frac{1}{\lambda}(N - 2\varpi) \geq 0, \tag{41}$$

$$\lambda^2(\beta - 2A - C^2)^2B^2 + D^2(D^2\lambda M + 2\lambda(\beta - 2A - C^2)\varpi) \geq 0. \tag{42}$$

If $D = 0$, then

$$k_2 = \frac{\beta\lambda}{2} - \lambda\sqrt{\frac{\beta^2}{4} - \frac{2}{\lambda}\left(\frac{N}{2} - k_4 B\right)}, \tag{43}$$

and k_4 satisfies the following equation:

$$-\frac{B}{\lambda(\beta - 2A)}k_4^2 + \left(\frac{\beta}{2} - A + \sqrt{\frac{\beta^2}{4} - \frac{2}{\lambda}\left(\frac{N}{2} - k_4 B\right)}\right)k_4 + \frac{MB}{\beta - 2A} + R = 0. \tag{44}$$

Furthermore, by x term and a term, we have

$$k_1 = \frac{1}{\lambda(\beta - A)}k_3 k_4 - \frac{P}{\beta - A}, \tag{45}$$

$$k_3 = -\frac{BP + P'(\beta - A)}{(\beta - A)\left(\beta - \frac{B}{\lambda(\beta - A)}k_4 - \frac{1}{\lambda}k_2\right)}. \tag{46}$$

Finally, we have

$$k_5 = \frac{2\lambda k_2 + k_3^2}{2\lambda\beta}. \tag{47}$$

Therefore, the optimal score function takes the form:

$$\Psi^*(x, a) = \lambda^{-1}\nabla_a Q(x, a) = \lambda^{-1}(k_2 a + k_3 + k_4 x). \tag{48}$$

D Continuous Actor-Critic Q-Learning Algorithms

Score Evaluation of the Q-Function. We provide a complete description of how the HJB equation is used to construct the continuous-time Q-learning algorithm. For estimating $Q(x, a; \Psi)$, a number of algorithms can be developed based on two types of objectives: to minimize the martingale loss function or to satisfy the martingale orthogonality conditions. We summarize these methods in the Q-learning context below.

(1) Minimize the martingale loss function:

$$\frac{1}{2}\mathbb{E}^{\mathbb{P}}\left[\int_0^T \left[h(X_T, a_T) - Q^\theta(t, X_t, a_t) + \int_t^T \left(r(s, X_s, a_s) - \frac{\lambda}{2}\|\Psi^v(s, X_s, a_s)\|^2\right)\mathrm{d}s\right]^2 \mathrm{d}t\right]. \tag{49}$$

This method is intrinsically offline because the loss function involves the whole horizon $[0, T]$. We can apply stochastic gradient decent to update

$$\begin{aligned} \theta &\leftarrow \theta + \alpha_\theta \int_0^T \frac{\partial Q^\theta}{\partial \theta}(t, X_t, a_t)G_{t:T}\,\mathrm{d}t, \\ v &\leftarrow v + \alpha_v \int_0^T \int_t^T \lambda\Psi^v(s, X_s, a_s)\frac{\partial \Psi^v}{\partial v}(s, X_s, a_s)\,\mathrm{d}s G_{t:T}\,\mathrm{d}t, \end{aligned} \tag{50}$$

where $G_{t:T} = h(X_T, a_T) - Q^\theta(t, X_t, a_t) + \int_t^T \left(r(s, X_s, a_s) - \frac{\lambda}{2}\|\Psi^v(s, X_s, a_s)\|^2\right) \mathrm{d}s$. We present Algorithm 2 based on this updating rule. Note that this algorithm is analogous to the classical gradient Monte Carlo method or TD(1) for MDPs [44] because full sample trajectories are used to compute gradients.

(2) We leverage the martingale orthogonality condition, which states that for any $T > 0$ and a suitable test process ξ,

$$\mathbb{E}^{\mathbb{P}} \int_0^T \xi_t \left\{ \mathrm{d}Q^\theta(t, X_t, a_t; \Psi) + \left[r(t, X_t, a_t) - \frac{1}{2}\lambda \|\Psi^v(t, X_t, a_t)\|_2^2 \right] \mathrm{d}t \right\} = 0. \tag{51}$$

We use stochastic approximation to update θ either offline by

$$\theta \leftarrow \theta + \alpha_\theta \int_0^T \xi_t \left\{ \mathrm{d}Q^\theta(t, X_t, a_t; \Psi) + \left[r(t, X_t, a_t) - \frac{1}{2}\lambda \|\Psi^v(t, X_t, a_t)\|_2^2 \right] \mathrm{d}t \right\}, \tag{52}$$

or online by

$$\theta \leftarrow \theta + \alpha_\theta \xi_t \left\{ \mathrm{d}Q^\theta(t, X_t, a_t; \Psi) + \left[r(t, X_t, a_t) - \frac{1}{2}\lambda \|\Psi^v(t, X_t, a_t)\|_2^2 \right] \mathrm{d}t \right\}. \tag{53}$$

Typical choices of test functions are $\xi_t = \frac{\partial Q^\theta}{\partial \theta}$ or $\xi_t = \int_0^t \rho^{s-t} \frac{\partial Q^\theta}{\partial \theta} \mathrm{d}s, 0 < \rho \leq 1$ which lead to Q-learning algorithms based on stochastic approximation. However, relying solely on this specific choice does not, in general, guarantee satisfaction of the full martingale condition. Moreover, the convergence of the resulting stochastic approximation algorithm is not assured without additional assumptions. As discussed in [23], the selection of test functions must be carefully tailored to the structure of the Q-function, highlighting the need for more robust and theoretically grounded choices in continuous-time settings.

(3) Choose the same type of test functions ξ_t as above but now minimize the GMM objective functions:

$$\begin{aligned} &\mathbb{E}^{\mathbb{P}} \left[\int_0^T \xi_t \left[\mathrm{d}J^\theta(t, X_t^{\pi^\psi}) + r(t, X_t^{\pi^\psi}, a_t^{\pi^\psi})\mathrm{d}t - q^\psi(t, X_t^{\pi^\psi}, a_t^{\pi^\psi})\mathrm{d}t - \beta J^\theta(t, X_t^{\pi^\psi})\mathrm{d}t \right]^\top \right] \\ &A_\theta \mathbb{E}^{\mathbb{P}} \left[\int_0^T \xi_t \left[\mathrm{d}J^\theta(t, X_t^{\pi^\psi}) + r(t, X_t^{\pi^\psi}, a_t^{\pi^\psi})\mathrm{d}t - q^\psi(t, X_t^{\pi^\psi}, a_t^{\pi^\psi})\mathrm{d}t - \beta J^\theta(t, X_t^{\pi^\psi})\mathrm{d}t \right] \right], \end{aligned} \tag{54}$$

where $A_\theta \in \mathbb{S}^{L_\theta}$. Typical choices of these matrices are $A_\theta = I_{L_\theta}$ or $A_\theta = (\mathbb{E}^{\mathbb{P}}[\int_0^T \xi_t \xi_t^\top \mathrm{d}t])^{-1}$. Again, we refer the reader to [23] for discussions on these choices and the connection with the classical GTD algorithms and GMM method.

Score Gradient. We aim to compute the score gradient $g(x, a; v) := \frac{\partial}{\partial v} Q(x, a; \Psi^v) \in \mathbb{R}^{L_v}$ at the current state-action pair (x, a). Based on the HJB of the Q-function, we take the derivative in v on both sides to get

$$\mathcal{L}g(x, a; v) - \beta g(x, a; v) + (\nabla_a Q(x, a; \Psi^v) - \lambda \Psi^v(x, a)) \frac{\partial \Psi^v}{\partial v}(x, a) = 0. \tag{55}$$

Thus, a Feynman-Kac formula represents g as

$$g(x, a; v) = \mathbb{E}^{\mathbb{P}} \left[\int_t^\infty e^{-\beta s} (\nabla_a Q(X_s, a_s; \Psi^v) - \lambda \Psi^v(X_s, a_s)) \frac{\partial \Psi^v}{\partial v}(X_s, a_s)\, \mathrm{d}s \middle| X_t = x, a_t = a \right]. \tag{56}$$

To treat the online case, assume that v^* is the optimal point of $Q(x, a; \Psi^v)$ for any (x, a) and that the first-order condition holds (e.g., when v^* is an interior point). Then $g(x, a; v^*) = 0$. It follows that

$$0 = \mathbb{E}^{\mathbb{P}} \left[\int_t^\infty \eta_s (\nabla_a Q(X_s, a_s; \Psi^v) - \lambda \Psi^v(X_s, a_s)) \frac{\partial \Psi^v}{\partial v}(X_s, a_s)\, \mathrm{d}s \middle| X_t = x, a_t = a \right], \tag{57}$$

for any $\eta \in L^2_{\mathcal{F}}([0, T]; Q(\cdot, X_\cdot, a_\cdot; \Psi))$. If we take $\eta_s = e^{-\beta s}$, then the right hand side (57) coincides with $g(x, a; v^*)$. More importantly, besides the flexibility of choosing different sets of test functions,

(57) provides a way to derive a system of equations based on only past observations and, hence, enables online learning. For example, by taking $\eta_s = 0$ on $[T, \infty]$, (57) involves sample trajectories up to time T. Thus, learning the optimal policy either offline or online boils down to solving a system of equations (with suitably chosen test functions) via stochastic approximation to find v^*. Online learning of (57) is the same as the update rule: $v \in \arg\min \frac{1}{2}\|\Psi^v(x,a) - \lambda^{-1}\nabla_a Q(x,a)\|^2$.

Here we present Offline Continuous Q-Score Matching (CQSM) Martingale Loss Algorithm 2 in the infinite-horizon setting .

Algorithm 2 Offline Continuous Q-Score Matching (CQSM) Martingale Loss Algorithm

1: **Inputs:** Initial state x_0, a_0, horizon T, time step Δt, number of episodes N, number of mesh grids K, initial learning rates α_θ, α_v and a learning rate schedule function $l(\cdot)$, functional forms of parameterized action value function $Q^\theta(\cdot,\cdot)$ and score function $\Psi^v(\cdot,\cdot)$ and regularization parameter λ.
2: Initialize θ, v.
3: **for** episode $j = 1$ to N **do**
4: Initialize $k = 0$. Observe initial state x_0 and store $x_{t_k} \leftarrow x_0$.
5: Choose action by iteratively denoising $a^T \sim \mathcal{N}(0,1) \to a^0$ using $\Psi^v : a_{t_k} \sim \pi^v(x_{t_k})$;
6: Step environment $\{r_{t_k}, x_{t_{k+1}}\} = env(a_{t_k})$.
7: Obtain one observation $\{a_{t_k}, r_{t_k}, x_{t_k}\}_{k=0,\cdots,K-1}$.
8: For every $k = 0$ to $K-1$, compute

$$G_{t_k:T} = -e^{-\beta t_k} Q^\theta(x_{t_k}, a_{t_k}) + \sum_{i=k}^{K-1} e^{-\beta t_i}[r(x_{t_i}, a_{t_i}) - \frac{\lambda}{2}\|\Psi^v(x_{t_i}, a_{t_i})\|^2]\Delta t. \tag{58}$$

9: Update θ and v by

$$\theta \leftarrow \theta + l(j)\alpha_\theta \sum_{k=0}^{K-1} \frac{\partial Q^\theta}{\partial \theta}(x_{t_k}, a_{t_k}) G_{t_k:T}\Delta t. \tag{59}$$

$$v \leftarrow v + l(j)\alpha_v \sum_{k=0}^{K-1} \left[\sum_{i=k}^{K-1} \lambda \Psi^v(x_{t_i}, a_{t_i}) \frac{\partial \Psi^v}{\partial v}(x_{t_i}, a_{t_i})\Delta t\right] G_{t_k:T}\Delta t. \tag{60}$$

10: **end for**

E Scored Based Diffusion Models

Consider the forward SDE with state space $Y_t \in \mathbb{R}^d$ is defined as

$$\mathrm{d}Y_t = f(t, Y_t)\,\mathrm{d}t + g(t)\,\mathrm{d}B_t^a, Y_0 \sim \pi_{\text{data}}(\cdot), \tag{61}$$

where $f : \mathbb{R}_+ \times \mathbb{R}^d \to \mathbb{R}^d$ and $g : \mathbb{R}_+ \to \mathbb{R}_+$. Denote $\pi(t,\cdot)$ as the probability density of Y_t.

Set time horizon $T > 0$ to be fixed, and run the SDE (61) until time T to get $Y_T \sim \pi(T,\cdot)$. The time reversal $Y_t^{rev} := Y_{T-t}$ for $0 \le t \le T$ satisfies an SDE, under some mild conditions on f and g:

$$\mathrm{d}Y_t^{rev} = \left(-f(T-t, Y^{rev}) + \frac{1+\eta^2}{2} g^2(T-t)\nabla_y \log \pi(T-t, Y^{rev})\right)\mathrm{d}t + \eta g(T-t)\,\mathrm{d}B_t^a, \tag{62}$$

where $\nabla_y \log \pi(t,y)$ is known as the stein score function and $\eta \in [0,1]$ is a constant. In addition, a special but important case by taking $\eta = 0$ in the (62), this results to a flow ODE [42]:

$$\mathrm{d}Y_t^{rev} = \left(-f(T-t, Y^{rev}) + \frac{1}{2} g^2(T-t)\nabla_y \log \pi(T-t, Y^{rev})\right)\mathrm{d}t, \tag{63}$$

which can enable faster sampling and likelihood computation thanks to the deterministic generation.

Since the score function $\nabla_y \log \pi(t,y)$ is unknown, diffusion models learn a function approximation $s_\theta(t,y)$, parameterized by θ (usually a neural network), to the true score by minimizing the MSE or

the Fisher divergence (between the learned distribution and true distribution), evaluated by samples generated through the forward process (61):

$$\theta^* = \arg\max_{\theta} \mathbb{E}_{t\sim \text{Uni}(0,T)} \left\{ \lambda(t) \mathbb{E}_{y_0\sim\pi_{\text{data}}} \mathbb{E}_{y_t\sim\pi(t,\cdot|y_0)} \left[\| s_\theta(t,y_t) - \nabla_{y_t} \log \pi(t,y_t|y_0) \|_2^2 \right] \right\} \quad (64)$$

where $\lambda : [0,T] \to \mathbb{R}_{>0}$ is a chosen positive weighting function. When choosing the weighting functions as $\lambda(t) = g^2(t)$, this score matching objective is equivalent to maximizing an evidence lower bound (ELBO) of the log-likelihood.

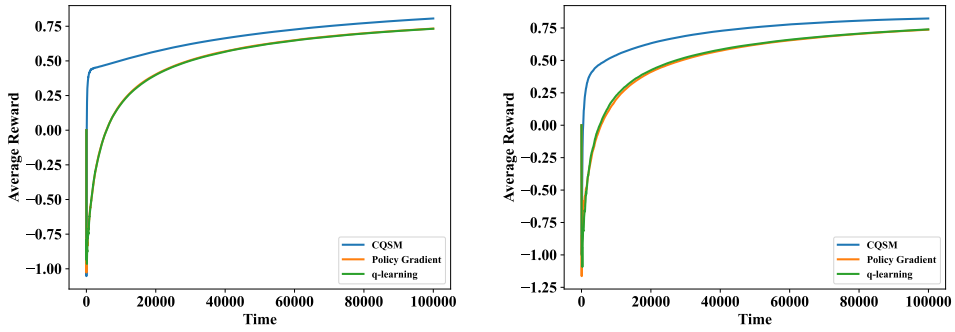
For action sampling, we view the action process as the reverse process. First we set $\sigma_a(t, X, a^t) = \sigma_t^2 I$ to untrained time dependent constants. Experimentally, we set $\beta_t = \sigma_t^2$ and $\alpha_t = 1 - \beta_t, \bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$. When applied to score matching with denoising diffusion probabilistic modeling (DDPM) [18], samples can be generated by starting from $a^T \sim \mathcal{N}(0, I)$ and following the reverse Markov chain as below

$$a^{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(a^t + \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \Psi(a_t, x_t) \right) + \sigma_t Z, Z \sim \mathcal{N}(0, I). \quad (65)$$

F Additional Experiments

F.1 Deterministic Case of LQ Control Tasks

We set $C = D = 0$ to eliminate stochasticity. Figure F.1 shows the running average reward of CQSM, PG and little q-learning. We observe that the resulting performance is comparable to the stochastic case.



(a) $\Delta t = 0.1$ (b) $\Delta t = 1$

Figure F.1: Running average rewards of three RL algorithms. A single state trajectory of length $T = 10^5$ is generated and discretized using two different step sizes: $\Delta t = 0.1$ in panel (a), $\Delta t = 1$ in panel (b).

F.2 Continuous Control Benchmark Tasks

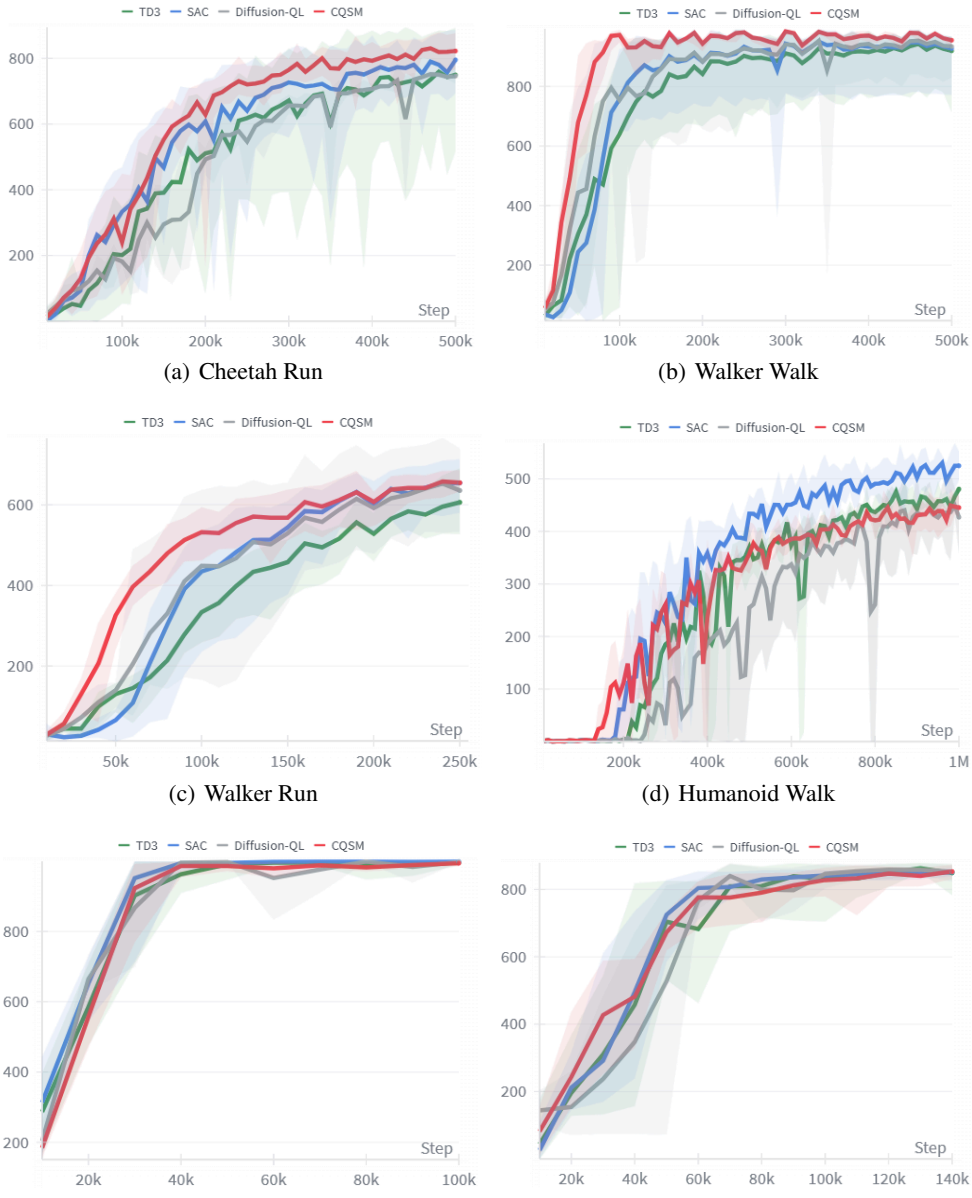
We evaluate CQSM on continuous control benchmarks from the DeepMind Control Suite. DeepMind Control Suite [51] is a set of control tasks implemented in MuJoCo [50]. We choose TD3 [9], SAC [16], and Diffusion-QL [55] as three baselines for comparison. Below, we first provide a brief review of prior methods.

Policy gradient methods seek to directly optimize the policy by computing gradients of the expected reward with respect to the policy parameters [45]. Deterministic policy gradient algorithms for MDPs (with discrete time and continuous action space) are developed in [41] (DPG) and later extended to incorporate deep neural networks in [29] (DDPG). Recent studies have focused on stochastic policies with entropy regularization, also known as the softmax method; see for example, [31] (A3C); [39] (PPO).

- TD3: [9] proposed Twin Delayed Deep Deterministic Policy Gradient (TD3), which mitigates overestimation bias in DDPG by using clipped double Q-learning and delayed policy updates. This yields more stable and accurate policy learning in continuous control tasks.
- SAC: The Soft Actor-Critic algorithm [16] optimizes a stochastic policy that maximizes both expected reward and policy entropy. The policy follows $\pi(a|x) \sim e^{\frac{1}{\lambda}Q(x,a)}$ and is reparameterized using a neural transformation of samples from a fixed distribution.
- Diffusion-QL: [55] integrates diffusion models with Q-learning by adding a term that maximizes action-values to the diffusion model's training loss. The final policy-learning objective is a linear combination of policy regularization and policy improvement.

We evaluate CQSM against the baselines (TD3, SAC, and Diffusion-QL) on a range of MuJoCo continuous control tasks, from high-dimensional domains (Cheetah Run, Walker Walk, Walker Run, Humanoid Walk) to simpler environments (Cartpole Balance, Cartpole Swingup). Each experiment is repeated with 10 random seeds, and we report the average episode return across runs (removing extremely poor results).

As shown in panels (a)-(d) of Figure F.2, CQSM matches or outperforms the TD3, SAC, and Diffusion-QL baselines, particularly in the early stages of training where it achieves higher rewards. This early performance gain highlights a key distinction between our approach and conventional actor-critic methods. Algorithms like SAC and TD3 may frequently sample actions with low Q-values in the initial stage because they do not utilize the action derivative Q_a, while Diffusion-QL can sometimes get stuck at suboptimal solutions. In contrast, our approach benefits from a more immediate policy adjustment via the learned score function, enabling the policy to better approximate the true optimal policy in the early stages. As a result, our method can achieve higher rewards in the early steps. In simpler tasks, shown in panels (e)-(f) of Figure F.2, CQSM achieves performance comparable to the baselines, demonstrating both its stability and general applicability across task complexities.



(a) Cheetah Run (b) Walker Walk

(c) Walker Run (d) Humanoid Walk

(e) Cartpole Balance (f) Cartpole Swingup

Figure F.2: Experimental Results Across A Suite of Six Continuous Control Tasks.