Expert Systems With Applications 317 (2026) 131838

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Risk-sensitive actor-critic with static spectral risk measures for online and offline reinforcement learning

Mehrdad Moghimi, Hyejin Ku*

Department of Mathematics and Statistics, York University, 4700 Keele Street, Toronto, M3J 1P3, Ontario, Canada

ARTICLE INFO

Keywords:
Distributional reinforcement learning
Risk-sensitive RL
Offline RL
Static spectral risk measure

ABSTRACT

The development of Distributional Reinforcement Learning (DRL) has introduced a natural way to incorporate risk sensitivity into value-based and actor-critic methods by employing risk measures other than expectation in the value function. While this approach is widely adopted in many online and offline RL algorithms due to its simplicity, the naive integration of risk measures often results in suboptimal policies. This limitation can be particularly harmful in scenarios where the need for effective risk-sensitive policies is critical and worst-case outcomes carry severe consequences. To address this challenge, we propose a novel framework for optimizing static Spectral Risk Measures (SRM), a flexible family of risk measures that generalizes objectives such as CVaR and Mean-CVaR, and enables the tailoring of risk preferences. Our method is applicable to both online and offline RL algorithms. We establish theoretical guarantees by proving convergence in the finite state-action setting. Moreover, through extensive empirical evaluations, we demonstrate that our algorithms consistently outperform existing risk-sensitive methods in both online and offline environments across diverse domains.

1. Introduction

Risk management plays a crucial role in sequential decision-making tasks across various domains, including finance, healthcare, and robotics. To address these risks, many approaches change the standard objective of maximizing expected returns to alternative risk measures, such as the Conditional Value-at-Risk (CVaR) (Bäuerle & Ott, 2011) or coherent risk measures (Tamar et al., 2017). Another strategy is to impose constraints, such as variance-based limits (Tamar et al., 2012) or dynamic risk measures (Chow & Pavone, 2013), to control exposure to worst-case scenarios.

Recently, Distributional Reinforcement Learning (DRL) has gained significant attention as a framework for risk-sensitive RL (RSRL) by estimating the entire distribution of returns rather than focusing solely on the expected value (Bellemare et al., 2017; Morimura et al., 2010). This approach offers a straightforward method for risk mitigation by employing a risk measure, such as a Mean-Variance or Distortion risk measure, instead of the expected value (Dabney et al., 2018a; Ma et al., 2020). However, within the DRL framework, applying a fixed risk measure at each time step does not necessarily result in policies that optimize static or dynamic risk measures over the entire trajectory (Lim & Malik, 2022). With such iterative risk measures, action selection at different states may deviate from the agent's overall risk preferences, potentially leading to suboptimal policies. For example, optimizing $\mathrm{CVaR}_{0.1}$ at a future state may not align with optimizing the worst 10% of returns from the initial state. This phenomenon, referred to as time inconsistency, is a well-documented challenge in risk-sensitive decision-making (Shapiro et al., 2014).

Due to the simplicity of incorporating iterative risk measures with DRL, this approach has also been adopted in offline RL (Ma et al., 2021; Urpí et al., 2021). In offline RL, policies are learned from previously collected data (Levine et al., 2020) and risk-sensitive offline RL with iterative risk measures not only face the same issues observed in the online setting but also encounter additional challenges unique to offline RL, such as the inability to explore new high-reward states outside the dataset and the distributional shift problem, where the policy is trained on one state-action distribution but evaluated on another. To address these issues, existing offline RL methods often employ techniques like policy constraints (Fujimoto et al., 2019; Nair et al., 2021; Peng et al., 2019) and value function regularization (Kumar et al., 2020).

Optimizing static risk measures offers a more interpretable solution by focusing on policies that maximize worst-case returns. Unlike dynamic or iterative risk measures, which assume that worst-case scenarios can occur at each step, static risk measures impose a limit on the worst-case scenarios that can occur over an entire episode (Chow et al., 2015). Consequently, with static risk measures, the risk level of the agent changes implicitly based on the observed path (Moghimi & Ku, 2025; Pflug & Pichler, 2016). This concept is illustrated in Fig. 1, where the

* Corresponding author.
E-mail addresses: moghimi@yorku.ca (M. Moghimi), hku@yorku.ca (H. Ku).

https://doi.org/10.1016/j.eswa.2026.131838
Received 3 July 2025; Received in revised form 31 January 2026; Accepted 23 February 2026
Available online 3 March 2026

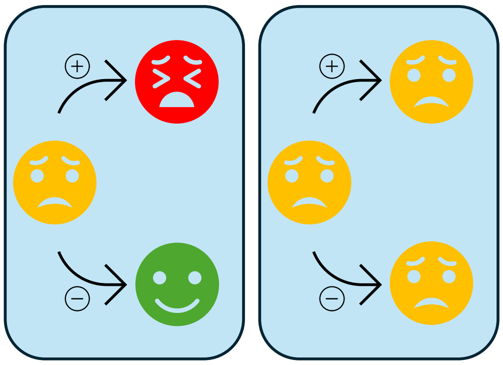


Fig. 1. Conceptual illustration of risk sensitivity dynamics. The symbols + and - represent favorable and unfavorable outcomes, respectively. The icon colors represent the agent's effective level of risk aversion: Red indicates high risk aversion, Green indicates low risk aversion, and Yellow represents a fixed/moderate level. **(Left)** Static Risk Measures: The agent's effective risk sensitivity is path-dependent, implicitly adapting based on the accumulated history to satisfy a global risk budget. **(Right)** Iterative Risk Measures: The agent enforces a fixed risk distortion at every time step (constant icon), independent of the observed history.

symbols (+ and -) represent favorable and unfavorable outcomes. Unlike iterative measures, which maintain a constant risk-aversion level (Right), a static risk measure (Left) allows the agent's effective risk-aversion to adapt based on the realization of outcomes. While this behavior may seem unintuitive, it reflects a key feature of static measures: after observing a favorable sequence of events, the agent becomes more cautious in subsequent steps to respect the overall bound on worst-case scenarios.

Although optimizing static CVaR has been the focus of many studies due to its simple formulation, one of its major drawbacks is that it exclusively focuses on the left tail of the return distribution, ignoring potential upsides. This limitation, often referred to as "Blindness to Success" (Greenberg et al., 2022), can result in suboptimal policies. To address this issue, as well as the previously mentioned challenges, we introduce a simple yet powerful actor-critic framework for optimizing static Spectral Risk Measures (SRM) that can be applied in both offline and online settings. By breaking down the steps required to optimize an SRM, we demonstrate how such a framework can be derived. Moreover, since SRM is a flexible risk measure that encompasses various measures, such as Mean-CVaR and the Exponential SRM, this framework fully leverages the potential of DRL.

In scenarios where avoiding worst-case outcomes is critical and collecting new data is costly, learning effective risk-sensitive policies from offline datasets is especially valuable, as it eliminates the need for environment interaction. Despite this importance, existing work on risk-sensitive offline reinforcement learning has largely focused on iterative risk measures. To the best of our knowledge, the optimization of static risk measures remains unexplored in the offline setting. A further advantage of learning risk-sensitive policies offline is the ability to tailor policies to different levels of risk aversion. In this regard, the flexibility of static spectral risk measures enables the learning of a diverse set of policies adapted to varying risk preferences. As we demonstrate in our experiments, our algorithms not only achieve strong performance under risk-sensitive criteria but also allow for the customization of policies to suit specific risk profiles.

Our framework can be utilized with both stochastic and deterministic policies. Stochastic policies are well-suited for environments with uncertainty and imperfect information due to their inherent randomness (Sutton & Barto, 2018). However, this randomness can degrade the performance of risk-averse policies, and the exploration advantages of stochastic policies are lost in the offline setting (Urpí et al., 2021). This motivates the use of deterministic policies in our framework. For this purpose, we utilize TD3 (Fujimoto et al., 2018) and TD3BC (Fujimoto & Gu, 2021) as high-performing online and offline RL algorithms with deterministic policies. Accordingly, in addition to our stochastic policy-based methods, AC-SRM for online learning and OAC-SRM for offline learning, we introduce TD3-SRM and TD3BC-SRM as their deterministic counterparts for online and offline settings, respectively. Our empirical results demonstrate that optimizing static SRM not only addresses theoretical concerns but also produces policies that outperform other risk-sensitive offline algorithms across various environments.

In summary, our key contributions are as follows:

- We propose a risk-sensitive actor-critic framework for optimizing the static Spectral Risk Measures, which is applicable to both online and offline learning.
- We prove the convergence of our algorithm in the online MDP setting with finite state and action spaces.
- We perform a comprehensive empirical analysis of our framework with both stochastic and deterministic policies in online and offline settings, demonstrating its overall effectiveness and superiority compared to existing methods.

2. Related work

2.1. Risk-sensitive online RL

Risk-sensitive reinforcement learning (RL) focuses on identifying policies that maximize risk-adjusted returns, formulated as $\max_{\pi\in\pi} \rho(Z^{\pi})$. In value-based methods, several approaches have been proposed: Bäuerle and Ott (2011) and Chow et al. (2015) study the Conditional Value at Risk (CVaR), while Bäuerle and Rieder (2014) focus on utility functions. In the distributional RL framework, Dabney et al. (2018a) explore the use of risk measures beyond expected value, such as distortion risk measures, for action selection. Further developments include optimizing static CVaR, as shown in Bellemare et al. (2023) and Lim and Malik (2022), and general statistical functionals of the return distribution, as discussed by Pires et al. (2025). In policy gradient methods, Chow et al. (2018) address CVaR-constrained problems, Tamar et al. (2012) study variance-based objectives, and Tamar et al. (2015) focus on optimizing CVaR. Dynamic coherent and convex risk measures have also been considered by Tamar et al. (2017) and Coache and Jaimungal (2023), respectively. Additionally, Bisi et al. (2022) propose a state-augmentation approach for optimizing risk measures such as mean-variance and utility functions, and Ma et al. (2020) incorporate risk sensitivity into SAC using a distributional critic.

There have been a few works that address the optimization of static SRM. Bäuerle and Glauner (2021) decompose the problem into two parts: identifying a risk function corresponding to the SRM, and then optimizing a policy under that risk function. They use a value-based method for policy optimization and rely on global optimization to learn the risk function. Kim et al. (2024) use SRM as the constraint of the optimization problem and employ a gradient-based method to learn the parameters of the risk function, which can be computationally expensive. More recently, Moghimi and Ku (2025) propose a value-based approach for optimizing SRM using distributional RL, and use a closed-form solution to compute the risk function. In this paper, we leverage this closed-form solution due to its simplicity but significantly advance the methodology in three key aspects. First, we propose a unified actor-critic framework that supports both stochastic and deterministic policies, enabling the method to scale to high-dimensional continuous control tasks where the previous value-based approach is not applicable. Second, we extend the framework to the offline setting by integrating policy constraints to specifically address the challenge of distributional shift. Third, we establish theoretical guarantees by proving the convergence of our policy optimization algorithm in the online tabular setting, providing a rigorous foundation for optimizing static spectral risk measures via gradient ascent.

2.2. Offline RL

Offline RL, or batch RL, involves learning policies from static datasets without further interaction with the environment. Addressing the key challenge of *distributional shift*, where the state-action distribution of the dataset differs from that of the learned policy, has led to a variety of approaches. These include policy constraint methods, such as Batch-Constrained Q-learning (BCQ) (Fujimoto et al., 2019) and Advantage-Weighted Actor-Critic (AWAC) (Nair et al., 2021), which constrain the learned policy to stay close to the behavior policy and reduce the risk of poor decisions in underexplored areas. Regularization techniques, like Conservative Q-Learning (CQL) (Kumar et al., 2020), address overestimation of actions not well supported by the dataset. Additionally, ensemble methods (Agarwal et al., 2020) improve robustness by aggregating multiple value estimates. These methods represent a few of the many strategies proposed to tackle offline RL challenges. For a comprehensive review, we refer readers to the surveys by Levine et al. (2020) and Prudencio et al. (2023).

2.3. Risk-sensitive Offline RL

Despite the importance of learning risk-averse policies from datasets, only a few works have addressed this problem. The first work in this area is the Offline Risk-Averse Actor-Critic (ORAAC) algorithm (Urpí et al., 2021) which extends the BCQ algorithm (Fujimoto et al., 2019) with a distributional critic and uses risk-sensitive action-selection. Similarly, the Conservative Offline Distributional Actor-Critic (CODAC) algorithm (Ma et al., 2021) extends the CQL algorithm (Kumar et al., 2020) with a distributional critic and learns risk-averse policies by utilizing the return-distribution of state-action pairs. Bai et al. (2022) propose a new architecture called Monotonic Quantile Network (MQN) to improve the learning of the distributional value function in the offline setting and uses the CVaR of this value function to learn risk-averse policies. Finally, Rigter et al. (2023) proposes an actor-critic algorithm that optimizes a dynamic risk measure, which requires learning the MDP model from the dataset. While model-based algorithms offer improved sample efficiency in the online setting by leveraging learned environment dynamics, they are computationally expensive, and the model bias can lead to compounding errors over multiple time-steps, which can degrade policy performance.

3. Preliminary studies

In this work, we consider a Markov Decision Process (MDP) defined by the tuple $(\mathcal{X}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma, \xi_0)$, where $\mathcal{X}$ and $\mathcal{A}$ represent the state and action spaces. The reward function $\mathcal{R} : \mathcal{X} \times \mathcal{A} \to \mathscr{P}(\mathbb{R})$ specifies a distribution over rewards, while the transition function $\mathcal{P} : \mathcal{X} \times \mathcal{A} \to \mathscr{P}(\mathcal{X})$ defines the probability of moving to the next state. The initial state follows the distribution ξ_0, and $\gamma \in [0, 1)$ is the discount factor. We assume that rewards are bounded within $[R_{\min}, R_{\max}]$, where $R_{\min} \geq 0$.

A Markov stochastic policy is a mapping $\pi : \mathcal{X} \to \mathscr{P}(\mathcal{A})$ that assigns a probability distribution over actions given a state. We denote the space of Markov policies as Π and state occupancy measure under policy π as $d^{\pi}_{\xi_0}$. The discounted return when starting from state x, taking action a, and following policy π is defined as

$$G^{\pi}(x, a) := \sum_{t=0}^{\infty} \gamma^t R_t,$$

where rewards R_t are drawn from $\mathcal{R}(X_t, A_t)$, actions A_t are sampled from $\pi(\cdot|X_t)$, and next states X_{t+1} follow the transition dynamics $\mathcal{P}(X_t, A_t)$. The return from state x under policy π is denoted as $G^{\pi}(x)$, while G^{π} represents the return variable obtained by first sampling the initial state $x_0 \sim \xi_0$ and then generating a trajectory under policy π. $G^{\pi}(x_0)$ denotes the corresponding return random variable conditioned on a fixed initial state x_0.

For a parameterized policy π_θ, the value of the policy is defined as

$$J(\pi_\theta) = \mathbb{E}\left[G^{\pi_\theta}\right],$$

and our goal is to find the optimal parameters θ^* that maximize this value:

$$\theta^* = \arg\max_{\theta} J(\pi_\theta).$$

The Policy Gradient Theorem (Sutton et al., 1999) provides a way for updating the policy π_θ by computing the gradient of the performance objective $J(\pi_\theta)$, given by:

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{x \sim d^{\pi_\theta}_{\xi_0},\, a \sim \pi_\theta}\left[\nabla_\theta \log \pi_\theta(a \mid x) Q^{\pi_\theta}(x, a)\right], \tag{1}$$

where $d^{\pi}_{\xi_0}(x) = \mathbb{E}_{x_0 \sim \xi_0}\left[d^{\pi}_{x_0}(x)\right]$ is the discounted state visitation distribution under initial distribution ξ_0 and $d^{\pi}_{x_0}(x)$ is defined as $d^{\pi}_{x_0}(x) := (1-\gamma)\sum_{t=0}^{\infty} \gamma^t \Pr^{\pi}(x_t = x \mid x_0)$ where $\Pr^{\pi}(x_t = x \mid x_0)$ is the probability of visiting state x after starting at x_0 and executing π. The Q-value function $Q^{\pi}(x, a)$ represents the expected return after taking action a in state x:

$$Q^{\pi}(x, a) = \mathbb{E}\left[G^{\pi}(x, a)\right].$$

On-policy learning methods update the policy using data collected by executing the current policy in the environment. In contrast, off-policy learning allows learning about a target policy π_θ using data generated by a different behavior policy. This enables more sample-efficient learning by reusing past experiences.

In online off-policy learning, experiences collected through ongoing interaction are stored in a replay buffer, from which the agent samples to update its policy. In contrast, offline RL (also known as batch RL) involves learning solely from a fixed dataset of transitions generated by an unknown behavior policy. In this setting, the agent has no further interaction with the environment, and the key challenge is to learn a high-performing policy purely from static data.

3.1. Distributional RL

Distributional reinforcement learning (Bellemare et al., 2017; Morimura et al., 2010) extends standard RL by modeling the full distribution of returns $G^{\pi}(x, a)$, denoted as $\eta^{\pi}(x, a)$. This return distribution is the unique fixed point of the distributional Bellman operator $\mathcal{T}^{\pi} : \mathscr{P}(\mathbb{R})^{\mathcal{X}\times\mathcal{A}} \to \mathscr{P}(\mathbb{R})^{\mathcal{X}\times\mathcal{A}}$, defined as

$$(\mathcal{T}^{\pi}\eta)(x, a) = \mathbb{E}_{\pi}\left[(b_{R,\gamma})_{\#}\eta(X', A') \mid X = x, A = a\right],$$

where $A' \sim \pi(\cdot)$ and $b_{r,\gamma}(z) = r + \gamma z$. The push-forward distribution $(b_{R,\gamma})_{\#}\eta(X', A')$ represents the return distribution of $(b_{R,\gamma})G(X', A')$. The operator can also be written in terms of return variables:

$$G^{\pi}(x, a) = R(x, a) + \gamma \mathbb{E}_{x' \sim \mathcal{P}(x'|x,a), a' \sim \pi(x')}\left[G^{\pi}(x', a')\right].$$

In this work, we model the return distribution using quantiles and update our estimates via quantile regression, following the QR-DQN approach (Dabney et al., 2018b). Given cumulative probabilities $\tau_i = i/N$ for $i = 0, \ldots, N$, the return distribution is parameterized as

$$\eta_q(x, a) = \frac{1}{N}\sum_{i=1}^{N} \delta_{q_i(x,a)},$$

where δ_z denotes the Dirac delta distribution centered at z, and the support points are defined by

$$q_i(x, a) = F^{-1}_{G(x,a)}(\hat{\tau}_i), \quad \hat{\tau}_i = (\tau_{i-1} + \tau_i)/2, \quad 1 \leq i \leq N.$$

To estimate the quantiles of a distribution ν, we minimize the quantile regression Huber loss, defined as:

$$\mathcal{L}(G, Z) = \sum_{i=1}^{N} \mathbb{E}_{Z\sim\nu}\left[l^{\kappa}_{\hat{\tau}_i}(Z - q_i)\right], \tag{2}$$

where $l^{\kappa}_{\tau}(u) = |\tau - \mathbb{1}\{u < 0\}| \cdot \mathcal{L}^{huber}_{\kappa}(u)$ and $\mathcal{L}^{huber}_{\kappa}(u)$ denotes the Huber loss (Huber, 1992), which combines squared and absolute error to avoid constant gradients near zero:

$$\mathcal{L}^{huber}_{\kappa}(u) = \begin{cases} \frac{1}{2}u^2, & \text{if } |u| \le \kappa, \\ \kappa\left(|u| - \frac{1}{2}\kappa\right), & \text{otherwise.} \end{cases}$$

3.2. Risk-sensitive Policies

The availability of return distributions in actor-critic methods with a distributional critic enables the use of risk measures other than the expectation when estimating state-action values. A simple approach is to apply a risk measure ρ at each step by replacing the Q-values in Eq. (1) with:

$$Q^{\pi}(x, a) = \rho(G^{\pi}(x, a)).$$

This method has been widely used in both online (Dabney et al., 2018a; Ma et al., 2020) and offline (Ma et al., 2021; Urpí et al., 2021) RL for risk-sensitive policy optimization.

However, as discussed in Lim and Malik (2022), this iterative risk application can be problematic. It can lead to overly optimistic or overly conservative estimates, and optimizing a policy based on these Q-values does not necessarily align with maximizing either the static risk-adjusted value:

$$J(\pi) = \rho(G^{\pi}),$$

or the dynamic risk-adjusted value:

$$J_d(\pi) = \rho\big(R_0 + \gamma\rho\big(R_1 + \gamma\rho\big(R_2 + \gamma(\cdots)\big)\big)\big).$$

Dynamic risk measures, which repeatedly apply a risk function at each step, introduce further challenges. Computing policy gradients in this setting require knowledge of the MDP model to adjust transition probabilities, making both optimization and interpretation more difficult. Additionally, selecting an appropriate risk parameter is nontrivial, as the fixed risk level can lead to overly conservative policies. Given these issues, we instead focus on directly optimizing the static risk-adjusted value.

3.3. Spectral Risk Measure

Let $Z \in \mathcal{Z}$ denote the random variable representing the return, and let $\rho : \mathcal{Z} \to \mathbb{R}$ be a risk measure, where $\rho(Z)$ is the risk-adjusted value of Z. The quantile function of Z is defined as

$$F^{-1}_{Z}(u) = \inf\{z \in \mathbb{R} \mid F_Z(z) \ge u\}, \quad u \in [0, 1],$$

where $F_Z(z) = \mathbb{P}(Z \le z)$ is the cumulative distribution function (CDF). While expectation assigns equal weights to all quantiles, a natural extension is to assign different weights based on risk preferences. Spectral Risk Measure (SRM), introduced by Acerbi (2002), formalizes this idea by defining

$$\mathrm{SRM}_{\phi}(Z) = \int_0^1 F^{-1}_{Z}(u)\phi(u)\,\mathrm{d}u,$$

where $\phi : [0, 1] \to \mathbb{R}_+$ is a left-continuous, non-increasing risk spectrum function satisfying $\int_0^1 \phi(u)\,du = 1$. The function $\phi(u)$ expresses the risk preference across different quantiles of the return distribution.[1] Various risk spectrums have been explored in the literature, which are visualized in Fig. 2:

- CVaR: $\phi_{\alpha}(u) = \frac{1}{\alpha}\mathbb{1}_{[0,\alpha]}(u)$,
- Mean-CVaR (MC): $\phi_{\alpha,\omega}(u) = \omega\mathbb{1}_{[0,1]}(u) + (1-\omega)\frac{1}{\alpha}\mathbb{1}_{[0,\alpha]}(u)$,

[1] Dabney et al. (2018a) employ the Distortion Risk Measure (DRM). A coherent DRM, characterized by a concave distortion function g, is equivalent to the Spectral Risk Measure where $g'(u) = \phi(u)$ (Henryk & Silvia, 2006).

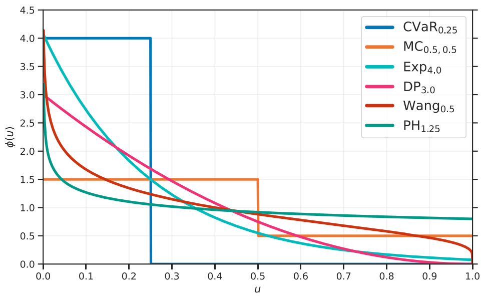


Fig. 2. Illustration of various risk spectrums.

- Exponential SRM (Exp): $\phi_{\alpha}(u) = \frac{\alpha e^{-\alpha u}}{1-e^{-\alpha}}$,
- Dual Power (DP): $\phi_{\alpha}(u) = \alpha(1-u)^{\alpha-1}$,
- Wang: $\phi_{\alpha}(u) = e^{(\alpha\Phi^{-1}(u) - \alpha^2/2)}$,
- Proportional Hazard (PH): $\phi_{\alpha}(u) = \frac{1}{\alpha}u^{(1/\alpha - 1)}$.

where $\Phi^{-1}(u)$ denotes the inverse CDF of a standard Normal distribution (Wang, 1995, 2000).

Our goal in this work is to find a policy that maximizes its static risk-adjusted value. In the next section, we show that achieving this requires using the supremum representation of SRM, which applies to any SRM with a bounded spectrum:

$$\mathrm{SRM}_{\phi}(Z) = \sup_{h \in \mathcal{H}} \left\{ \mathbb{E}[h(Z)] + \int_0^1 \hat{h}(\phi(u))\mathrm{d}u \right\}. \tag{3}$$

Here, $\mathcal{H}$ is the set of concave functions, and $\hat{h}$ is the concave conjugate of h (Pichler, 2015). The supremum is attained by the function $h_{\phi,Z} : \mathbb{R} \to \mathbb{R}$, given by

$$h_{\phi,Z}(z) = \int_0^1 \left[F^{-1}_{Z}(\alpha) + \frac{1}{\alpha}(z - F^{-1}_{Z}(\alpha))^{-} \right] \mu(\mathrm{d}\alpha), \tag{4}$$

where the probability measure $\mu : [0, 1] \to [0, 1]$ satisfies

$$\phi(u) = \int_u^1 \frac{1}{\alpha}\mu(\mathrm{d}\alpha), \tag{5}$$

and $h_{\phi,Z}(z)$ satisfies

$$\int_0^1 \hat{h}_{\phi,Z}(\phi(u))\,\mathrm{d}u = 0.$$

4. Method

In this section, we present a unified actor-critic framework for optimizing static Spectral Risk Measures (SRM) through a bi-level optimization approach. By leveraging the supremum representation of SRMs, we decompose the objective into an inner loop that optimizes the policy for a fixed risk function and an outer loop that updates the risk function based on the return distribution. This framework supports both online and offline settings, as well as stochastic and deterministic policies. We first introduce AC-SRM for online learning, followed by OAC-SRM for offline learning with policy constraints, and finally extend our method to deterministic policy gradients via TD3-SRM and TD3BC-SRM.

To be more precise, we aim to optimize the static risk-adjusted return of a policy under a Spectral Risk Measure (SRM), given by:

$$J(\pi) = \mathrm{SRM}_{\phi}(G^{\pi}).$$

We leverage the supremum representation of the SRM (Eq. (3)) to reformulate this objective as a bi-level optimization problem. To that end, we define an auxiliary value function for a fixed risk function $h \in \mathcal{H}$:

$$J(\pi, h) := \mathbb{E}\big[h(G^{\pi})\big] + \int_0^1 \hat{h}(\phi(u))\,\mathrm{d}u.$$

Using this formulation, the overall optimization problem becomes:

$$\max_{\pi\in\Pi} J(\pi) = \max_{\pi\in\Pi}\max_{h\in\mathcal{H}} J(\pi, h) = \max_{h\in\mathcal{H}}\left(\max_{\pi\in\Pi} J(\pi, h)\right). \tag{6}$$

For clarity, we omit the subscript h in π_h throughout the remainder of the paper, and simply write π when the dependence on h is clear from context.

For the inner optimization, we adopt an actor-critic method with a distributional critic, enabling simultaneous learning of both the optimal policy and its associated return distribution. In this context, the optimal policy is Markovian in an extended state space $\bar{\mathcal{X}} := \mathcal{X}\times\mathcal{S}\times\mathcal{C}$ (Bäuerle & Glauner, 2021), where $\mathcal{S}$ represents the space of accumulated discounted rewards, and $\mathcal{C}$ captures the space of discount factors up to the decision time. The state transitions are also defined as:

$$S_{t+1} = S_t + C_t R_t, \quad C_{t+1} = \gamma C_t,$$

where we initialize $S_0 = 0$ and $C_0 = 1$.

Crucially, this formulation addresses the time-inconsistency issue inherent in static risk measures. While static measures generally do not admit a recursive decomposition, fixing h transforms the inner problem into maximizing the expectation $\mathbb{E}[h(S + cG^\pi)]$. On the augmented state space, this objective satisfies the Tower Property of expectation, allowing the value function to be defined recursively. Consequently, the return distribution G^π can be estimated via standard distributional Bellman updates, and the policy can be optimized using consistent actor-critic methods without violating the Dynamic Programming Principle.

For the outer optimization, Eq. (4) suggests that the function h should be updated based on the return distribution of the initial state. This is where the distributional critic plays a key role: it provides an estimate of this return distribution, which is then used to refine h. Proposition 4.1 explains how using the quantile representation of the return distribution modifies the definition of the function h. The detailed proof is provided in Appendix C.

Proposition 4.1. *Let Z be a bounded random variable with quantile representation defined by $q_i = F_Z^{-1}(\hat{\tau}_i)$ for $i = 1,\dots,N$, where $\tau_i = i/N$ and $\hat{\tau}_i = (\tau_{i-1}+\tau_i)/2$. Then, the function h in Eq. (4) can be approximated by the piecewise linear function*

$$\tilde{h}_{\phi,Z}(z) := \sum_{i=1}^{N} w_i\left(q_i + \frac{1}{\hat{\tau}_i}(z - q_i)^-\right),$$

where the weights are given by $w_i = \hat{\tau}_i\big(\phi(\tau_{i-1}) - \phi(\tau_i)\big)$. Furthermore, the approximation error converges to zero as $N\to\infty$.

This framework is summarized in Algorithm 1, and a conceptual diagram is illustrated in Fig. 3. In this algorithm, $\bar{\mathcal{T}}^\pi : \mathscr{P}(\mathbb{R})^{\bar{\mathcal{X}}\times\mathcal{A}} \to \mathscr{P}(\mathbb{R})^{\bar{\mathcal{X}}\times\mathcal{A}}$, represents the distributional Bellman operator for the extended MDP.

Algorithm 1: Static spectral risk actor-critic framework: Bi-level optimization.

```
Input: A random initialization of $G^{\pi_0}$
for $k = 0, 1, \cdots$ do
    Step 1: // The Closed-form Solution in Eq. (4)
        $h_{k+1} = \arg\max_h J(\pi_k, h) = \tilde{h}_{\phi, G^{\pi_k}}$
    Step 2: // The Inner Optimization
        $\pi_{k+1} = \arg\max_\pi J(\pi, h_{k+1})$
        $G^{\pi_{k+1}} = \bar{\mathcal{T}}^{\pi_{k+1}} G^{\pi_{k+1}}$
end
```

In Step 2 of Algorithm 1, we use a distributional critic to estimate the return distribution for state-action pairs. The corresponding Q-value function for policy gradient is defined as:

$$Q_h^\pi(\bar{x}, a) := \mathbb{E}\big[h(s + cG^\pi(\bar{x}, a))\big]/c, \tag{7}$$

where the division by $c = \gamma^t$ offsets the discounting effect applied to $G^\pi(\bar{x}, a)$. To build intuition for the Q-value function, we examine the structure of h from Eq. (4). The integrand can be rewritten as:

$$\begin{aligned} K_\alpha &+ \frac{1}{\alpha}\min\left(s + cG^\pi(\bar{x}, a), q_\alpha\right) \\ &= c\left(\bar{K}_\alpha(s, c) + \frac{1}{\alpha}\min\left(G^\pi(\bar{x}, a), \frac{q_\alpha - s}{c}\right)\right), \end{aligned} \tag{8}$$

where $q_\alpha = F_{G^\pi}^{-1}(\alpha)$ is the quantile function of G^π, and $K_\alpha := \frac{\alpha q_\alpha - q_\alpha}{\alpha}$ and $\bar{K}_\alpha(s, c) := \frac{\alpha q_\alpha - q_\alpha + s}{\alpha c}$ are constant terms for a given α and (s, c), independent of the action. This formulation highlights that the factor c can be factored out of h, justifying the division by c in Eq. (7).

Eq. (8) also demonstrates how the extended state variables s and c allow the function h to apply the same risk preference at different time-steps. Comparing the return G^π to returns received at future time-steps requires alignment to a common reference frame. On the left-hand side of the equation, the term $s + cG^\pi(\bar{x}, a)$ adjusts the future return to the initial time-step by incorporating the discount factor c and past return s. On the right-hand side, the term $(q_\alpha - s)/c$ represents the adjusted value of q_α at time t, enabling a direct comparison with $G^\pi(\bar{x}, a)$.

4.1. Risk-sensitive policy learning in the online setting: AC-SRM

We begin with AC-SRM (Actor-Critic with Spectral Risk Measure), designed for the online reinforcement learning setting, where the agent interacts with the environment to collect data during training. AC-SRM directly optimizes a static spectral risk measure (SRM) of the return, allowing the agent to learn policies that reflect specific risk preferences. Using the Q-value function from Eq. (7), the state value function is computed as $V_h^\pi(\bar{x}) = \mathbb{E}_{a\sim\pi(a|\bar{x})}\big[Q_h^\pi(\bar{x}, a)\big]$, and the advantage function is defined as $A_h^\pi(\bar{x}, a) := Q_h^\pi(\bar{x}, a) - V_h^\pi(\bar{x})$. With this advantage function, the gradient of the objective with respect to the parameters of policy π_θ can be expressed as follows:

$$\nabla_\theta J(\pi_\theta, h) = \mathbb{E}_{d_{\xi_0}^{\pi_\theta}, \pi_\theta}\left[\nabla_\theta \log \pi_\theta(a \mid \bar{x}) A_h^{\pi_\theta}(\bar{x}, a)\right]/(1-\gamma), \tag{9}$$

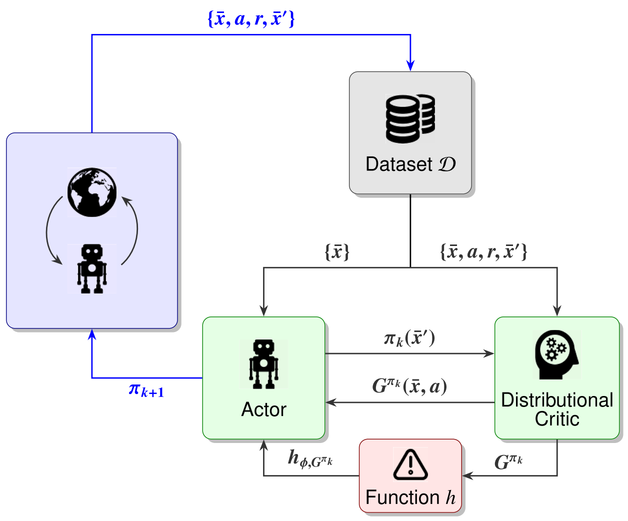
where $d_{\xi_0}^{\pi_\theta}$ denote the state occupancy measure when following policy π_θ. The following theorem establishes the convergence of this policy gradient method in the finite states and actions setting, with the proof available in Appendix D. Notably, the assumption of finite extended states implies finite horizon and that the rewards are drawn from a finite set $\hat{\mathcal{R}} \subset [R_{\min}, R_{\max}]$.

Theorem 4.2. *In the tabular setting (finite states and actions), let the policy π_θ be parameterized by θ using a softmax function, and let $J(\pi_\theta, h)$ be the performance objective. Assume the parameters θ are updated according to the Natural Policy Gradient (NPG) scheme (Kakade, 2001), based on the policy gradient given in Eq. (9). If the updates use a learning rate η_t satisfying the Robbins-Monro condition ($\sum_t \eta_t = \infty, \sum_t \eta_t^2 < \infty$), then π_θ converges to the optimal policy of the inner optimization within the class of representable softmax policies Π, i.e. $\pi_h^* := \arg\max_{\pi_\theta\in\Pi} J(\pi_\theta, h)$.*

While Theorem 4.2 discusses the convergence of the inner optimization, our next theorem discusses the convergence of the overall approach, with the proof available in Appendix E.

Theorem 4.3. *The policy updates in Algorithm 1 yield monotonic improvement of the objective $J(\pi)$. When using parameterized policies π_{θ_k}, the policy improvement is monotonic as well, meaning $J(\pi_{\theta_{k+1}}) \ge J(\pi_{\theta_k})$. Additionally, the sequence $\{J(\pi_{\theta_k})\}_{k=1,\dots}$ converges, and $\liminf_k \|\nabla_\theta J(\pi_{\theta_k})\| = 0$.*

Algorithm 2 presents the practical implementation of our framework, utilizing neural network function approximation to scale to high-dimensional state spaces. To support this approximation and ensure training stability, we incorporate standard deep RL techniques. Specifically, to mitigate Q-value overestimation, we employ two critic networks (Hasselt, 2010). Furthermore, we use a target network and reduce the frequency of policy updates (Fujimoto et al., 2018; Mnih et al., 2015).



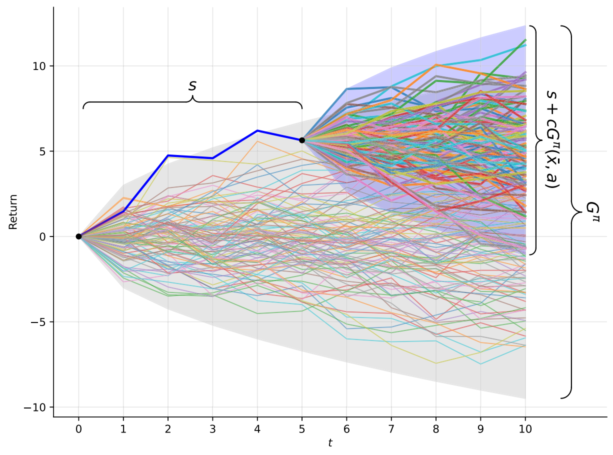


Fig. 3. **(Left)** Diagram of our framework. The blue connections indicating the interaction of the new policy with the environment and storing the new interaction in the dataset are not present in the offline setting. Finding the solution of the outer optimization can be interpreted as updating the actor's perception of the initial state's return distribution (G^{π_k}), which is then combined with the actor's risk preference ϕ to calculate the Q values. **(Right)** The static SRM is defined for G^{π}, so calculating the value of a future state-action requires an adjustment of $G^{\pi}(\bar{x}, a)$ to the initial timestep using c and s.

4.2. *Offline risk-sensitive learning with policy constraints: OAC-SRM*

We next present OAC-SRM (Offline Actor-Critic with Spectral Risk Measure), which extends our framework to the offline reinforcement learning setting, where the agent must learn solely from a fixed dataset $\mathcal{D}$ generated by an unknown behavior policy π_β without further interaction with the environment. To mitigate distributional shift and reduce extrapolation error, OAC-SRM incorporates a policy constraint that encourages the learned policy to stay close to the behavior policy.

This framework is especially valuable in multi-stakeholder or high-stakes applications, where risk preferences vary across users or tasks. For example, in financial portfolio management, clients often exhibit different levels of risk aversion. In wealth management or portfolio optimization, historical market data is often the only source available (i.e., the setting is inherently offline). By optimizing policies under different spectral risk measures using the same offline dataset, our framework enables the development of investment strategies tailored to individual risk profiles.

By incorporating this constraint into the policy optimization problem, we arrive at the following objective:

$$\max_{\pi \in \Pi} \mathrm{SRM}_\phi(G^\pi) \quad \text{s.t.} \quad \mathrm{D}_{\mathrm{KL}}(\pi, \pi_\beta) < \epsilon.$$

Here, $\mathrm{D}_{\mathrm{KL}}(\pi, \pi_\beta)$ denotes the expected Kullback—Leibler divergence between the learned policy and the behavior policy, defined as $\mathbb{E}_{\bar{x}\sim\mathcal{D}}[\mathrm{D}_{\mathrm{KL}}(\pi(\cdot|\bar{x})||\pi_\beta(\cdot|\bar{x}))] = \mathbb{E}_{\bar{x}\sim\mathcal{D}, a\sim\pi(\cdot|\bar{x})}\left[\log \frac{\pi(a|\bar{x})}{\pi_\beta(a|\bar{x})}\right]$. As in the previous section, we can use the supremum form of the SRM. Since the constraint does not depend on the function h, we can reformulate the problem as an inner-outer optimization:

$$\max_{h \in \mathcal{H}} \left(\max_{\pi \in \Pi} J(\pi, h) \quad \text{s.t.} \quad \mathrm{D}_{\mathrm{KL}}(\pi, \pi_\beta) < \epsilon \right).$$

The following theorem closely follows the results of Peng et al. (2019) and Nair et al. (2021), which derive policy updates under a KL divergence constraint. Our formulation adapts this idea to the risk-sensitive setting by incorporating the risk-adjusted advantage $A^{\pi}_h(\bar{x}, a)$, and we treat the Lagrange multiplier λ as a tunable hyperparameter. The selection of λ follows standard offline RL practices, balancing the trade-off between maximizing the risk-adjusted return and staying within the data support. In practice, we recommend using the default values established in the baseline algorithms, as our sensitivity analysis (Appendix B) indicates these values provide a robust performance, with degradation occurring as λ deviates significantly from this range.

This adaptation enables us to retain the tractability of the update rule while aligning it with the underlying risk-sensitive objective. Our formulation thus generalizes the advantage-weighted actor-critic framework to account for risk preferences in the offline policy learning. The proof of this theorem is available in Appendix F.

Theorem 4.4. *For policy search in the inner optimization, the policy constraint can be imposed implicitly (without approximating π_β) using the following policy update rule:*

$$\nabla_\theta J(\pi_\theta, h) = \mathbb{E}_{\bar{x}, a\sim\mathcal{D}}\left[\nabla_\theta \log \pi_\theta(a \mid \bar{x}) \exp\left(\frac{1}{\lambda} A^{\pi_\theta}_h(\bar{x}, a)\right)\right] / (1-\gamma), \tag{10}$$

where the Lagrange multiplier λ is treated as a hyperparameter.

The update of the function h in the outer optimization follows the same procedure as in the online setting. Although the distributional critic is less accurate in the offline setting, we empirically show in Section 5 that our algorithm provides an effective approach for learning risk-sensitive policies in offline settings. The detailed algorithm for OAC-SRM is outlined in Algorithm 3.

4.3. *Risk-sensitive deterministic policies: TD3-SRM and TD3BC-SRM*

Finally, we extend our framework to deterministic policies and introduce TD3-SRM and TD3BC-SRM, which build on TD3 (Fujimoto et al., 2018) and TD3BC (Fujimoto & Gu, 2021) for online and offline settings, respectively. Deterministic policies are well-suited for continuous control tasks and are particularly effective in offline learning, where the benefits of stochastic exploration diminish. To support such settings, we derive a risk-sensitive deterministic policy gradient based on the spectral risk-adjusted value function. Note that, unlike stochastic actors which can be tailored for both discrete and continuous action spaces (e.g., using softmax and Gaussian parameterizations, respectively), deterministic actors are limited to MDPs with continuous action spaces.

Our proposed TD3-SRM retains the core components of TD3, including two distributional critics and a deterministic actor. The critic estimates the return distribution via quantile regression, and the actor is trained to maximize the SRM-adjusted Q-value, defined by applying a spectral risk measure to the estimated quantiles. In the offline setting, we adapt this framework as TD3BC-SRM, where the actor objective is augmented with a behavior cloning loss to keep the learned policy close to the behavior policy.

Algorithm 2: AC-SRM.

```
Input     : Batch size $M$, Number of quantiles $N$, Policy
            update frequency $d$, Target smoothing
            coefficient $\nu$, Number of policy updates $T_{inner}$,
            Number of function $h$ updates $T_{outer}$
Initialize : Critic networks $G_{\theta_1}, G_{\theta_2}$ and Actor network
            $\pi_\theta = \mathcal{N}(f_\theta, \sigma)$ or $\pi_\theta = \text{Categorical}(\text{softmax}(f_\theta))$
            with random parameters $\theta_1, \theta_2, \theta$, Target
            network parameters $\theta'_1 \leftarrow \theta_1, \theta'_2 \leftarrow \theta_2, \theta' \leftarrow \theta$,
            Dataset $\mathcal{D}$
for 1 to $T_{outer}$ do
    // Update risk function
    $h = \tilde{h}_{\phi, G_{\theta_1}(x_0, \pi_\theta(x_0))}$
    for $t = 1$ to $T_{inner}$ do
        // Collect New Data
        Observe state $\bar{x}$, select action $a \sim \pi_\theta(\bar{x})$
        Execute $a$, observe reward $r$ and next state $\bar{x}'$
        Store transition $(\bar{x}, a, r, \bar{x}')$ into $\mathcal{D}$
        // Update Critic
        Sample mini-batch of $M$ transitions $(\bar{x}, a, r, \bar{x}')$
          from $\mathcal{D}$
        foreach transition in the mini-batch do
            Sample target action:
                $a' \sim \pi_{\theta'}(\bar{x}')$
            Compute Q-values for $k = 1, 2$:
                $Q_k = \mathbb{E}\left[h\left(s' + c' G_{\theta'_k}(\bar{x}', a')\right)\right] / c'$
            Select target quantile set:
                $G'(\bar{x}', a') = \begin{cases} G_{\theta'_1}(\bar{x}', a') & \text{if } Q_1 \le Q_2 \\ G_{\theta'_2}(\bar{x}', a') & \text{otherwise} \end{cases}$
            Compute target quantiles:
                $Y(\bar{x}, a) = r + \gamma G'(\bar{x}', a')$
        end
        Minimize quantile regression loss
        $\mathcal{L}(G_{\theta_k}, Y), k = 1, 2$ (Eq. (2)) for the mini-batch
        // Update Actor
        if $t \bmod d = 0$ then
            Compute the advantage function:
                $A(\bar{x}, a) = Q_1(\bar{x}, a) - \mathbb{E}_{\tilde{a} \sim \pi_{\theta'}}\left[Q_1(\bar{x}, \tilde{a})\right]$
            Update $\theta$ using the advantage function and
            the policy gradient in Eq. (9)
            Update target networks:
                $\theta'_k \leftarrow \nu\theta_k + (1-\nu)\theta'_k, k = 1, 2 \quad \theta' \leftarrow \nu\theta + (1-\nu)\theta'$
        end
    end
end
```

Building on the definition of the action value function (Eq. (7)) and the deterministic policy gradient (Silver et al., 2014), we derive a risk-sensitive deterministic policy gradient, which can be approximated as follows.

$$\begin{aligned}
&\nabla_\theta J(\pi_\theta, h) \\
&\approx \mathbb{E}_{\bar{x} \sim d_{\xi_0}^{\pi_\theta}}\left[\nabla_\theta \pi_\theta(\bar{x}) \nabla_a Q_h^{\pi_\theta}(\bar{x}, a)\Big|_{a=\pi_\theta(\bar{x})}\right] \\
&= \mathbb{E}_{\bar{x} \sim d_{\xi_0}^{\pi_\theta}}\left[\nabla_\theta \pi_\theta(\bar{x}) \mathbb{E}\left[h'(s + cG^{\pi_\theta}(\bar{x}, \pi_\theta(\bar{x}))) \nabla_a G^{\pi_\theta}(\bar{x}, a)\Big|_{a=\pi_\theta(\bar{x})}\right]\right]. \quad (11)
\end{aligned}$$

Since $h_{\phi, G^{\pi_\theta}}(z)$ is differentiable almost everywhere with derivative $h'_{\phi, G^{\pi_\theta}}(z) = \phi\big(F_{G^{\pi_\theta}}(z)\big)$, we have:

$$h'(s + cG^{\pi_\theta}(\bar{x}, \pi_\theta(\bar{x}))) = \phi\big(F_{G^{\pi_\theta}}(s + cG^{\pi_\theta}(\bar{x}, \pi_\theta(\bar{x})))\big)$$

Compared to the D4PG algorithm (Barth-Maron et al., 2018), which employs a deterministic policy gradient with a distributional critic, our approach has an additional $\phi\big(F_{G^{\pi_\theta}}(s + cG^{\pi_\theta}(\bar{x}, a))\big)$ term for risk sensitivity. This term evaluates the relationship between G^{π_θ} and $s + cG^{\pi_\theta}(\bar{x}, a)$, and adjusts the coefficient of $\nabla_a G^{\pi_\theta}(\bar{x}, a)$ according to the agent's risk preferences, defined by the risk spectrum ϕ.

To effectively apply this policy gradient in the offline setting while addressing the challenge of distributional shift, a straightforward yet robust approach is to incorporate a regularization term for behavior cloning:

$$\begin{aligned}
\nabla_\theta J(\pi_\theta, h) \approx \mathbb{E}_{\bar{x}, \hat{a} \sim \mathcal{D}}\Big[&\nabla_\theta \pi_\theta(\bar{x}) \nabla_a Q_h^{\pi_\theta}(\bar{x}, a)\Big|_{a=\pi_\theta(\bar{x})} \\
&- \lambda \nabla_\theta \pi_\theta(\bar{x})\big(\pi_\theta(\bar{x}) - \hat{a}\big)\Big]. \quad (12)
\end{aligned}$$

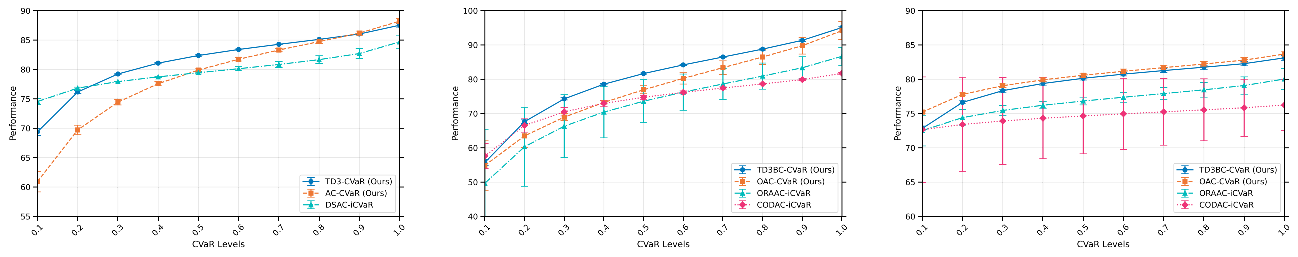
in which λ controls the strength of the regularizer. The regularization term encourages the policy to produce actions that align with those observed in the dataset.

Deterministic policies, including their risk-sensitive variants, share common issues such as sensitivity to hyperparameter selection and exploitation of errors in the Q-function. To address these challenges, modifications proven to improve deterministic policy performance can be applied in the risk-sensitive context as well. The Twin Delayed DDPG (TD3) algorithm introduces three key techniques that enhance the stability and performance of standard DDPG: Clipped Double-Q Learning, Delayed Policy Updates, and Target Policy Smoothing. We incorporate these techniques to stabilize training with our risk-sensitive deterministic policy gradients. The detailed steps for TD3-SRM and TD3BC-SRM are provided in Algorithm 4.

5. Experiment results

In this section, we benchmark our algorithm against other risk-neutral and risk-sensitive algorithms in both online and offline settings. As our baselines, we select state-of-the-art model-free off-policy algorithms. In the online setting, we benchmark our algorithm against risk-neutral algorithms such as SAC (Haarnoja et al., 2018) and TD3 (Fujimoto et al., 2018), as well as risk-sensitive algorithms like DSAC (Ma et al., 2020). Similarly, in the offline setting, we use AWAC (Nair et al., 2021), CQL (Kumar et al., 2020), IQL (Kostrikov et al., 2021), and TD3 + BC (Fujimoto & Gu, 2021) for risk-neutral comparisons, and ORAAC (Urpí et al., 2021) and CODAC (Ma et al., 2021) for risk-sensitive evaluations.

Risk-sensitive algorithms are identified by their name and the risk measure they optimize. For example, our actor-critic algorithm using CVaR is referred to as AC-CVaR, while DSAC with the iterative CVaR risk measure is called DSAC-iCVaR. To demonstrate the generality of our framework beyond standard CVaR optimization, we conduct experiments using diverse Spectral Risk Measures, including the Exponential SRM (Section 5.1) and the Mean-CVaR (Sections 5.2 and 5.3).The results in both online and offline settings are normalized so that a score of 0 corresponds to a random policy, and a score of 100 corresponds to an



(a) Online (b) Offline (Expert) (c) Offline (Random)

Fig. 4. Risk-sensitive performance comparison of online and offline RL algorithms with the $\mathrm{CVaR}_{0.2}$ objective.

expert policy. The hyperparameters of each model, as well as the values used for normalization, are available in the Appendix A.

We evaluate these models in environments related to finance, healthcare, and robotics. Descriptions of these environments are provided in each section. In offline RL, the policy used to generate the dataset plays a critical role in shaping the learned policy. Following established conventions in the offline RL literature (Fu et al., 2021), we detail the datasets employed in each of our offline experiments. The *Medium* dataset contains transitions from an early-stopped SAC policy, while the *Medium-Replay* dataset is comprised of the replay buffer collected during the training of the Medium policy. The *Medium-Expert* dataset includes a mixture of sub-optimal and expert-level data, and the *Expert-Replay* dataset consists of the replay buffer from training the SAC policy.

5.1. Mean-reverting trading.

In this environment, the agent aims to make a profit by trading an asset that follows a mean-reverting Ornstein-Uhlenbeck process, described by the equation

$$\mathrm{d}P_t = \kappa(\zeta - P_t)\mathrm{d}t + \sigma \mathrm{d}W_t,$$

where $\zeta = 1.0$ is the long-term mean, $\kappa = 2.0$ controls the speed of reversion to the mean, $\sigma = 1.0$ is the volatility of the random fluctuations, and W_t is a standard Wiener process. At each time step, $t = 0, \dots, T-1$, the agent takes an action $a_t \in (-a_{\max}, a_{\max})$, which adjusts its inventory $q_t \in (-q_{\max}, q_{\max})$, representing the quantity of the asset traded.

The state is represented by a 3-dimensional vector comprising the asset price, the quantity of assets the agent possesses, and the remaining time. The rewards are also structured as follows: for $t = 0$ to $T-2$, the agent's reward is $r_t = -a_t P_t - \varphi a_t^2$, incorporating transaction costs ($\varphi = 0.005$). At the final time step, $t = T-1$, the reward is modified to include an additional term penalizing the agent for holding any remaining inventory, given by $r_{T-1} = -a_{T-1} P_{T-1} - \varphi a_{T-1}^2 + q_T P_T - \psi q_T^2$, where $\psi = 0.5$ represents the terminal penalty.

First, we compare our online algorithms, TD3-CVaR and AC-CVaR, to DSAC-iCVaR. All three algorithms are optimized using the CVaR objective with $\alpha = 0.2$. To evaluate their performance, we simulate 10,000 trajectories and compute the CVaR_α of their returns. The x-axis in Fig. 4(a) represents CVaR levels, ranging from 0.1 to 1.0, while the y-axis shows the average normalized score ± standard deviation across five random seeds. The results indicate that while TD3-CVaR and DSAC-iCVaR achieve similar $\mathrm{CVaR}_{0.2}$ values, TD3-CVaR performs better in terms of expected return.

In the offline setting, we observe similar trends. These experiments are conducted using the Expert-Replay and Random datasets. As shown in Fig. 4(b), TD3BC-CVaR and CODAC-iCVaR achieve comparable $\mathrm{CVaR}_{0.2}$, but differences emerge in their expected returns. When trained on the Random dataset, all models experience a decline in performance due to the lack of expert demonstrations, as expected. However, as illustrated in Fig. 4(c), both of our models successfully extract useful policies from the dataset and outperform ORAAC-iCVaR and CODAC-iCVaR.

To isolate the impact of the risk measure from other design choices, we conduct an additional experiment comparing our TD3-Exp model to its iterative risk measure counterpart, TD3-iExp. We utilize the Exponential SRM, defined by the risk spectrum $\phi_\alpha(u) = \frac{\alpha e^{-\alpha u}}{1-e^{-\alpha}}$, where the parameter $\alpha > 0$ controls the degree of risk aversion. Fig. 5 highlights that iterative risk measures lead to more conservative policies. While both risk measures produce policies that align with their respective risk-sensitive objectives, policies optimized with iterative risk measures yield lower expected returns at higher risk levels.

5.2. Portfolio allocation

To evaluate our algorithms in a more realistic setting, we design a portfolio allocation environment based on real-world market data. While RL has proven to be a suitable and widely used approach for portfolio management (Almahdi & Yang, 2017; Park et al., 2020; Pendharkar & Cusatis, 2018; Wang & Ku, 2022), the problem has received little attention in the offline RL setting, despite its practical relevance in financial applications where interaction with the market is costly. Our primary goal with this environment is to demonstrate the effectiveness of learning various risk-sensitive policies in an offline setting. The environment is based on historical daily price data obtained from Yahoo Finance for a set of core assets: SPY (an ETF tracking the S&P 500 index, representing U.S. equities), GLD (an ETF tracking the price of gold, representing a key alternative asset), and a risk-free cash component with 0% return. This selection provides a balance between equity market exposure, a non-correlated safe-haven asset, and a risk-free option, capturing essential trade-offs in portfolio allocation among growth, diversification, and capital preservation. The historical data spans twenty years, from January 1, 2005, to December 31, 2024, covering a range of market regimes including bull markets, recessions, and periods of financial stress. To enable realistic backtesting and prevent look-ahead bias, the data is chronologically split: the initial 80% of the time series serves as the training dataset for policy learning, while the remaining 20% forms a held-out test set for evaluating the generalization performance of trained agents on unseen future data.

Within the environment, we use historical daily closing prices $p_{i,t}$ for each asset i at time t to compute log-returns defined as $r_{i,t} = \log(p_{i,t}/p_{i,t-1})$. The agent observes a state composed of the most recent $m = 5$ log-returns for each asset, along with the current portfolio weights. A TD3 algorithm is trained to learn an expert policy that outputs the portfolio allocation weights for the next trading period. To reflect practical trading constraints, we enforce that the policy produces non-negative weights that sum to one, effectively prohibiting short selling. The reward at each step is the log-return of the portfolio value, computed using the new weights and asset returns. We also deduct a transaction cost equal to 0.25% of the trading value at each allocation step to simulate realistic market frictions.

The replay buffer accumulated during TD3 training serves as the Expert-Replay dataset, which we subsequently use to train alter-

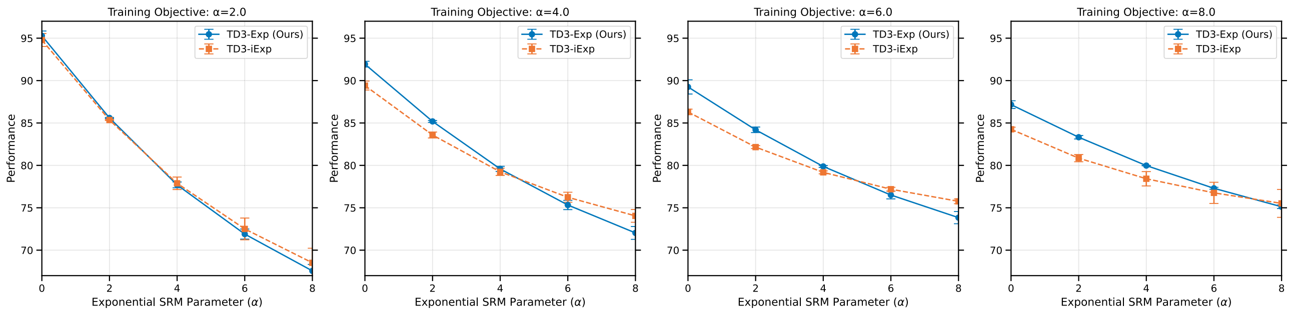


Fig. 5. Risk-sensitive performance comparison of TD3-Exp and TD3-iExp using the Exponential SRM. The α value above each subfigure indicates the risk aversion parameter used in the training objective. The x-axis represents the Exponential SRM risk aversion parameter used to evaluate the trained policies.

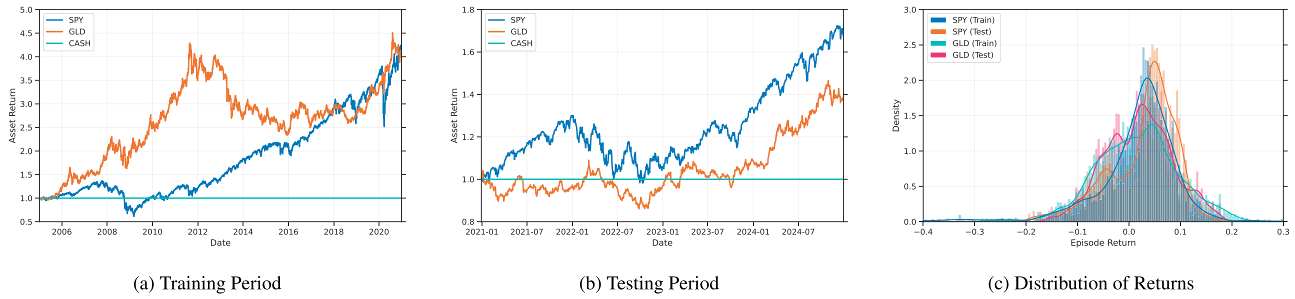


Fig. 6. Fig. 6(a) and (b) display the asset returns during the training and testing period. Fig. 6(c) illustrates the distribution of asset returns in an episode.

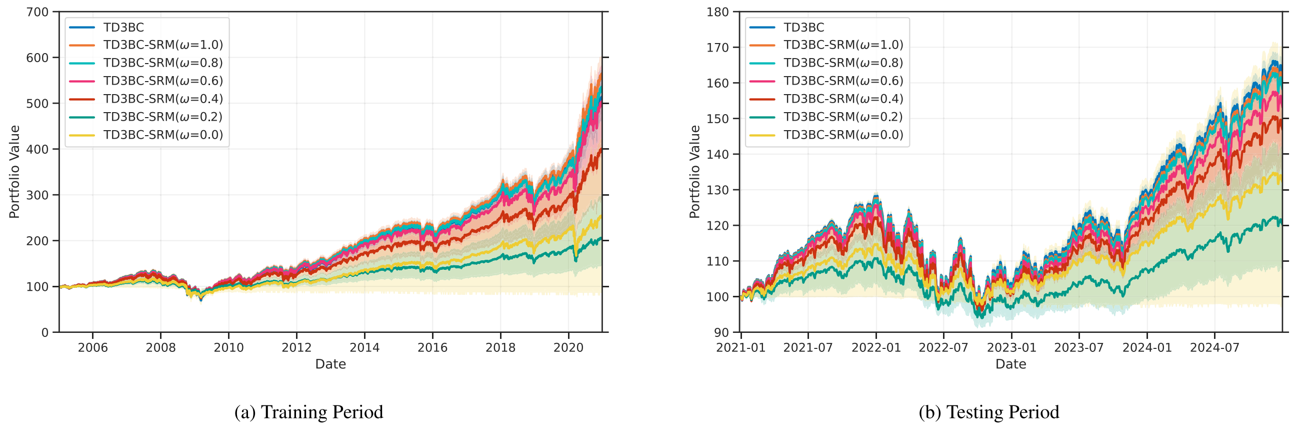


Fig. 7. The value of the portfolio in the training and testing period for different policies. The shaded regions correspond to 95% confidence interval.

native policies tailored to different risk preferences. This approach enables us to leverage a single dataset to learn a diverse set of risk-sensitive policies without requiring further interaction with the environment.

Each episode in the environment spans 63 trading days, corresponding to approximately three calendar months. The starting point of each episode is randomly selected from the historical data to provide diverse market conditions. Since rewards are defined as log-returns, the cumulative reward over an episode corresponds to the log-return of the portfolio over the entire episode, offering a coherent measure of long-term performance. Fig. 6 provides an overview of the asset data used in the environment. Fig. 6(a) and (b) show the asset return trajectories for SPY, GLD, and the risk-free asset (CASH) during the training and testing periods, respectively. Fig. 6(c) illustrates the distribution of episode returns across assets, highlighting the variability and skewness in return profiles over different market regimes.

Using the Expert-Replay dataset, we train TD3BC and TD3BC-MC (TD3BC optimized with the Mean-CVaR objective) with $\alpha = 0.2$ and $\omega \in \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$, where ω controls the weight of the expected return in the Mean-CVaR objective, given by:

$$\mathrm{MC}_{\alpha,\omega}(G^\pi) = \omega\mathbb{E}[G^\pi] + (1-\omega)\,\mathrm{CVaR}_\alpha(G^\pi). \tag{13}$$

The practical impact of varying risk preferences on the portfolio value over time is visualized in Fig. 7. This figure tracks the cumulative value of the portfolio comprising SPY, GLD, and CASH assets across the training and testing periods. As risk aversion increases (represented by decreasing values of ω in the Mean-CVaR objective), the trajectories become notably smoother, exhibiting lower volatility and smaller drawdowns compared to the risk-neutral TD3BC baseline.

This setup allows us to capture a diverse set of policies aligned with different levels of risk sensitivity. We evaluate the performance of these algorithms during both the training and testing periods using four different metrics. To facilitate fair comparison across algorithms, we first normalize each algorithm's performance relative to the expert TD3 policy on the training (2.81% average return) and testing (2.48% average return) period, following the normalization procedure discussed earlier. We then compute and report the Mean and Conditional Value-at-

Table 1
Results for the offline algorithms evaluated in the training period of the portfolio allocation environment. The ± symbol represents the standard deviation across seeds.

Metric	TD3BC	TD3BC-MC ($\omega = 1.0$)	TD3BC-MC ($\omega = 0.8$)	TD3BC-MC ($\omega = 0.6$)	TD3BC-MC ($\omega = 0.4$)	TD3BC-MC ($\omega = 0.2$)	TD3BC-MC ($\omega = 0.0$)
$\mathbb{E}$	92.56±0.88	**93.88±1.39**	92.49±2.43	91.55±4.68	88.71±4.61	80.26±5.35	80.79±11.33
$\text{CVaR}_{0.2}$	1.20±5.79	6.93±3.13	8.30±3.17	11.51±8.49	14.38±15.14	**28.61±24.38**	22.66±48.01
SR	0.65±0.04	**0.69±0.03**	0.68±0.03	0.66±0.08	0.60±0.07	0.38±0.12	0.20±0.41
MDD	−48.77±2.83	−45.45±1.66	−44.88±2.15	−43.19±5.78	−41.72±9.70	**−31.12±12.37**	−36.21±25.66

Table 2
Results for the offline algorithms evaluated in the testing period of the portfolio allocation environment. The ± symbol represents the standard deviation across seeds.

Metric	TD3BC	TD3BC-MC ($\omega = 1.0$)	TD3BC-MC ($\omega = 0.8$)	TD3BC-MC ($\omega = 0.6$)	TD3BC-MC ($\omega = 0.4$)	TD3BC-MC ($\omega = 0.2$)	TD3BC-MC ($\omega = 0.0$)
$\mathbb{E}$	**99.97±1.85**	99.17±0.68	97.71±2.32	96.26±6.10	94.25±5.66	84.54±8.75	87.43±16.53
$\text{CVaR}_{0.2}$	0.80±2.19	0.91±1.71	3.56±5.37	5.81±8.89	11.69±15.52	30.65±15.93	**33.23±40.60**
SR	0.82±0.01	0.81±0.02	0.81±0.01	0.76±0.13	0.76±0.07	0.46±0.26	**0.86±0.01**
MDD	−24.28±0.54	−24.23±0.46	−23.89±0.47	−23.36±1.64	−21.23±4.62	**−15.29±4.71**	−24.47±0.21

Risk (CVaR) at a confidence level $\alpha = 0.2$ for this normalized performance, capturing both the average and the worst-case behavior of each algorithm.

In addition, we report the Sharpe Ratio (SR), defined as:

$$\text{SR} = \frac{\mathbb{E}[R]}{\sigma(R)},$$

where R represents the daily log-returns over the testing period, $\mathbb{E}[R]$ is the mean return, and $\sigma(R)$ is its standard deviation. A higher Sharpe Ratio indicates better risk-adjusted performance. We also include the Maximum Drawdown (MDD), which measures the largest loss from a peak to a subsequent trough in portfolio value over time. Formally, for a time series of portfolio values $\{V_t\}_{t=1}^{T}$, MDD is defined as:

$$\text{MDD} = \max_{t \in [1,T]} \left(\max_{s \in [1,t]} \frac{V_s - V_t}{V_s} \right).$$

Together, these four metrics provide a comprehensive evaluation of each algorithm's return profile and risk characteristics. Tables 1 and 2 summarize the results for all algorithms in the training and testing periods, respectively.

Unlike TD3BC, which lacks the flexibility to adapt to different risk preferences, our algorithm is explicitly designed to address worst-case outcomes and produce policies tailored to varying levels of risk aversion. As seen in Table 1, during the training period, we observe that the worst-case performance improves as more weight is allocated to CVaR in the Mean-CVaR objective, an expected result given that training and evaluation occur on the same data. Notably, the Mean-CVaR objective with $\omega = 0.2$ achieves the best $\text{CVaR}_{0.2}$ and MDD performance. This outcome can be attributed to more stable training under the Mean-CVaR objective compared to directly optimizing CVaR, as reflected by lower standard deviations across different random seeds.

Turning to the results of the policies on the unseen testing period (Table 2), we observe similar improvements in the worst-case performance of TD3BC-MC, especially compared to the risk-neutral TD3BC. In this period, TD3BC-MC with $\omega = 0.0$ achieves the highest $\text{CVaR}_{0.2}$ and Sharpe Ratio, demonstrating the effectiveness of our risk-sensitive algorithm. An interesting observation is that, in Table 1 for the training period, we see a significant increase in $\text{CVaR}_{0.2}$ and a decline in average performance as ω decreases from 0.4 to 0.2. Table 2 shows that this pattern is also present in the testing period. This finding has important practical implications as different risk-sensitive policies trained offline on the dataset exhibit consistent behavior when evaluated on unseen data. Consequently, users can train a diverse set of risk-sensitive policies without additional environment interaction and select among them based on their desired risk-return profile.

Finally, we compare our results against a naive Equal Weighted (EW) strategy, a common baseline in finance. In the test period, the EW strategy yielded an expected return of 89.04, a $\text{CVaR}_{0.2}$ of 40.84, a Sharpe Ratio of 1.08, and a Maximum Drawdown (MDD) of -3.6. These results indicate that the EW portfolio exhibits strong risk-adjusted performance and low volatility in this market setting. Importantly, this comparison highlights a fundamental challenge in offline reinforcement learning: policy performance is limited by the support of the training dataset. The expert-replay dataset used here was generated by an agent focused on maximizing returns, resulting in aggressive trading behavior. Consequently, the highly diversified, low-volatility nature of the EW strategy is effectively out-of-distribution relative to the behavior policy. Diversification patterns similar to an EW portfolio are not represented in the replay buffer and cannot, in general, be recovered without violating the support constraints of the data. Nevertheless, our method demonstrates its value by significantly improving risk metrics within the constraints of the available data; specifically, it reduces the MDD of the behavior policy from -24.28 to -15.29, a 37% improvement. This confirms the algorithm's ability to successfully extract the most risk-averse behavior possible from the provided historical data.

5.3. HIV treatment.

While our previous experiments focused on financial applications, our method is not limited to this domain. To demonstrate its broader applicability, we evaluate it in the context of healthcare using the HIV treatment simulator. The HIV treatment simulator, originally introduced by Ernst et al. (2006) and implemented following the work of Geramifard et al. (2015), provides a simplified yet widely adopted model of HIV patient treatment in reinforcement learning research (Keramati et al., 2020; Rigter et al., 2023). The environment features a six-dimensional state space representing concentrations of various cells and viral loads in the patient's bloodstream. The agent controls the dosages of two drugs, administered either individually or in combination, and receives rewards based on the patient's health outcomes. Due to stepwise variability in drug efficacy, the environment is inherently stochastic. Each treatment is applied for 20 consecutive days, with a total of 50 time steps per episode, resulting in an episode length of 1,000 days.

Compared to the trading environment, the HIV treatment environment exhibits substantially greater stochasticity and carries higher real-world stakes, where minimizing adverse outcomes is essential. Accordingly, we prioritize risk aversion in this setting by employing the Mean-CVaR objective with $\alpha = 0.1$ and $\omega = 0.4$. As shown in Table 3, while the iterative Mean-CVaR objective leads to modest improvements in the risk-sensitive performance of DSAC, our algorithm optimized for the static Mean-CVaR objective achieves substantial gains in both expected return and $\text{CVaR}_{0.1}$ when applied to TD3 and AC. Notably, TD3-MC achieves the highest performance with respect to both metrics.

Table 3
Normalized results for the online experiments in the HIV Treatment environment. We report both the expected return and CVaR$_\alpha$, averaged across 5 different seeds. The ± symbol represents the standard deviation across seeds.

Metric	AC-MC (Ours)	TD3-MC (Ours)	DSAC-iMC	AC	TD3	DSAC
$\mathbb{E}$	54.99±12.22	**100.33±19.43**	87.92±13.89	48.13±13.18	71.65±5.61	81.15±16.67
CVaR$_{0.1}$	16.24±10.63	**53.80±10.19**	30.79±8.99	10.76±8.08	44.59±5.37	27.65±10.57

Table 4
Normalized results for the offline experiments in the HIV Treatment environment. We report the expected return and CVaR$_\alpha$, averaged across 5 seeds. The ± symbol represents the standard deviation across seeds.

Metric	OAC-MC (Ours)	TD3BC-MC (Ours)	ORAAC-iMC	CODAC-iMC	CQL	IQL	AWAC	TD3BC
$\mathbb{E}$	52.81±0.61	**76.01±0.83**	35.01±12.35	49.05±3.97	68.61±1.56	58.40±0.44	46.65±2.23	71.82±1.21
CVaR$_{0.1}$	10.28±1.65	**43.73±1.66**	1.89±0.58	10.94±0.96	35.11±2.44	19.59±1.49	3.47±0.63	37.33±2.43

Table 5
Normalized results for the online experiments. We report the expected return and CVaR$_\alpha$, averaged across 5 seeds. The ± symbol represents the standard deviation across seeds.

Environment	Metric	AC-MC (Ours)	TD3-MC (Ours)	DSAC-iMC	AC	TD3	DSAC
HalfCheetah	$\mathbb{E}$	91.59±0.96	**97.68±1.41**	78.57±9.73	66.85±16.13	82.15±3.75	80.86±1.52
	CVaR$_{0.2}$	89.14±1.25	**93.30±2.69**	69.91±9.97	40.37±24.17	76.35±4.87	67.58±1.54
Hopper	$\mathbb{E}$	98.73±17.38	101.20±4.30	54.43±12.48	**105.21±2.75**	100.14±3.70	62.24±18.44
	CVaR$_{0.2}$	96.77±18.59	99.11±6.21	46.58±13.91	**104.62±2.88**	95.30±5.75	52.85±24.36
Walker2D	$\mathbb{E}$	90.36±5.33	**97.20±13.48**	80.17±25.85	97.15±13.47	82.31±5.79	75.55±28.26
	CVaR$_{0.2}$	84.76±9.74	**93.57±10.66**	27.31±28.69	85.14±16.82	66.99±11.67	31.02±28.35

Table 6
Normalized results for the offline experiments with the Medium dataset. We report the expected return and CVaR$_\alpha$, averaged across 5 seeds. The ± symbol represents the standard deviation across seeds.

Environment	Metric	OAC-MC (Ours)	TD3BC-MC (Ours)	ORAAC-iMC	CODAC-iMC	CQL	IQL	AWAC	TD3BC
HalfCheetah	$\mathbb{E}$	**58.82±18.77**	52.75±2.65	52.07±1.11	27.19±12.71	58.47±0.24	58.24±0.33	56.47±15.57	52.77±3.90
	CVaR$_{0.2}$	**47.63±15.96**	38.55±13.09	19.15±0.95	14.24±7.04	22.03±0.41	21.35±1.05	36.54±22.84	43.16±2.64
Hopper	$\mathbb{E}$	54.45±6.89	54.17±7.15	36.50±2.76	37.66±4.77	41.08±31.96	**78.01±5.29**	59.61±11.31	68.33±16.16
	CVaR$_{0.2}$	42.50±3.48	43.43±3.93	27.38±2.01	26.28±5.09	29.48±30.04	**67.34±3.61**	46.42±13.27	56.62±14.59
Walker2D	$\mathbb{E}$	76.30±3.86	**88.73±2.89**	25.99±10.05	5.65±2.06	77.79±4.95	51.33±5.34	68.74±9.08	86.28±7.67
	CVaR$_{0.2}$	43.08±21.66	**81.89±4.45**	6.47±0.55	3.25±2.71	51.98±11.82	3.61±2.53	28.00±11.98	69.81±20.47

In the offline setting, we use the Medium-Replay dataset, which consists of the replay buffer collected during SAC training, up to the point where the policy reaches 40% of the expert policy's performance (Rigter et al., 2023). Using this dataset, our static Mean-CVaR algorithms, OAC-MC and TD3BC-MC, again demonstrate improvements in both expected return and CVaR$_{0.1}$ over risk-neutral baselines such as AWAC and TD3BC, as shown in Table 4. Among all models evaluated, TD3BC-MC attains the best performance. Conversely, CODAC-iMC which is a distributional variant of CQL with an iterative risk measure underperforms, suggesting that iterative risk measures may be less effective in this context. One possible explanation is that the combined conservativeness of CQL and iterative risk measures prevent the policy from reaching higher-reward states and ultimately reduce both expected return and worst-case performance.

5.4. Stochastic MuJoCo.

To further showcase the generality of our method, we evaluate it in a robotics setting using MuJoCo continuous control tasks. MuJoCo (Todorov et al., 2012) is a physics engine for simulating complex physical interactions, widely used in robotics research. These robotics simulators are significantly more complex than previous environments due to their larger state and action spaces. In these environments, transitions are deterministic, with stochasticity only arising from the initial state. To introduce additional stochasticity, we use a stochastic version of MuJoCo, where small perturbations are added to the rewards (Ma et al., 2021; Urpí et al., 2021).

For HalfCheetah, we modify the reward function as follows:

$$R_t(s,a) = \bar{r}_t(s,a) - 70\,\mathbb{1}_{v>\bar{v}} \cdot \mathcal{B}_{0.1},$$

where $\bar{r}_t(s,a)$ is the original environment reward, v denotes the forward velocity, and $\mathcal{B}_{0.1}$ is a Bernoulli random variable with success probability $p = 0.1$. We set the velocity threshold $\bar{v}$ to 4 for the medium dataset. The maximum episode length for HalfCheetah is set to 200 steps.

For Walker2D and Hopper, the reward function is modified to:

$$R_t(s,a) = \bar{r}_t(s,a) - p\,\mathbb{1}_{|\theta|>\bar{\theta}} \cdot \mathcal{B}_{0.1},$$

where $\bar{r}_t(s,a)$ is the original reward, θ represents the pitch angle, and $\mathcal{B}_{0.1}$ introduces stochastic penalties. The penalty threshold $\bar{\theta}$ is set to 0.5 for Walker2D and 0.1 for Hopper, with corresponding penalty magnitudes $p = 30$ and $p = 50$, respectively. Additionally, if $|\theta|$ exceeds $2\bar{\theta}$, the agent is considered to have fallen, terminating the episode. This stochastic penalty mechanism penalizes unstable postures, encouraging the agent to maintain balance. Both Walker2D and Hopper are evaluated over a maximum of 500 time steps.

These environments are the most complex in our experiments, with added stochasticity making them even more challenging. We use the Mean-CVaR objective with $\alpha = 0.2$ and $\omega = 0.2$ for risk-sensitive algorithms in these settings. Results in Table 5 indicates that in the HalfCheetah and Walker2D environments, both AC-MC and TD3-MC improve performance over their risk-neutral counterparts, AC and TD3, demonstrating that incorporating risk sensitivity can help discover more effective policies. Notably, TD3-MC achieves the best results across all tested algorithms. Moreover, AC-MC and TD3-MC surpass DSAC-iMC in these environments, further highlighting the advantages of our approach over iterative risk measures. In the Hopper environment, AC, TD3, AC-MC, and TD3-MC perform similarly, with little difference in their results. A notable finding is that DSAC-iMC performs worse than DSAC in the Hopper and Walker2D environments. This decline may be due to the use of an iterative risk measure in environments with long horizons which results in overly conservative policies.

In the offline setting of the Stochastic MuJoCo environment, our results in Table 6 suggest that some performance gains achieved with the Mean-CVaR objective may diminish in more complex scenarios. While our models generally outperform other risk-sensitive offline algorithms, such as ORAAC and CODAC, part of this improvement can be attributed to the strength of the underlying offline algorithms, like TD3BC and AWAC. For example, in Walker2D, the Mean-CVaR objective improves performance compared to TD3BC and AWAC. However, in other environments, such as Hopper, it actually decreases performance. The Mean-CVaR objective assigns different weights to various quantiles of the return distribution. While lower quantiles are crucial in risk-sensitive applications, their estimation is inherently less accurate due to the smaller sample size used. As a result, the noisy estimation of these quantiles may lead to decreased performance in some cases.

5.5. Discussion

Our empirical results highlight the advantages of using a static risk measure over both risk-neutral baselines and risk-sensitive methods based on iterative risk formulations. We also demonstrate that our offline framework can be used to learn diverse risk-sensitive policies aligned with different risk preferences. One notable observation from our experiments is that while stochastic policies are well-suited for inherently stochastic environments, the added randomness can hinder the effectiveness of risk-sensitive policies. This trend appears consistently across both online and offline settings, where TD3-SRM and TD3BC-SRM achieve strong risk-sensitive performance in their respective domains.

Another important insight is that naively incorporating risk sensitivity into distributional actor-critic methods with entropy-based exploration can lead to degraded performance. This issue arises because soft critics estimate both future rewards and policy entropy. Since risk adjustments are meaningful only for returns and not for entropy, the risk measure should be applied solely to the reward component. Decoupling return estimation from entropy may improve the performance of risk-sensitive methods while retaining the benefits of entropy-based exploration, presenting a compelling direction for future work.

6. Conclusion

Despite rapid progress in risk-sensitive reinforcement learning, two important gaps remain. First, the dominant use of iterative risk measures in DRL, while easy to implement, often leads to suboptimal policies when the goal is to optimize static risk preferences. Second, despite growing interest in offline RL, there is little support for directly optimizing static risk measures, even though they are essential in safety-critical settings.

In this paper, we address these challenges by introducing a unified actor-critic framework for optimizing static SRMs in both online and offline RL. Our approach leverages the supremum representation of SRMs to develop a simple and efficient optimization method with minimal computational overhead. We provide convergence guarantees in the online tabular setting and show how policy constraints can be integrated to mitigate distributional shift in the offline setting, enabling practical use on fixed datasets.

Our framework enables learning a wide range of risk-sensitive policies from a single offline dataset, with each policy tailored to a different level of risk preference. This is especially valuable in domains like finance, healthcare, and robotics, where environment interaction is expensive or risky and the ability to manage worst-case outcomes is critical. Through extensive experiments, we show that our methods consistently outperform existing risk-sensitive algorithms, especially when it comes to controlling risk while maintaining strong returns.

Overall, our contributions bring together solid theoretical grounding and practical tools, pushing risk-sensitive RL closer to deployment in real-world applications.

Credit author statement

Mehrdad Moghimi: Data Curation, Investigation, Methodology, Writing — original draft **Hyejin Ku:** Conceptualization, Methodology, Supervision, Writing — review & editing

Data availability

The data and code used in this study are available via the links provided in Appendix A.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Details of the experiments

A.1. Baselines

Online Setting. The implementation of our model, as well as the baselines, follow the single-file implementation of RL algorithms from CleanRL (Huang et al., 2022) for clarity. We use the JAX (Bradbury et al., 2018) implementation of the SAC and TD3 algorithms from CleanRL[2]'s repository. We implement the AC algorithm with JAX according to Algorithm 2.

Offline setting. For our risk-sensitive baselines, we use the original implementation of the O-RAAC[3] and CODAC[4] algorithms. For TD3BC, AWAC, CQL, and IQL, we use the JAX implementations in the JAX-CORL[5] repository. For each algorithm, we use the default hyperparameters introduced in their corresponding paper.

The Expert-Replay datasets used in the trading and portfolio allocation environments are generated by training expert policies using SAC and TD3, respectively. For the HIV treatment environment, we use the Medium-Replay dataset provided by Rigter et al. (2023)[6]. The Medium datasets for the Stochastic MuJoCo environments are obtained from the implementation of Urpí et al. (2021, O-RAAC). All datasets used in our experiments are publicly available through the project's GitHub repository.

A.2. Implementation details

As described in Section 4, the risk-sensitive variants of the AC and TD3 algorithms in the online setting, as well as the AWAC and TD3BC algorithms in the offline settings, are developed by adding three components: 1) a distributional Critic, 2) an extended state space, and 3) a periodically updated function h based on the return distribution of the initial state. We use the QR-DQN algorithm for our distributional critic and we

[2] https://github.com/vwxyzjn/cleanrl
[3] https://github.com/nuria95/O-RAAC
[4] https://github.com/JasonMa2016/CODAC
[5] https://github.com/nissymori/JAX-CORL
[6] https://huggingface.co/datasets/marcrigter/1R2R-datasets

Table A.1
Default hyperparameters in different models.

Hyperparameter	Value
Learning Rate	3e-4
Discount Factor (γ)	0.99
Batch Size	256
Number of Quantiles	50
Target Smoothing Coefficient	5e-3
Hidden Layer dimension	256
Number of layers	2
Activation Function	ReLu

Table A.2
Expected return of a random policy and an expert policy after 1 million time-steps.

Environment	Random Policy	Expert Policy
Trading	−6.17	1.72
Portfolio (Train)	−0.0776	0.0248
Portfolio (Test)	−0.0815	0.0281
HIV Treatment	−48.7	5557.9
HalfCheetah	−57.95	659.49
Hopper	−13.37	1602.04
Walker2D	−17.32	1790.02

set the number of quantiles to 50. We also use the return-distribution of the initial state to update the function h every 500 time-steps.

Table A.1 lists the shared hyperparameters for both online and offline settings. In the online experiments, the Trading environment is trained for 500,000 timesteps and the HIV Treatment environment and the Stochastic MuJoCo environment for 1,000,000 timesteps. For the offline experiments, all environments are trained for 500,000 timesteps. Additionally, the Lagrange multiplier for AWAC and OAC-SRM is fixed at 1.0, while the behavior cloning coefficient in TD3BC and TD3BC-SRM is set to 2.5.

The code for the project is available at https://github.com/MehrdadMoghimi/ACSRM.

A.3. Evaluation

For both the online and offline settings, we train each algorithm using 5 different random seeds. Once training is complete, we generate 1,000 trajectories using the learned policies. The returns are normalized following the method outlined by Fu et al. (2021), utilizing the values from Table A.2.

A.4. Computational requirement

In the online setting, each experiment with 1 million time-steps takes approximately $1-2$ h on a Windows 11 PC with 16GB of RAM, Intel Core i5-12600K CPU, and Nvidia RTX 3060 GPU. In the offline setting, 1 million time-steps of training take $10-20$ min on the same system.

Appendix B. Sensitivity analysis for the behavior cloning parameter λ

Fig. B.1 presents the sensitivity analysis for the Lagrange multiplier λ in the HIV Treatment environment, evaluating both OAC-MC and the baseline AWAC across values ranging from 0.1 to 100.0 on a logarithmic scale. The results demonstrate that our proposed OAC-MC consistently outperforms the risk-neutral AWAC in terms of both Expected Return and worst-case performance (CVaR$_{0.1}$) across a broad range of hyperparameters ($\lambda \in [0.1, 5.0]$). Notably, the peak performance of OAC-MC surpasses the maximum achieved by AWAC in both metrics. We observe performance degradation for both algorithms at lower λ values due to extrapolation errors, as well as at very high λ values where the policy becomes over-constrained to the behavior policy. However, OAC-MC consistently surpasses the baseline throughout the effective operating range, confirming that its superior risk-aversion is driven by the spectral risk objective rather than the tuning of the offline constraint.

Appendix C. Proof of Proposition 4.1

As seen in Section 3.1, we have $\tau_i = \frac{i}{N}$ and $\hat{\tau}_i = \frac{\tau_{i-1}+\tau_i}{2}$ for $i = 1, \dots, N$. The quantile representation of the random variable Z with $q_i = F_Z^{-1}(\hat{\tau}_i)$ allows us to express Eq. (4) as a piecewise linear function:

$$\tilde{h}_{\phi,Z}(z) \approx \sum_{i=1}^{N} w_i\left(q_i + \frac{1}{\hat{\tau}_i}(z-q_i)^-\right), \tag{C.1}$$

where $w_i = \hat{\tau}_i\big(\phi(\tau_{i-1}) - \phi(\tau_i)\big)$. To show this, using the definition of function h, we can write:

$$\begin{aligned} h(z) &= \int_0^1 \left(F_Z^{-1}(\alpha) + \frac{1}{\alpha}\left(z - F_Z^{-1}(\alpha)\right)^-\right)\mu(\mathrm{d}\alpha) \\ &= \sum_{i=1}^{N}\left(\int_{\tau_{i-1}}^{\tau_i}\left(F_Z^{-1}(\alpha) + \frac{1}{\alpha}\left(z - F_Z^{-1}(\alpha)\right)^-\right)\mu(\mathrm{d}\alpha)\right) \\ &\overset{(a)}{=} \sum_{i=1}^{N}\left(q_i\int_{\tau_{i-1}}^{\tau_i}\mu(\mathrm{d}\alpha) + \left(z-q_i\right)^-\int_{\tau_{i-1}}^{\tau_i}\frac{1}{\alpha}\mu(\mathrm{d}\alpha)\right) \\ &\overset{(b)}{\approx} \sum_{i=1}^{N}\left(q_i\hat{\tau}_i\big(\phi(\tau_{i-1})-\phi(\tau_i)\big) + \left(z-q_i\right)^-\big(\phi(\tau_{i-1})-\phi(\tau_i)\big)\right) \\ &\approx \sum_{i=1}^{N} w_i\left(q_i + \frac{1}{\hat{\tau}_i}(z-q_i)^-\right). \end{aligned} \tag{C.2}$$

In this calculation, the integration interval $[0, 1]$ is partitioned into N subintervals: $[\tau_0, \tau_1), [\tau_1, \tau_2), \dots, [\tau_{N-1}, \tau_N]$. Each integral $\int_{\tau_{i-1}}^{\tau_i}\mu(\mathrm{d}\alpha)$ is evaluated over $[\tau_{i-1}, \tau_i)$, meaning it includes the lower limit τ_{i-1} but excludes the upper limit τ_i. In step (a), we used the fact that q_i is constant within each interval $[\tau_{i-1}, \tau_i)$. In step (b), we apply the relationship between ϕ and μ as given in Eq. (5). More specifically, the approximation for the first term relies on the idea that if the interval is sufficiently small, it is reasonable to approximate α in the integrand by its midpoint value $\hat{\tau}_i$. This yields:

$$\int_{\tau_{i-1}}^{\tau_i}\frac{\mu(\mathrm{d}\alpha)}{\alpha} \approx \frac{1}{\hat{\tau}_i}\int_{\tau_{i-1}}^{\tau_i}\mu(\mathrm{d}\alpha) \implies \int_{\tau_{i-1}}^{\tau_i}\mu(\mathrm{d}\alpha) \approx \hat{\tau}_i\big[\phi(\tau_{i-1}) - \phi(\tau_i)\big].$$

Also, the discretization error can be controlled and vanishes as the partition is refined. We analyze the approximation error of $\tilde{h}_{\phi,Z}(z)$ by viewing it as a Riemann-sum approximation of the integral in Eq. (4). Let the integrand be denoted by $\psi(\alpha) = F_Z^{-1}(\alpha) + \frac{1}{\alpha}(z - F_Z^{-1}(\alpha))^-$. The error is defined as the absolute difference between the true integral $\int_0^1 \psi(\alpha)\mu(d\alpha)$ and its discrete approximation. We observe that since the returns are bounded within $[G_{\min}, G_{\max}]$, the function $\psi(\alpha)$ is bounded and of bounded variation on $(0, 1]$. Moreover, the spectral measure μ is assumed to be a probability measure on $[0, 1]$ that is absolutely continuous with respect to the Lebesgue measure. Consequently, under the refinement of the partition $\{\tau_i\}_{i=0}^N$, the Riemann-sum approximation converges to the true integral, and the resulting discretization error vanishes as $N \to \infty$.

Finally, using the quantile value of the any state-action pair $\tilde{q}_j = F_{G^\pi(\bar{x},a)}^{-1}(\hat{\tau}_j)$ and setting $q_i = F_{G^\pi}^{-1}(\hat{\tau}_i)$, we can estimate the Q-values with

$$Q^\pi(\bar{x}, a) = C + \frac{1}{N}\sum_{j=1}^{N}\left(\sum_{i=1}^{N}\big(\phi(\tau_{i-1}) - \phi(\tau_i)\big)\left(s + c\tilde{q}_j - q_i\right)^-\right), \tag{C.3}$$

where $C = \int_0^1 F_G^{-1}(\alpha)\mu(\mathrm{d}\alpha)$ is a constant for all state actions.

Appendix D. Proof of Theorem 4.2

An important result in traditional RL is the Performance Difference Lemma (Kakade & Langford, 2002), which states that the difference in

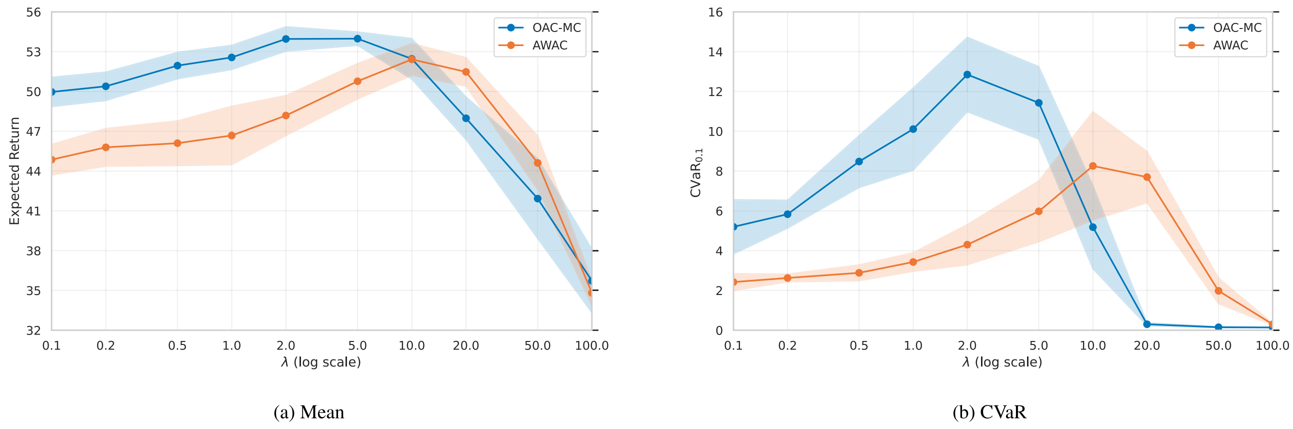


Fig. B.1. Sensitivity analysis of the Lagrange multiplier λ for OAC-MC and AWAC in the HIV Treatment environment. The shaded regions represent the standard deviation across 5 seeds.

value between two policies can be represented using the advantage function of one policy and the state occupancy measure of another. Kim et al. (2024) extend this result to show that this relationship also holds for any non-decreasing concave function. For completeness, we provide the proof of this lemma here using our notation.

Lemma D.1 (Risk-Sensitive Performance Difference Lemma). *The performance difference of policies π and π' for a non-decreasing concave function h is given by*

$$J(\pi', h) - J(\pi, h) = \mathbb{E}_{d_{\xi_0}^{\pi'},\pi'}\left[A_h^\pi(\bar{x}, a)\right]/(1-\gamma). \tag{D.1}$$

Proof. We can prove this lemma by writing:

$$\begin{aligned}
&\mathbb{E}_{d_{\xi_0}^{\pi'},\pi'}\left[A_h^\pi(\bar{x}, a)\right]/(1-\gamma) = \mathbb{E}_{\pi'}\left[\sum_{t=0}^{\infty}\gamma^t A_h^\pi(\bar{x}_t, a_t)\right]\\
&= \mathbb{E}_{\pi'}\left[\sum_{t=0}^{\infty}\left(c_t Q_h^\pi(\bar{x}_t, a_t) - c_t V_h^\pi(\bar{x}_t)\right)\right]\\
&\overset{(a)}{=} \mathbb{E}_{\pi'}\left[\sum_{t=0}^{\infty}\left(c_t V_h^\pi(\bar{x}_{t+1}) - c_t V_h^\pi(\bar{x}_t)\right)\right]\\
&= \mathbb{E}_{\pi'}\left[c_1 V_h^\pi(\bar{x}_1) - c_0 V_h^\pi(\bar{x}_0) + c_2 V_h^\pi(\bar{x}_2) - c_1 V_h^\pi(\bar{x}_1) + \cdots +\right]\\
&= \lim_{t\to\infty}\mathbb{E}_{\pi'}\left[c_t V_h^\pi(\bar{x}_t)\right] - \mathbb{E}_{\pi'}\left[c_0 V_h^\pi(\bar{x}_0)\right]\\
&= \lim_{t\to\infty}\mathbb{E}_{\pi'}\left[c_t V_h^\pi(\bar{x}_t)\right] - \mathbb{E}[h(G^\pi)]\\
&= \lim_{t\to\infty}\mathbb{E}_{\pi'}\left[\mathbb{E}\left[h\left(s_t + c_t G^\pi(\bar{x}_t)\right)\right]\right] - \mathbb{E}[h(G^\pi)]\\
&= \lim_{t\to\infty}\mathbb{E}_{\pi'}\left[\mathbb{E}\left[h\left(\sum_{t'=0}^{t}\gamma^{t'} R_{t'}^{\pi'} + \gamma^t G^\pi(\bar{x}_t)\right)\right]\right] - \mathbb{E}[h(G^\pi)]\\
&= \mathbb{E}[h(G^{\pi'})] - \mathbb{E}[h(G^\pi)]
\end{aligned}$$

where (a) uses the relationship between Q and V, i.e. $Q_h^\pi(\bar{x}, a) = \mathbb{E}_{\bar{x}'\sim P(\bar{x},a)}\left[V_h^\pi(\bar{x}')\right]$ □

With the risk-sensitive performance difference lemma established, the proof of Theorem 4.2 follows the overall structure of the convergence proof for traditional policy gradient methods in the finite state and action case (Agarwal et al., 2021). However, our analysis requires a key adaptation: we incorporate a risk-adjusted return objective rather than the standard expected return. This modification introduces additional complexity in bounding the updates, which we address by leveraging the structure of the spectral risk measure and its impact on the advantage function. We assume the rewards are positive and bounded, i.e., $R \in [R_{\min}, R_{\max}]$ with $R_{\min} \geq 0$, which implies that the state and state-action values are also bounded:

$$V_h^\pi(\bar{x}), Q_h^\pi(\bar{x}, a) \in [h(s + cG_{\min})/c, h(s + cG_{\max})/c]$$

where $G_{\min} := R_{\min}/(1-\gamma)$ and $G_{\max} := R_{\max}/(1-\gamma)$. Since h is a $\phi(0)$-Lipschitz function, the upper bound of the advantage function can be calculated as:

$$\begin{aligned}
\max|A_h^\pi(\bar{x}, a)| &= \max|Q_h^\pi(\bar{x}, a) - V_h^\pi(\bar{x})|\\
&\leq \left(h(s + cG_{\max}) - h(s + cG_{\min})\right)/c\\
&\leq \phi(0)(G_{\max} - G_{\min}).
\end{aligned}$$

Furthermore, we denote the policy as π instead of π_h for brevity, even though it depends on the function h. The softmax policy parameterization is given by:

$$\pi_\theta(a \mid \bar{x}) := \frac{\exp(\theta(\bar{x}, a))}{\sum_{a'}\exp(\theta(\bar{x}, a'))}, \forall(\bar{x}, a) \in \bar{X}, A$$

Let π_t denote π_{θ_t}. The policy can be written as:

$$\pi_{t+1}(a \mid \bar{x}) = \pi_t(a \mid \bar{x})\frac{\exp(\theta_{t+1}(\bar{x}, a) - \theta_t(\bar{x}, a))}{Z_t(\bar{x})}$$

where $Z_t(\bar{x}) := \sum_{a\in A}\pi_t(a \mid \bar{x})\exp(\theta_{t+1}(\bar{x}, a) - \theta_t(\bar{x}, a))$. Our goal is to show that, starting from π_0, we can use the policy gradient defined in Eq. (9) to reach the optimal policy in the inner optimization. By using the gradient updates as in the Natural Policy Gradient algorithm, i.e.:

$$F(\theta) = \mathbb{E}_{\bar{x}\sim d_{\xi_0}^{\pi_\theta}, a\sim\pi_\theta(\cdot|\bar{x})}\left[\nabla_\theta \log\pi_\theta(a \mid \bar{x})\left(\nabla_\theta \log\pi_\theta(a \mid \bar{x})\right)^\top\right],$$

$$\theta_{t+1} = \theta_t + \eta_t F\left(\theta_t\right)^\dagger \nabla_\theta J(\pi_t, h),$$

where F denotes the Fisher information matrix (induced by π_θ) and $F^\dagger$ denotes the Moore-Penrose pseudoinverse of F, the updates of the policy parameters take a simple form:

$$\theta_{t+1} = \theta_t + \frac{\eta_t}{1-\gamma}\left(A_h^t\right), \tag{D.2}$$

as the pseudoinverse of the Fisher information cancels out the effect of the state distribution in Eq. (9) (Agarwal et al., 2021, Lemma 15). To continue, we need to prove some intermediate results.

Lemma D.2 (Improvement lower bound). *For any initial state distribution ξ_0, the sequence of policies generated by D.2 follows:*

$$J(\pi_{t+1}, h) - J\left(\pi_t, h\right) \geq \frac{1-\gamma}{\eta_t}\sum_{\bar{x}\in\bar{X}}\xi_0(\bar{x})\log Z_t(\bar{x})$$

Proof. First, note that $\log Z_t(\bar{x})$ is non-negative, which results from the concavity of the logarithmic function and $\sum_{a\in A}\pi_t(a \mid \bar{x})A_h^t(\bar{x}, a) = 0$:

$$\begin{aligned}
\log Z_t(\bar{x}) &= \log\sum_{a\in A}\pi_t(a \mid \bar{x})\exp\left(\eta_t A_h^t(\bar{x}, a)/(1-\gamma)\right)\\
&\geq \sum_{a\in A}\pi_t(a \mid \bar{x})\log\exp\left(\eta_t A_h^t(\bar{x}, a)/(1-\gamma)\right)\\
&= \frac{\eta_t}{1-\gamma}\sum_{a\in A}\pi_t(a \mid \bar{x})A_h^t(\bar{x}, a) = 0
\end{aligned}$$

Then for any ξ_0:

$$\begin{aligned}
&J(\pi_{t+1}, h) - J(\pi_t, h)\\
&= \frac{1}{1-\gamma} \sum_{\bar{x} \in \bar{X}} d_{\xi_0}^{\pi_{t+1}}(\bar{x}) \sum_{a \in A} \pi_{t+1}(a \mid \bar{x}) \big(A_h^t(\bar{x}, a)\big)\\
&= \frac{1}{\eta_t} \sum_{\bar{x} \in \bar{X}} d_{\xi_0}^{\pi_{t+1}}(\bar{x}) \sum_{a \in A} \pi_{t+1}(a \mid \bar{x}) \log \big(\pi_{t+1}(a \mid \bar{x}) Z_t(\bar{x}) / \pi_t(a \mid \bar{x})\big)\\
&= \frac{1}{\eta_t} \sum_{\bar{x} \in \bar{X}} d_{\xi_0}^{\pi_{t+1}}(\bar{x}) \big(\mathrm{D}_{\mathrm{KL}}\big(\pi_{t+1}(\cdot \mid \bar{x}) \| \pi_t(\cdot \mid \bar{x})\big) + \log Z_t(\bar{x})\big)\\
&\geq \frac{1}{\eta_t} \sum_{\bar{x} \in \bar{X}} d_{\xi_0}^{\pi_{t+1}}(\bar{x}) \log Z_t(\bar{x})\\
&\overset{(a)}{\geq} \frac{1-\gamma}{\eta_t} \sum_{\bar{x} \in \bar{X}} \xi_0(\bar{x}) \log Z_t(\bar{x}).
\end{aligned}$$

where (a) uses the fact that $d_{\xi_0}^{\pi}(\bar{x}) = (1-\gamma) \sum_t \gamma^t \mathbb{P}(\bar{x}_t = \bar{x}) \geq (1-\gamma)\xi_0(\bar{x})$ and $\log Z_t(\bar{x}) \geq 0$. □

Lemma D.3 (Improvement upper bound). *The sequence of policies generated by Eq. (D.2) follows:*

$$J(\pi_{t+1}, h) - J(\pi_t, h) \leq \eta_t \big(\phi(0)(G_{\max} - G_{\min})\big)^2 / (1-\gamma)^2$$

Proof.

$$\begin{aligned}
&J(\pi_{t+1}, h) - J(\pi_t, h)\\
&= \frac{1}{1-\gamma} \sum_{\bar{x}} d_{\xi_0}^{\pi_{t+1}}(\bar{x}) \sum_{a} \pi_{t+1}(a \mid \bar{x}) \big(A_h^t(\bar{x}, a)\big)\\
&= \frac{1}{1-\gamma} \sum_{\bar{x}} d_{\xi_0}^{\pi_{t+1}}(\bar{x}) \sum_{a} \big(\pi_{t+1}(a \mid \bar{x}) - \pi_t(a \mid \bar{x})\big) \big(A_h^t(\bar{x}, a)\big)\\
&\overset{(a)}{\leq} \frac{1}{1-\gamma} \sum_{\bar{x}} d_{\xi_0}^{\pi_{t+1}}(\bar{x}) \max_a \left|A_h^t(\bar{x}, a)\right| \sum_a \left|\pi_{t+1}(a \mid \bar{x}) - \pi_t(a \mid \bar{x})\right|\\
&\overset{(b)}{\leq} \frac{\|\theta_{t+1} - \theta_t\|_\infty}{1-\gamma} \left\|A_h^t\right\|_\infty\\
&\overset{(c)}{=} \frac{\eta_t}{(1-\gamma)^2} \left\|A_h^t\right\|_\infty^2\\
&\leq \frac{\eta_t}{(1-\gamma)^2} \big(\phi(0)(G_{\max} - G_{\min})\big)^2.
\end{aligned}$$

In this derivation, (a) follows Hölder's inequality, (b) follows the fact that $\|\pi_{t+1} - \pi_t\|_1 \leq \|\theta_{t+1} - \theta_t\|_\infty$ (Mei et al., 2020, Lemma 24), and (c) is resulted from policy updates in Eq. (D.2). □

By combining the results of Lemma D.2 (with the state occupancy measure of the optimal policy ($d_{\xi_0}^{\pi^*}$) as the initial state distribution) and Lemma D.3, we obtain:

$$\frac{1-\gamma}{\eta_t} \sum_{\bar{x} \in \bar{X}} d_{\xi_0}^{\pi^*}(\bar{x}) \log Z_t(\bar{x}) \leq \frac{\eta_t}{(1-\gamma)^2} \big(\phi(0)(G_{\max} - G_{\min})\big)^2. \tag{D.3}$$

Using this, we can show that for any policy π_t and the optimal policy π^*, we have:

$$\begin{aligned}
&J(\pi^*, h) - J(\pi_t, h)\\
&= \frac{1}{1-\gamma} \sum_{\bar{x} \in \bar{X}} d_{\xi_0}^{\pi^*}(\bar{x}) \sum_{a \in A} \pi^*(a \mid \bar{x}) \big(A_h^t(\bar{x}, a)\big)\\
&= \frac{1}{\eta_t} \sum_{\bar{x} \in \bar{X}} d_{\xi_0}^{\pi^*}(\bar{x}) \sum_{a \in A} \pi^*(a \mid \bar{x}) \log \big(\pi_{t+1}(a \mid \bar{x}) Z_t(\bar{x}) / \pi_t(a \mid \bar{x})\big)\\
&= \frac{1}{\eta_t} \sum_{\bar{x} \in \bar{X}} d_{\xi_0}^{\pi^*}(\bar{x}) \big(\mathrm{D}_{\mathrm{KL}}\big(\pi^*(\cdot \mid \bar{x}) \| \pi_t(\cdot \mid \bar{x})\big)\\
&\quad - \mathrm{D}_{\mathrm{KL}}\big(\pi^*(\cdot \mid \bar{x}) \| \pi_{t+1}(\cdot \mid \bar{x})\big) + \log Z_t(\bar{x})\big)\\
&\leq \frac{1}{\eta_t} \sum_{\bar{x} \in \bar{X}} d_{\xi_0}^{\pi^*}(\bar{x}) \big(\mathrm{D}_{\mathrm{KL}}\big(\pi^*(\cdot \mid \bar{x}) \| \pi_t(\cdot \mid \bar{x})\big)\\
&\quad - \mathrm{D}_{\mathrm{KL}}\big(\pi^*(\cdot \mid \bar{x}) \| \pi_{t+1}(\cdot \mid \bar{x})\big)\big)\\
&\quad + \frac{\eta_t}{(1-\gamma)^3} \big(\phi(0)(G_{\max} - G_{\min})\big)^2
\end{aligned}$$

With $K = \frac{1}{(1-\gamma)^3} \big(\phi(0)(G_{\max} - G_{\min})\big)^2$, we can write:

$$\begin{aligned}
&\sum_t \big(\eta_t \big(J(\pi^*, h) - J(\pi_t, h)\big)\big)\\
&\leq \sum_{\bar{x} \in \bar{X}} d_{\xi_0}^{\pi^*}(\bar{x}) \mathrm{D}_{\mathrm{KL}}\big(\pi^*(\cdot \mid \bar{x}) \| \pi_0(\cdot \mid \bar{x})\big)\\
&\quad + \sum_t \frac{\eta_t^2}{(1-\gamma)^3} \big(\phi(0)(G_{\max} - G_{\min})\big)^2\\
&= \mathrm{D}_{\mathrm{KL}}\big(\pi^* \| \pi_0\big) + K \sum_t \eta_t^2
\end{aligned}$$

where we have

$$\mathrm{D}_{\mathrm{KL}}\big(\pi^* \| \pi_0\big) = \sum_{\bar{x} \in \bar{X}} d_{\xi_0}^{\pi^*}(\bar{x}) \mathrm{D}_{\mathrm{KL}}\big(\pi^*(\cdot \mid \bar{x}) \| \pi_0(\cdot \mid \bar{x})\big)$$

for brevity. Since π_0 assigns positive probability to all actions, $\mathrm{D}_{\mathrm{KL}}\big(\pi^* \| \pi_0\big)$ remains bounded. Furthermore, by the Robbins-Monro condition ($\sum_t \eta_t = \infty, \sum_t \eta_t^2 < \infty$), the right-hand side converges to a finite limit. Since the Robbins-Monro condition holds, we conclude that $\lim_{t \to \infty} J(\pi_t, h) = J(\pi^*, h)$, which implies that the policy π_t converges to the optimal policy.

Appendix E. Proof of Theorem 4.3

Proof. The alternating update of the policy π and the function h can be analyzed within the Block Cyclic Coordinate Ascent framework (Tseng, 2001; Wright, 2015). We treat the policy parameters, denoted by $\theta \in \Theta$, and the quantiles of G^π required to define the function h, denoted by $q \in Q = \big[G_{\min}, G_{\max}\big]^N$, with constraints $q_i \leq q_{i+1}, \forall i \in \{1, \ldots, N-1\}$, as two separate parameter blocks for optimizing the objective. A standard assumption used throughout the proof is that the parameter space $\bar{\Theta} := \{\theta \in \Theta, q \in Q \mid J(\pi_\theta, h_q) \geq J(\pi_{\theta_0})\}$ is compact, where θ_0 represents the initial policy parameter. Another standard assumption is $\sup_{\theta \in \Theta} \|\nabla_\theta \log \pi_\theta\| < \infty$. Combined with the boundedness of the advantage function and the policy gradient expression in Eq. (9), this ensures that $J(\pi_\theta, h_q)$ is Lipschitz continuous with respect to θ. Importantly, since this property does not depend on the specific quantiles defining h_q, we can conclude that $J(\pi_\theta)$ is also Lipschitz continuous with respect to θ.

First, we observe that monotonic policy improvement in this context can be demonstrated as follows:

$$\begin{aligned}
J(\pi_{k+1}) &= \max_{h \in \mathcal{H}'} J(\pi_{k+1}, h)\\
&\geq \mathbb{E}\big[h_{k+1}(G^{\pi_{k+1}})\big]\\
&\overset{(a)}{\geq} \mathbb{E}\big[h_{k+1}(G^{\pi_k})\big]\\
&\overset{(b)}{=} J(\pi_k)
\end{aligned}$$

Here, (a) follows from the fact that π_{k+1} is the maximizing policy when function h_{k+1} is used. (b) also follows from the fact that h_{k+1} is constructed using the quantiles of G^{π_k} to maximize $\mathbb{E}[h(G^{\pi_k})]$. The argument for parameterized policies follows similarly and shows that $J(\pi_{\theta_{k+1}}) \geq J(\pi_{\theta_k})$. Since $J(\pi_\theta)$ is bounded from above, the sequence $\{J(\pi_{\theta_k})\}_{k=1,\cdots}$ converges to some J^*.

The proof of the theorem's final result follows closely from Proposition 2 in Zhang et al. (2021). However, unlike their setting, our quantile vector q is defined implicitly through an inner optimization, and thus requires an adaptation of their argument to account for the subgradient structure of h_q. Using Theorem 4.1(c) of Tseng (2001), we know that for any convergent subsequence $\big\{(\theta_k, q_k)\big\}_{k \in \mathcal{K}}$, the limit $(\theta_{\mathcal{K}}, q_{\mathcal{K}})$ satisfies $\nabla_\theta J\big(\pi_{\theta_{\mathcal{K}}}, h_{q_{\mathcal{K}}}\big) = 0$ and $\nabla_q J\big(\pi_{\theta_{\mathcal{K}}}, h_{q_{\mathcal{K}}}\big) = \nabla_q \mathbb{E}\Big[h_q\big(G^{\pi_{\theta_{\mathcal{K}}}}\big)|_{q=q_{\mathcal{K}}}\Big] = 0$. Our objective is to show that $\nabla_\theta J\big(\pi_{\theta_{\mathcal{K}}}\big) = \nabla_\theta J\big(\pi_{\theta_{\mathcal{K}}}, h_{q_{\mathcal{K}}}\big) = 0$, which confirms that the subsequence $\{\theta_k\}_{k\in\mathcal{K}}$ converges to a stationary point of $J(\pi_\theta)$.

Since $h_q(z)$ is not differentiable at $z=q_i$, the condition $\nabla_{q_i}\mathbb{E}\left[h_q\left(G^{\pi_{\theta_\mathcal{K}}}\right)\right]=0$, interpreted in the sense of the subgradient containing zero, implies that q_i is a $\hat{\tau}_i$-quantile of $G^{\pi_{\theta_\mathcal{K}}}$ for non-zero w_i. This means that $F_{G^{\pi_{\theta_\mathcal{K}}}}(q_i^-)\le\hat{\tau}_i\le F_{G^{\pi_{\theta_\mathcal{K}}}}(q_i)$. To see why, note that $G^{\pi_{\theta_\mathcal{K}}}$ follows an N-quantile distribution with quantile locations represented as $g_\mathcal{K}=[g_1,g_2,\dots,g_N]$, where $F^{-1}_{G^{\pi_{\theta_\mathcal{K}}}}(\tau)=g_{\lceil\tau N\rceil}$. Additionally, recall the structure of $h_q(z)$:

$$h_q(z)=\sum_{i=1}^N w_i\left(q_i+\frac{1}{\hat{\tau}_i}(z-q_i)^-\right).$$

The subgradient of $h_q(z)$ with respect to q_i is given by:

$$\partial_{q_i}h_q(z)=\begin{cases}w_i\left(1-\frac{1}{\hat{\tau}_i}\right), & z<q_i,\\ w_i\left[1-\frac{1}{\hat{\tau}_i},1\right], & z=q_i,\\ w_i, & z>q_i.\end{cases}$$

For any random variable Z, the subgradient of $\mathbb{E}[h_q(Z)]$ becomes:

$$\begin{aligned}\partial_{q_i}\mathbb{E}\left[h_q(Z)\right]&=\mathbb{E}\left[\partial_{q_i}h_q(Z)\right]\\&=w_i\Bigg(\mathbb{P}(Z<q_i)\cdot(1-\frac{1}{\hat{\tau}_i})+\mathbb{P}(Z>q_i)\cdot 1\\&\quad+\mathbb{P}(Z=q_i)\cdot[1-\frac{1}{\hat{\tau}_i},1]\Bigg)\\&=w_i\Bigg(1-\mathbb{P}(Z=q_i)-\mathbb{P}(Z<q_i)\cdot(\frac{1}{\hat{\tau}_i})\\&\quad+\mathbb{P}(Z=q_i)\cdot[1-\frac{1}{\hat{\tau}_i},1]\Bigg)\\&=w_i\left[1-\mathbb{P}(Z\le q_i)\cdot(\frac{1}{\hat{\tau}_i}),1-\mathbb{P}(Z<q_i)\cdot(\frac{1}{\hat{\tau}_i})\right]\\&=w_i\left(1-\frac{1}{\hat{\tau}_i}\cdot\left[\mathbb{P}(Z<q_i),\mathbb{P}(Z\le q_i)\right]\right)\\&=w_i\left(1-\frac{1}{\hat{\tau}_i}\cdot\left[F_Z(q_i^-),F_Z(q_i)\right]\right).\end{aligned}$$

For non-zero w_i, the condition

$$0\in w_i\left(1-\frac{1}{\hat{\tau}_i}\cdot[F_{G^{\pi_{\theta_\mathcal{K}}}}(q_i^-),F_{G^{\pi_{\theta_\mathcal{K}}}}(q_i)]\right)$$

ensures that $\hat{\tau}_i\in[F_{G^{\pi_{\theta_\mathcal{K}}}}(q_i^-),F_{G^{\pi_{\theta_\mathcal{K}}}}(q_i)]$, which implies q_i is a $\hat{\tau}_i$-quantile of $G^{\pi_{\theta_\mathcal{K}}}$. Consequently, for non-zero w_i, the quantiles of $q_\mathcal{K}$ match with those of $g_\mathcal{K}$ and therefore, we can conclude that $\nabla_\theta J\left(\pi_{\theta_\mathcal{K}}\right)=\nabla_\theta J\left(\pi_{\theta_\mathcal{K}},h_{q_\mathcal{K}}\right)=0$.

Ultimately, to establish the existence of a convergent subsequence, we need to prove that $\Theta_0:=\{\theta\in\Theta\mid J(\pi_\theta)\ge J(\pi_{\theta_0})\}$ is compact. Let $\{\theta^i\}_{i=1,\dots}$ be a sequence within Θ_0 that converges to a limit θ^∞. For each i, define $q^i:=\arg\max_q J(\pi_{\theta^i},q)$. Since $J(\pi_\theta)$ is Lipschitz continuous with respect to θ, the sequence (θ^i,q^i) must converge to (θ^∞,q^∞). The compactness of $\bar{\Theta}$ guarantees that $J(\pi_{\theta^\infty},h_{q^\infty})\ge J(\pi_{\theta_0})$, given that $J(\pi_{\theta^i},h_{q^i})=J(\pi_{\theta^i})\ge J(\pi_{\theta_0})$ for all i. This implies that $J(\pi_{\theta^\infty})\ge J(\pi_{\theta_0})$, confirming that $\theta^\infty\in\Theta_0$ and Θ_0 is compact. This completes the proof. □

Appendix F. Proof of Theorem 4.4

Proof. The proof of this theorem closely follows the results of Peng et al. (2019) and Nair et al. (2021), which derive policy updates under a KL divergence constraint. Our formulation adapts this idea to the risk-sensitive setting by incorporating the risk-adjusted advantage $A_h^\pi(\bar{x},a)$, and we treat the Lagrange multiplier λ as a tunable hyperparameter. This adaptation enables us to retain the tractability of the update rule while aligning it with the underlying risk-sensitive objective. Our formulation thus generalizes the advantage-weighted actor-critic framework to account for risk preferences in the offline policy learning.

Suppose we write the inner optimization as a search to find a policy that maximizes the expected improvement $I_h(\pi)=(1-\gamma)(J(\pi,h)-J(\pi_k,h))$. This expected improvement lets us use the risk-sensitive performance difference lemma from the previous section to write the objective involving the advantage function. As in the proof of Theorem 4.2, we omit the subscript h from π_h for brevity.

$$I_h(\pi)=\mathbb{E}_{\bar{x}\sim d_\pi(\bar{x})}\mathbb{E}_{a\sim\pi(a|\bar{x})}\left[A_h^{\pi_k}(\bar{x},a)\right]$$

Following Schulman et al. (2015) and assuming that each policy π_k is close to π_β in term of the KL-divergence, we can estimate $I_h(\pi)$ by using the state distribution of π_β:

$$\hat{I}_h(\pi)=\mathbb{E}_{\bar{x}\sim d_{\pi_\beta}(\bar{x})}\mathbb{E}_{a\sim\pi(a|\bar{x})}\left[A_h^{\pi_k}(\bar{x},a)\right].$$

With this change, we derive the following constrained inner optimization problem:

$$\begin{aligned}&\arg\max_\pi\int_{\bar{x}}d_{\pi_\beta}(\bar{x})\int_a\pi(a\mid\bar{x})A_h^{\pi_k}(\bar{x},a)\,da\,d\bar{x}\\&\text{s.t.}\ \int_{\bar{x}}d_{\pi_\beta}(\bar{x})\mathrm{D}_{\mathrm{KL}}(\pi(\cdot\mid\bar{x})\|\pi_\beta(\cdot\mid\bar{x}))d\bar{x}\le\epsilon.\\&\qquad\int_a\pi(a\mid\bar{x})da=1,\quad\forall\bar{x}.\end{aligned}$$

The Lagrangian of this optimization problem can be written as:

$$\begin{aligned}\mathcal{L}_h(\pi,\lambda,\alpha)=&\int_{\bar{x}}d_{\pi_\beta}(\bar{x})\int_a\pi(a\mid\bar{x})A_h^{\pi_k}(\bar{x},a)\,da\,d\bar{x}\\&+\lambda\left(\epsilon-\int_{\bar{x}}d_{\pi_\beta}(\bar{x})\mathrm{D}_{\mathrm{KL}}(\pi(\cdot\mid\bar{x})\|\pi_\beta(\cdot\mid\bar{x}))d\bar{x}\right),\\&+\int_{\bar{x}}\alpha_{\bar{x}}\left(1-\int_a\pi(a\mid\bar{x})da\right)d\bar{x},\end{aligned}$$

where λ and $\alpha=\{\alpha_{\bar{x}}\mid\forall\bar{x}\in\bar{\mathcal{X}}\}$ are the Lagrange multipliers. The derivative of $\mathcal{L}(\pi,\lambda,\alpha)$ w.r.t $\pi(a\mid\bar{x})$ is

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial\pi(a\mid\bar{x})}=&\,d_{\pi_\beta}(\bar{x})A_h^{\pi_k}(\bar{x},a)-\lambda d_{\pi_\beta}(\bar{x})\log\pi(a\mid\bar{x})\\&+\lambda d_{\pi_\beta}(\bar{x})\log\pi_\beta(a\mid\bar{x})-\lambda d_{\pi_\beta}(\bar{x})-\alpha_{\bar{x}},\end{aligned}$$

and setting this derivative to zero and solving for $\pi(a\mid\bar{x})$ give rise to

$$\log\pi(a\mid\bar{x})=\frac{1}{\lambda}A_h^{\pi_k}(\bar{x},a)+\log\pi_\beta(a\mid\bar{x})-1-\frac{1}{d_{\pi_\beta}(\bar{x})}\frac{\alpha_{\bar{x}}}{\lambda}.$$

Therefore, we have:

$$\pi(a\mid\bar{x})=\pi_\beta(a\mid\bar{x})\exp\left(\frac{1}{\lambda}A_h^{\pi_k}(\bar{x},a)\right)\exp\left(-\frac{1}{d_{\pi_\beta}(\bar{x})}\frac{\alpha_{\bar{x}}}{\lambda}-1\right).$$

With $Z(\bar{x})=\exp\left(\frac{1}{d_{\pi_\beta}(\bar{x})}\frac{\alpha_{\bar{x}}}{\lambda}+1\right)$ as the partition function that normalizes the conditional action distribution, the optimal policy can be written as

$$\pi^*(a\mid\bar{x})=\frac{1}{Z(\bar{x})}\pi_\beta(a\mid\bar{x})\exp\left(\frac{1}{\lambda}A_h^{\pi_k}(\bar{x},a)\right).$$

Since $\int_{a'}\pi(a'\mid\bar{x})da'=1$, we also have

$$Z(\bar{x})=\int_{a'}\pi_\beta\left(a'\mid\bar{x}\right)\exp\left(\frac{1}{\lambda}A_h^{\pi_k}(\bar{x},a')\right)da'.$$

Finally, since we use function approximation to represent the optimal policy π^*, we can project the optimal policy into the parameterized policy spaces with the following problem:

$$\begin{aligned}\pi_{k+1}&=\arg\min_\pi\mathbb{E}_{\bar{x}\sim d_{\pi_\beta}(\bar{x})}\left[\mathrm{D}_{\mathrm{KL}}\left(\pi^*(\cdot\mid\bar{x})\|\pi(\cdot\mid\bar{x})\right)\right]\\&=\arg\min_\pi\mathbb{E}_{\bar{x}\sim d_{\pi_\beta}(\bar{x})}\left[\mathrm{D}_{\mathrm{KL}}\left(\frac{1}{Z(\bar{x})}\pi_\beta(a\mid\bar{x})\exp\left(\frac{1}{\lambda}A_h^{\pi_k}(\bar{x},a)\right)\|\pi(\cdot\mid\bar{x})\right)\right]\end{aligned}$$

$$= \arg\max_{\pi} \mathbb{E}_{\bar{x}\sim d_{\pi_\beta}(\bar{x})}\mathbb{E}_{a\sim\pi_\beta(a|\bar{x})}\left[\log \pi(a \mid \bar{x}) \exp\left(\frac{1}{\lambda}A_h^{\pi_k}(\bar{x},a)\right)\right]$$

□

Appendix G. Algorithm for OAC-SRM

```
Algorithm 3: OAC-SRM.
Input      : Batch size $M$, Number of quantiles $N$, Policy
             update frequency $d$, Target smoothing
             coefficient $v$, Number of policy updates $T_{inner}$,
             Number of function $h$ updates $T_{outer}$, Dataset $\mathcal{D}$,
             Lagrange multiplier $\lambda$
Initialize : Critic networks $G_{\theta_1}, G_{\theta_2}$ and Actor network
             $\pi_\theta = \mathcal{N}(f_\theta, \sigma)$ or $\pi_\theta = \text{Categorical}(\text{softmax}(f_\theta))$
             with random parameters $\theta_1, \theta_2, \theta$, Target
             network parameters $\theta'_1 \leftarrow \theta_1, \theta'_2 \leftarrow \theta_2, \theta' \leftarrow \theta$,
for 1 to $T_{outer}$ do
    // Update risk function
    $h = \tilde{h}_{\phi, G_{\theta_1}(x_0, \pi_\theta(x_0))}$
    for $t = 1$ to $T_{inner}$ do
        // Update Critic
        Sample mini-batch of $M$ transitions $(\bar{x}, a, r, \bar{x}')$
         from $\mathcal{D}$
        foreach transition in the mini-batch do
            Sample target action:
              $a' \sim \pi_{\theta'}(\bar{x}')$
            Compute Q-values for $k = 1, 2$:
              $Q_k = \mathbb{E}\left[h\left(s' + c' G_{\theta'_k}(\bar{x}', a')\right)\right] / c'$
            Select target quantile set:
              $G'(\bar{x}', a') = \begin{cases} G_{\theta'_1}(\bar{x}', a') & \text{if } Q_1 \le Q_2 \\ G_{\theta'_2}(\bar{x}', a') & \text{otherwise} \end{cases}$
            Compute target quantiles:
              $Y(\bar{x}, a) = r + \gamma G'(\bar{x}', a')$
        end
        Minimize quantile regression loss
         $\mathcal{L}(G_{\theta_k}, Y), k = 1, 2$ (Eq. (2)) for the mini-batch
        // Update Actor
        if $t \bmod d = 0$ then
            Compute the advantage function:
              $A(\bar{x}, a) = Q_1(\bar{x}, a) - \mathbb{E}_{\tilde{a}\sim\pi_{\theta'}}\left[Q_1(\bar{x}, \tilde{a})\right]$
            Update $\theta$ using the advantage function and
            the policy gradient in Eq. (10)
            Update target networks:
              $\theta'_k \leftarrow v\theta_k + (1 - v)\theta'_k, k = 1, 2 \quad \theta' \leftarrow v\theta + (1 - v)\theta'$
        end
    end
end
```

Appendix H. Algorithm for TD3-SRM and TD3BC-SRM

```
Algorithm 4: TD3-SRM & TD3BC-SRM.
Input      : Batch size $M$, Number of quantiles $N$, Noise
             parameters $\sigma, \tilde{\sigma}$, Clipping constant $c$, Policy
             update frequency $d$, Target smoothing
             coefficient $v$, Number of policy Updates $T_{inner}$,
             Number of function $h$ updates $T_{outer}$, Dataset $\mathcal{D}$,
             Behavior cloning parameter $\lambda$
Initialize : Critic networks $G_{\theta_1}, G_{\theta_2}$ and Actor network
             $\pi_\theta = \mathcal{N}(f_\theta, \sigma)$, with random parameters
             $\theta_1, \theta_2, \theta$, Target network parameters
             $\theta'_1 \leftarrow \theta_1, \theta'_2 \leftarrow \theta_2, \theta' \leftarrow \theta$, Dataset $\mathcal{D}$
for 1 to $T_{outer}$ do
    // Update risk function
    $h = \tilde{h}_{\phi, G_{\theta_1}(x_0, \pi_\theta(x_0))}$
    for $t = 1$ to $T_{inner}$ do
        // Collect New Data
        Observe state $\bar{x}$, select action $a = \pi_\theta(\bar{x}) + \epsilon$, where
         $\epsilon \sim \mathcal{N}(0, \sigma)$
        Execute $a$, observe reward $r$ and next state $\bar{x}'$
        Store transition $(\bar{x}, a, r, \bar{x}')$ into $\mathcal{D}$
        // Update Critic
        Sample mini-batch of $M$ transitions $(\bar{x}, a, r, \bar{x}')$
         from $\mathcal{D}$
        foreach transition in the mini-batch do
            Compute noisy target action:
              $a' = \pi_{\theta'}(\bar{x}') + \epsilon', \epsilon' \sim \text{clip}(\mathcal{N}(0, \tilde{\sigma}), -c, c)$
            Compute Q-values for $k = 1, 2$:
              $Q_k = \mathbb{E}\left[h\left(s' + c' G_{\theta'_k}(\bar{x}', a')\right)\right] / c'$
            Select target quantile set:
              $G'(\bar{x}', a') = \begin{cases} G_{\theta'_1}(\bar{x}', a') & \text{if } Q_1 \le Q_2 \\ G_{\theta'_2}(\bar{x}', a') & \text{otherwise} \end{cases}$
            Compute target quantiles:
            $Y(\bar{x}, a) = r + \gamma G'(\bar{x}', a')$
        end
        Minimize quantile regression loss
         $\mathcal{L}(G_{\theta_k}, Y), k = 1, 2$ (Eq. (2)) for the mini-batch
        // Update Actor
        if $t \bmod d = 0$ then
            Update $\theta$ using the Q values and the policy
             gradient in Eqs. (11) or (12)
            Update target networks:
              $\theta'_k \leftarrow v\theta_k + (1 - v)\theta'_k, k = 1, 2 \quad \theta' \leftarrow v\theta + (1 - v)\theta'$
        end
    end
end
```

References

Acerbi, C. (2002). Spectral measures of risk: A coherent representation of subjective risk aversion. *Journal of Banking and Finance*, *26*(7), 1505–1518. https://doi.org/10.1016/S0378-4266(02)00281-9

Agarwal, A., Kakade, S. M., Lee, J. D., & Mahajan, G., et al. (2021). On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *Journal of Machine Learning Research*, *22*(98), 1–76. http://jmlr.org/papers/v22/19-736.html.

Agarwal, R., Schuurmans, D., & Norouzi, M., et al. (2020). An optimistic perspective on offline reinforcement learning. In *Proceedings of the 37th international conference on Machine Learning* (pp. 104–114). PMLR. https://proceedings.mlr.press/v119/agarwal20c.html.

Almahdi, S., & Yang, S. Y. (2017). An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown. *Expert Systems with Applications*, *87*, 267–279. https://doi.org/10.1016/j.eswa.2017.06.023

Bai, C., Xiao, T., Zhu, Z., Wang, L., Zhou, F., Garg, A., He, B., Liu, P., & Wang, Z., et al. (2022). Monotonic quantile network for worst-case offline reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, (pp. 1–15). https://doi.org/10.1109/TNNLS.2022.3217189

Barth-Maron, G., Hoffman, M. W., Budden, D., Dabney, W., Horgan, D., Tb, D., Muldal, A., Heess, N., & Lillicrap, T., et al. (2018). Distributed distributional deterministic policy gradients. In *International conference on learning representations*. https://openreview.net/forum?id=SyZipzbCb.

Bäuerle, N., & Glauner, A. (2021). Minimizing spectral risk measures applied to Markov decision processes. *Mathematical Methods of Operations Research*, *94*(1), 35–69. https://doi.org/10.1007/s00186-021-00746-w

Bäuerle, N., & Ott, J. (2011). Markov decision processes with average-value-at-risk criteria. *Mathematical Methods of Operations Research*, *74*(3), 361–379. https://doi.org/10.1007/s00186-011-0367-0

Bäuerle, N., & Rieder, U. (2014). More risk-sensitive Markov decision processes. *Mathematics of Operations Research*, *39*(1), 105–120. https://doi.org/10.1287/moor.2013.0601

Bellemare, M. G., Dabney, W., & Munos, R., et al. (2017). A Distributional Perspective on Reinforcement Learning. In *Proceedings of the 34th international conference on machine learning* (pp. 449–458). PMLR. https://proceedings.mlr.press/v70/bellemare17a.html.

Bellemare, M. G., Dabney, W., & Rowland, M., et al. (2023). Distributional reinforcement learning. The MIT Press. https://doi.org/10.7551/mitpress/14207.001.0001

Bisi, L., Santambrogio, D., Sandrelli, F., Tirinzoni, A., Ziebart, B. D., & Restelli, M., et al. (2022). Risk-averse policy optimization via risk-neutral policy optimization. *Artificial Intelligence*, *311*(Complete). https://doi.org/10.1016/j.artint.2022.103765

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., & Zhang, Q. (2018). JAX: Composable transformations of Python+NumPy programs. https://github.com/jax-ml/jax.

Chow, Y., Ghavamzadeh, M., Janson, L., & Pavone, M., et al. (2018). Risk-constrained reinforcement learning with percentile risk criteria. *Journal of Machine Learning Research*, *18*(167), 1–51. http://jmlr.org/papers/v18/15-636.html.

Chow, Y., & Pavone, M. (2013). Stochastic optimal control with dynamic, time-consistent risk constraints. In *2013 American control conference* (pp. 390–395). https://doi.org/10.1109/ACC.2013.6579868

Chow, Y., Tamar, A., Mannor, S., & Pavone, M., et al. (2015). Risk-sensitive and robust decision-making: A CVaR optimization approach. In *Advances in neural information processing systems*. Curran Associates, Inc. (*vol. 28*). https://proceedings.neurips.cc/paper/2015/hash/64223ccf70bbb65a3a4aceac37e21016-Abstract.html.

Coache, A., & Jaimungal, S. (2023). Reinforcement learning with dynamic convex risk measures. *Mathematical Finance*. https://doi.org/10.1111/mafi.12388

Dabney, W., Ostrovski, G., Silver, D., & Munos, R., et al. (2018a). Implicit quantile networks for distributional reinforcement learning. In *Proceedings of the 35th international conference on machine learning* (pp. 1096–1105). PMLR. https://proceedings.mlr.press/v80/dabney18a.html.

Dabney, W., Rowland, M., Bellemare, M., & Munos, R., et al. (2018b). Distributional Reinforcement learning with quantile regression. *Proceedings of the AAAI conference on artificial intelligence*, *32*(1). https://doi.org/10.1609/aaai.v32i1.11791

Ernst, D., Stan, G.-B., Goncalves, J., & Wehenkel, L., et al. (2006). Clinical data based optimal STI strategies for HIV: A reinforcement learning approach. In *Proceedings of the 45th IEEE conference on Decision and Control* (pp. 667–672). https://doi.org/10.1109/CDC.2006.377527

Fu, J., Kumar, A., Nachum, O., Tucker, G., & Levine, S., et al. (2021). D4RL: Datasets for deep data-driven reinforcement learning. Preprint. https://doi.org/10.48550/arXiv.2004.07219

Fujimoto, S., & Gu, S. (2021). A minimalist approach to offline reinforcement learning. In *Advances in neural information processing systems*. https://openreview.net/forum?id=Q32U7dzWXpc.

Fujimoto, S., Hoof, H., & Meger, D., et al. (2018). Addressing function approximation error in actor-critic methods. In *Proceedings of the 35th international conference on machine learning* (pp. 1587–1596). PMLR. https://proceedings.mlr.press/v80/fujimoto18a.html.

Fujimoto, S., Meger, D., & Precup, D., et al. (2019). Off-policy deep reinforcement learning without exploration. In *Proceedings of the 36th international conference on machine learning* (pp. 2052–2062). PMLR. https://proceedings.mlr.press/v97/fujimoto19a.html.

Geramifard, A., Dann, C., Klein, R. H., Dabney, W., & How, J. P., et al. (2015). RLPy: A value-function-based reinforcement learning framework for education and research. *Journal of Machine Learning Research*, *16*(46), 1573–1578. http://jmlr.org/papers/v16/geramifard15a.html.

Greenberg, I., Chow, Y., Ghavamzadeh, M., & Mannor, S., et al. (2022). Efficient risk-averse reinforcement learning. In *Advances in neural information processing systems*. https://openreview.net/forum?id=LdAxczs3m0.

Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S., et al. (2018). Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th international conference on machine learning* (pp. 1861–1870). PMLR. https://proceedings.mlr.press/v80/haarnoja18b.html.

Hasselt, H. (2010). Double Q-learning. In *Advances in neural information processing systems*. Curran Associates, Inc. (*vol. 23*). https://papers.nips.cc/paper_files/paper/2010/hash/091d584fced301b442654dd8c23b3fc9-Abstract.html.

Henryk, G., & Silvia, M. (2006). On a relationship between distorted and spectral risk measures. https://mpra.ub.uni-muenchen.de/1940/.

Huang, S., Dossa, R. F. J., Ye, C., Braga, J., Chakraborty, D., Mehta, K., & Araújo, J. G. M., et al. (2022). CleanRL: High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of Machine Learning Research*, *23*(274), 1–18. http://jmlr.org/papers/v23/21-1342.html.

Huber, P. J. (1992). Robust estimation of a location parameter. In S. Kotz, & N. L. Johnson (Eds.), *Breakthroughs in statistics: Methodology and distribution* (pp. 492–518). New York, NY: Springer. https://doi.org/10.1007/978-1-4612-4380-9_35

Kakade, S., & Langford, J. (2002). Approximately optimal approximate reinforcement learning. In *Proceedings of the nineteenth international conference on machine learning* ICML '02 (pp. 267–274). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Kakade, S. M., et al. (2001). A natural policy gradient. In *Advances in neural information processing systems*. MIT Press (*vol. 14*). https://proceedings.neurips.cc/paper_files/paper/2001/hash/4b86abe48d358ecf194c56c69108433e-Abstract.html.

Keramati, R., Dann, C., Tamkin, A., & Brunskill, E., et al. (2020). Being optimistic to be conservative: Quickly learning a CVaR policy. *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*(04), 4436–4443. https://doi.org/10.1609/aaai.v34i04.5870

Kim, D., Cho, T., Han, S., Chung, H., Lee, K., & Oh, S., et al. (2024). Spectral-risk safe reinforcement learning with convergence guarantees. In *The Thirty-eighth annual conference on neural information processing systems*. https://openreview.net/forum?id=9JFSJitKC0.

Kostrikov, I., Nair, A., & Levine, S., et al. (2021). Offline reinforcement learning with implicit Q-learning. In *International conference on learning representations*. https://openreview.net/forum?id=68n2s9ZJWF8.

Kumar, A., Zhou, A., Tucker, G., & Levine, S., et al. (2020). Conservative Q-learning for offline reinforcement learning. In *Advances in neural information processing systems* (pp. 1179–1191). Curran Associates, Inc. (*vol. 33*). https://papers.neurips.cc/paper_files/paper/2020/hash/0d2b2061826a5df3221116a5085a6052-Abstract.html.

Levine, S., Kumar, A., Tucker, G., & Fu, J., et al. (2020). Offline reinforcement learning: Tutorial, review, and perspectives on open problems. Preprint. https://doi.org/10.48550/arXiv.2005.01643

Lim, S. H., & Malik, I. (2022). Distributional reinforcement learning for risk-sensitive policies. In *Advances in neural information processing systems*. https://openreview.net/forum?id=wSVEd3Ta42m.

Ma, X., Xia, L., Zhou, Z., Yang, J., & Zhao, Q., et al. (2020). DSAC: Distributional soft actor critic for risk-sensitive reinforcement learning. Preprint. https://doi.org/10.48550/arXiv.2004.14547

Ma, Y., Jayaraman, D., & Bastani, O., et al. (2021). Conservative offline distributional reinforcement learning. In *Advances in neural information processing systems* (pp. 19235–19247). Curran Associates, Inc. (*vol. 34*). https://proceedings.neurips.cc/paper/2021/hash/a05d886123a54de3ca4b0985b718fb9b-Abstract.html.

Mei, J., Xiao, C., Szepesvari, C., & Schuurmans, D., et al. (2020). On the global convergence rates of softmax policy gradient methods. In *Proceedings of the 37th international conference on Machine Learning* (pp. 6820–6829). PMLR. https://proceedings.mlr.press/v119/mei20b.html.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533H. https://doi.org/10.1038/nature14236

Moghimi, M., & Ku, H. (2025). Beyond CVaR: Leveraging static spectral risk measures for enhanced decision-making in distributional reinforcement learning. In *Forty-second international conference on machine learning*. https://openreview.net/forum?id=WeMpvGxXMn.

Morimura, T., Sugiyama, M., Kashima, H., Hachiya, H., & Tanaka, T. (2010). Nonparametric return distribution approximation for reinforcement learning. In *Proceedings of the 27th international conference on international conference on machine learning* ICML'10 (pp. 799–806). Madison, WI, USA: Omnipress.

Nair, A., Gupta, A., Dalal, M., & Levine, S., et al. (2021). AWAC: Accelerating online reinforcement learning with offline datasets. Preprint. https://doi.org/10.48550/arXiv.2006.09359

Park, H., Sim, M. K., & Choi, D. G., et al. (2020). An intelligent financial portfolio trading strategy using deep Q-learning. *Expert Systems with Applications*, *158*, 113573. https://doi.org/10.1016/j.eswa.2020.113573

Pendharkar, P. C., & Cusatis, P. (2018). Trading financial indices with reinforcement learning agents. *Expert Systems with Applications*, *103*, 1–13. https://doi.org/10.1016/j.eswa.2018.02.032

Peng, X. B., Kumar, A., Zhang, G., & Levine, S., et al. (2019). Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. Preprint. https://doi.org/10.48550/arXiv.1910.00177

Pflug, G. C., & Pichler, A. (2016). Time-consistent decisions and temporal decomposition of coherent risk functional. *Mathematics of Operations Research*, *41*(2), 682–699. https://doi.org/10.1287/moor.2015.0747

Pichler, A. (2015). Premiums and reserves, adjusted by distortions. *Scandinavian Actuarial Journal*, *2015*(4), 332–351. https://doi.org/10.1080/03461238.2013.830228

Pires, B. Á., Rowland, M., Borsa, D., Guo, Z. D., Khetarpal, K., Barreto, A., Abel, D., Munos, R., & Dabney, W., et al. (2025). Optimizing return distributions with distributional dynamic programming. Preprint. https://doi.org/10.48550/arXiv.2501.13028

Prudencio, R. F., Maximo, M. R. O. A., & Colombini, E. L., et al. (2023). A survey on offline reinforcement learning: taxonomy, review, and open problems. *IEEE Transactions on Neural Networks and Learning Systems*, (pp. 1). https://doi.org/10.1109/TNNLS.2023.3250269

Rigter, M., Lacerda, B., & Hawes, N., et al. (2023). One risk to rule them all: A risk-sensitive perspective on model-based offline reinforcement learning. *Advances in Neural Information Processing Systems*, *36*, 77520–77545. https://proceedings.neurips.cc/paper_files/paper/2023/hash/f49287371916715b9209fa41a275851e-Abstract-Conference.html.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., & Moritz, P., et al. (2015). Trust region policy optimization. In *Proceedings of the 32nd international conference on machine learning* (pp. 1889–1897). PMLR. https://proceedings.mlr.press/v37/schulman15.html.

Shapiro, A., Dentcheva, D., & Ruszczyński, A. P., et al. (2014). Lectures on stochastic programming: Modeling and theory. MOS-SIAM Series on Optimization (second edition ed.). Philadelphia, Pennsylvania: Society for Industrial and Applied Mathematics : Mathematical Optimization Society.

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., & Riedmiller, M., et al. (2014). Deterministic policy gradient algorithms. In *Proceedings of the 31st international conference on machine learning* (pp. 387–395). PMLR. https://proceedings.mlr.press/v32/silver14.html.

Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. Adaptive Computation and Machine Learning Series (second edition ed.). Cambridge, Massachusetts: The MIT Press.

Sutton, R. S., McAllester, D., Singh, S., & Mansour, Y., et al. (1999). Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*. MIT Press (*vol. 12*). https://papers.nips.cc/paper_files/paper/1999/hash/464d828b85b0bed98e80ade0a5c43b0f-Abstract.html.

Tamar, A., Chow, Y., Ghavamzadeh, M., & Mannor, S., et al. (2017). Sequential decision making with coherent risk. *IEEE Transactions on Automatic Control*, *62*(7), 3323–3338. https://doi.org/10.1109/TAC.2016.2644871

Tamar, A., Di Castro, D., & Mannor, S. (2012). Policy gradients with variance related risk criteria. In *Proceedings of the 29th international coference on international conference on machine learning* ICML'12 (pp. 1651–1658). Madison, WI, USA: Omnipress.

Tamar, A., Glassner, Y., & Mannor, S., et al. (2015). Optimizing the CVaR via Sampling. *Proceedings of the AAAI Conference on Artificial Intelligence*, *29*(1). https://doi.org/10.1609/aaai.v29i1.9561

Todorov, E., Erez, T., & Tassa, Y., et al. (2012). MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 5026–5033). https://doi.org/10.1109/IROS.2012.6386109

Tseng, P. (2001). Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, *109*(3), 475–494. https://doi.org/10.1023/A:1017501703105

Urpí, N. A., Curi, S., & Krause, A., et al. (2021). Risk-averse offline reinforcement learning. In *International conference on learning representations*. https://openreview.net/forum?id=TBIzh9b5eaz.

Wang, M., & Ku, H., et al. (2022). Risk-sensitive policies for portfolio management. *Expert Systems with Applications*, *198*, 116807. https://doi.org/10.1016/j.eswa.2022.116807

Wang, S. (1995). Insurance pricing and increased limits ratemaking by proportional hazards transforms. *Insurance: Mathematics and economics*, *17*(1), 43–54. https://doi.org/10.1016/0167-6687(95)00010-P

Wang, S. S. (2000). A class of distortion operators for pricing financial and insurance risks. *The Journal of Risk and Insurance*, *67*(1), 15–36. https://doi.org/10.2307/253675

Wright, S. J. (2015). Coordinate descent algorithms. *Mathematical Programming*, *151*(1), 3–34. https://doi.org/10.1007/s10107-015-0892-3

Zhang, S., Liu, B., & Whiteson, S., et al. (2021). Mean-variance policy iteration for risk-averse reinforcement learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, *35*(12), 10905–10913. https://doi.org/10.1609/aaai.v35i12.17302