Safety-constrained reinforcement learning with a distributional safety critic

Yang, Qisong; Simão, Thiago D; Tindemans, Simon H.; Spaan, Matthijs T.J.

DOI
10.1007/s10994-022-06187-8

Publication date
2022

Document Version
Final published version

Published in
Machine Learning

Citation (APA)
Yang, Q., Simão, T. D., Tindemans, S. H., & Spaan, M. T. J. (2022). Safety-constrained reinforcement learning with a distributional safety critic. *Machine Learning*, *112*(3), 859-887. https://doi.org/10.1007/s10994-022-06187-8

Machine Learning
https://doi.org/10.1007/s10994-022-06187-8

Safety-constrained reinforcement learning with a distributional safety critic

Qisong Yang[1,2] · Thiago D. Simão[1,3] · Simon H. Tindemans[2] · Matthijs T. J. Spaan[1]

Received: 17 November 2021 / Revised: 28 February 2022 / Accepted: 9 May 2022

Abstract
Safety is critical to broadening the real-world use of reinforcement learning. Modeling the safety aspects using a safety-cost signal separate from the reward and bounding the expected safety-cost is becoming standard practice, since it avoids the problem of finding a good balance between safety and performance. However, it can be risky to set constraints only on the expectation neglecting the tail of the distribution, which might have prohibitively large values. In this paper, we propose a method called Worst-Case Soft Actor Critic for safe RL that approximates the distribution of accumulated safety-costs to achieve risk control. More specifically, a certain level of conditional Value-at-Risk from the distribution is regarded as a safety constraint, which guides the change of adaptive safety weights to achieve a trade-off between reward and safety. As a result, we can compute policies whose worst-case performance satisfies the constraints. We investigate two ways to estimate the safety-cost distribution, namely a Gaussian approximation and a quantile regression algorithm. On the one hand, the Gaussian approximation is simple and easy to implement, but may underestimate the safety cost, on the other hand, the quantile regression leads to a more conservative behavior. The empirical analysis shows that the quantile regression method achieves excellent results in complex safety-constrained environments, showing good risk control.

1 Introduction

In traditional reinforcement learning (RL) problems (Sutton & Barto, 2018), agents can explore environments to learn optimal policies without safety concerns. However, unsafe interactions with environments are unacceptable in many safety-critical problems, for instance, an autonomous robot should never break equipment or harm humans. Even though RL agents can be trained in simulators, there are many real-world problems without simulators of sufficient fidelity. Constructing safe RL algorithms for dangerous

Editors: Dana Drachsler Cohen, Javier Garcia, Mohammad Ghavamzadeh, Marek Petrik, Philip S. Thomas.

✉ Qisong Yang
q.yang@tudelft.nl

Extended author information available on the last page of the article

environments is challenging because of the trial-and-error nature of RL (Pecka & Svoboda, 2014). In general, safety is still an open problem that hinders the wider application of RL (García & Fernández, 2015).

Ray et al. (2019) propose to make constrained optimization the main formalism of safe RL, where the reward function and cost function (related to safety) are distinct. This framework tries to mitigate the problem of designing a single reward function that needs to carefully select a trade-off between safety and performance, which is problematic in most instances (Roy et al., 2021). Then, the objective is changed to select a policy with maximum expected return among the policies that have a bounded expected cost-return. The issue with this approach is that the cost of individual episodes might exceed the given bound with a high probability, in particular we might observe some episodes with arbitrarily high costs. For safety-critical problems, it can be hazardous to use the expected cost-return as safety evaluation. Instead, better alternatives for safety-constrained RL are algorithms that compute policies based on varying risk requirements, specialized to risk-neutral or risk-averse behavior (Duan et al., 2020; Ma et al., 2020).

We propose a new criterion for *risk-averse constrained RL*, where we focus on the upper tail of the cost distribution, represented by the conditional Value-at-Risk (CVaR; Rockafellar & Uryasev, 2000). With the new formalism, we design the Worst-Case Soft Actor Critic (WCSAC) algorithm that uses a separate safety critic to estimate the distribution of accumulated cost to achieve risk control. In this way, policies can be optimized given different levels of CVaR, which determine the degree of risk aversion from a safety perspective. In addition, we use a Lagrangian formulation to constrain the risk during training.

We focus on off-policy algorithms since they require fewer samples to optimize a policy by using experiences from past policies, which can reduce the probability of performing unsafe interactions with the environment (Achiam et al., 2017; Schulman et al., 2015, 2017). Soft actor critic (SAC; Haarnoja et al., 2018a, b) is an off-policy method built on the actor critic framework, which encourages agents to explore by including a policy's entropy as a part of the reward. SAC-Lagrangian (Ha et al., 2020) combines SAC with Lagrangian methods to address safety-constrained RL with local constraints, i.e., constraints are set for each timestep instead of each episode. The SAC-Lagrangian can be easily generalized to constrain the expected cost-return, but it is not apt to handle the risk-averse setting.

To find a risk-averse policy, we investigate two safety critics that estimate the full cost-return distribution, which we can use to infer the CVaR cost-return as mentioned before. Namely, we investigate a Gaussian approximation and quantile regression (Dabney et al., 2018a). On the one hand, the Gaussian approximation is simple and easy to implement, but may underestimate the safety cost, on the other hand, the quantile regression leads to a more conservative behavior. Figure 1 shows a comparison of the three safety critic architectures. With the development in distributional RL, WCSAC is also possible to be further improved by directly deploying some new techniques in distribution-approximating.

Experimental analysis shows that by setting the level of risk control, our two WCSAC algorithms can both attain stronger adaptability (compared to expectation-based baselines) when facing RL problems with higher safety requirements. The two WCSAC algorithms can achieve safe behavior in environments with a Gaussian cost-return. Compared to the Gaussian approximation, the WCSAC with quantile regression has better performance in environments with non-Gaussian cost-return distribution, and shows better risk control in complex safety-constrained environments.

The main contributions of this article can be summarized as follows:

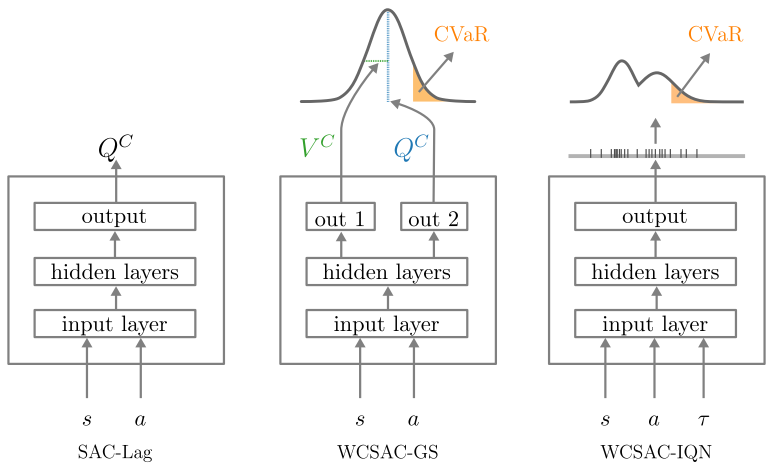


Fig. 1 Overview of the safety critics. The traditional safety critic only estimates the average of the cost-return distribution Q^C, while the critics of the WCSAC algorithms keep track of the full distribution, and estimate the CVaR for the safety constraints. The first assumes the distribution is Gaussian and the second uses the IQN algorithm for quantile regression

1. We propose a new criterion for *risk-averse constrained RL* to achieve risk control in safety critical problems.
2. We design an off-policy algorithm for *risk-averse constrained RL*, namely the WCSAC.
3. We show the versatility of WCSAC by using two methods to approximate the cost-return distribution.
4. We investigate the safety of these algorithms in environments with Gaussian and non-Gaussian cost-return distributions.

The above items 1 and 2 were partially covered in our previous work (Yang et al., 2021). We also improved the exposition of the core ideas of the paper. Taken as a whole, this paper demonstrates that the WCSAC algorithm is able to learn safe policies in a range of environments. In doing so, it highlights the value of using risk-averse metrics in RL algorithms.

2 Background

In this section, we formulate constrained RL problems, present algorithms used to solve them, and investigate how to estimate the distribution of long-term rewards/costs.

2.1 Constrained Markov decision processes

We formulate the safe RL problem as a Constrained Markov Decision Process (CMDP; Altman, 1999, Borkar, 2005), defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, c, d, T, \iota)$: where $\mathcal{S}$ is the state space and $\mathcal{A}$ is the action space. In a constrained RL an agent interacts with a CMDP, without knowledge about the transition, reward, and cost functions ($\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1], r : \mathcal{S} \times \mathcal{A} \to [r_{min}, r_{max}]$, and $c : \mathcal{S} \times \mathcal{A} \to [c_{min}, c_{max}]$). Each episode begins in a random state $s_0 \sim \iota : \mathcal{S} \to [0,1]$. At each timestep t of an episode, the agent observes the current state $s_t \in \mathcal{S}$, and takes an action $a_t \in \mathcal{A}$. Then, it observes a reward $r(s_t, a_t)$, a cost $c(s_t, a_t)$, and the next state $s_{t+1} \sim \mathcal{P}(\cdot \mid s_t, a_t)$. This process is repeated until some terminal condition is met, such as reaching the time horizon T. The behavior of the agent is defined by a policy $\pi : \mathcal{S} \times \mathcal{A} \to [0,1]$. This way, a policy π induces a distribution over full trajectories $\mathcal{T}_\pi = (s_0, a_0, s_1, \cdots)$ where $s_0 \sim \iota$, $a_t \sim \pi(\cdot \mid s_t)$, and $s_{t+1} \sim \mathcal{P}(\cdot \mid s_t, a_t)$.

In a CMDP there are two random variables of interest, the *return* $Z^r_\pi = \sum_{t=0}^{T} r(s_t, a_t)$ and the *cost-return* $Z^c_\pi = \sum_{t=0}^{T} c(s_t, a_t)$ that are, respectively, the sum of rewards and the sum of costs obtained in a trajectory following a fixed policy π.

Definition 1 (*Safety based on Expected Value*) A policy π is safe if its expected cost-return remains below a safety threshold d:

$$\mathbb{E}\left[Z^c_\pi\right] \le d$$

Over the episodes, the agent must learn a safe policy π that maximizes the expected return for each episode:

$$\max_\pi \mathbb{E}\left[Z^r_\pi\right] \quad \text{s.t.} \quad \mathbb{E}\left[Z^c_\pi\right] \le d. \tag{1}$$

For a complex and long-horizon problem ($T \gg 1$), it is common to introduce a discount factor $\gamma \in (0.0, 1.0)$ to make the problem tractable, since it allows the agent to compute a single stationary value function, instead of indexing it by the time step. Henceforth, we consider the discounted *return* and discounted *cost-return*, accumulated discounted rewards and costs, respectively, from (s, a) as

$$\begin{aligned} Z^r_\pi(s,a) &= \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s, a_0 = a, \text{ and} \\ Z^c_\pi(s,a) &= \sum_{t=0}^{\infty} \gamma^t c(s_t, a_t) \mid s_0 = s, a_0 = a. \end{aligned} \tag{2}$$

We will refer to the cost-return $Z^c_\pi(s,a)$ as C whenever π, s and a are clear from the context. So, we have $Q^r_\pi(s,a) = \mathbb{E}[Z^r_\pi(s,a)]$, and $Q^c_\pi(s,a) = \mathbb{E}[Z^c_\pi(s,a)] = \mathbb{E}[C]$.

2.2 SAC-lagrangian

When the agent knows nothing about the environment, the safety constraint cannot be strictly fulfilled during exploration. During the early steps of learning, we still hope to encourage exploration to learn more about the environment. But the policy's entropy must be carefully balanced with the safety constraint, and the policy must be allowed to converge to a relatively deterministic policy, which reduces risks in terms of (safety-related) cost. SAC-based methods with entropy constraints and adaptive entropy weights are candidates to meet these conditions. In this section, we describe the SAC-Lagrangian (SAC-Lag; Ha et al., 2020), a method designed for maximum entropy RL with safety constraints.

$$\begin{aligned} \max_{\pi} \quad & \mathbb{E}\left[Z^r_\pi\right] \\ \text{s.t.} \quad & \mathbb{E}[Z^c_\pi] \le \overline{d} \quad \text{and} \quad \mathop{\mathbb{E}}_{(s_t,a_t)\sim\mathcal{T}_\pi}\left[-\log(\pi_t(a_t \mid s_t))\right] \ge h \quad \forall t. \end{aligned} \tag{3}$$

Notice that we have a global constraint on the cost-return (over trajectories) and a local constraint on the policy entropy (for each time step).

In general, SAC-Lag is a SAC-based method that has two critics, where we use the reward critic to estimate the expected return (possibly with entropy) to promote reward during learning, while the safety critic estimates the cost-return to encourage safety. In SAC-Lag, the constrained optimization problems are solved by Lagrangian methods (Bertsekas, 1982). To manage a trade-off between exploration, reward, and safety, adaptive entropy and safety weights (Lagrange-multipliers) β and ω are introduced to the constrained optimization (1):

$$\min_{\beta\ge 0}\min_{\omega\ge 0}\max_{\pi} \mathcal{G}(\pi,\beta,\omega) \doteq f(\pi) - \beta e(\pi) - \omega g(\pi),$$

where $f(\pi) = \mathbb{E}_{s_0\sim\iota, a_0\sim\pi(\cdot|s_0)}\left[Z^r_\pi(\underline{s}_0, a_0)\right]$, $e(\pi) = \mathbb{E}_{(s_t,a_t)\sim\mathcal{T}_\pi}\left[\log(\pi(a_{\underline{t}} \mid s_t))\right] + h$, and $g(\pi) = \mathbb{E}_{s_0\sim\iota, a_0\sim\pi(\cdot|s_0)}\left[Z^c_\pi(s_0, a_0)\right] - \overline{d}$. h is the minimum entropy, and $\overline{d}$ is the discounted approximation of d, see Appendix A for details. The above max-min optimization problem is solved by gradient ascent on π, and descent on β and ω.

Ha et al. (2020) developed SAC-Lag for local constraints, which means that the safety cost is constrained at each timestep. However, it can be easily generalized to constrain the expected cost-return.[1] In this paper, we use J to denote loss functions, and θ to denote neural network parameters. Similar to the formulation used by Haarnoja et al. (2018b), we can get the actor loss:

$$J_\pi(\theta_\pi) = \mathop{\mathbb{E}}_{s_t\sim\mathcal{D}, a_t\sim\pi}[\beta \log \pi(a_t \mid s_t) - Q^r_\pi(s_t, a_t) + \omega Q^c_\pi(s_t, a_t)], \tag{4}$$

where the entropy weight β (Lagrange multiplier) manages the stochasticity of the policy π and also determines the relative importance of the entropy term compared to rewards and costs. $\mathcal{D}$ is the replay buffer and θ_π indicates the parameters of the policy π. Finally, let θ_ω and θ_β be the parameters learned for the safety and exploration weight such that $\omega = \mathrm{softplus}(\theta_\omega)$ and $\beta = \mathrm{softplus}(\theta_\beta)$, where

$$\mathrm{softplus}(x) = \log(\exp(x) + 1).$$

We can learn ω and β by minimizing the loss functions:

$$\begin{aligned} J_s(\theta_\omega) &= \mathop{\mathbb{E}}_{\substack{s_t \sim \iota \\ a_t \sim \pi}}\left[\omega(\overline{d} - Q^c_\pi(s_t, a_t))\right], \text{ and} \\ J_e(\theta_\beta) &= \mathop{\mathbb{E}}_{\substack{s_t \sim \mathcal{D} \\ a_t \sim \pi}}\left[-\beta(\log(\pi(a_t \mid s_t)) + h)\right]. \end{aligned} \tag{5}$$

[1] A similar approach has been used in the code available at https://github.com/openai/safety-starter-agents.

So the corresponding weight will be adjusted if the constraints are violated, that is, if we estimate that the current policy is unsafe or if it does not have enough entropy.

2.3 Distributional RL based on quantile regression

So far, we considered only the expected value of the return and the cost return. In this section we describe how we can estimate the full distribution of these random variables. Later, we will discuss how to use the tails of the cost-return to compute safer policies.

Distributional RL provides a means to estimate the return distribution instead of only modeling expected values (Bellemare et al., 2017; Dabney et al., 2018a, b; Yang et al., 2019). So it is natural to apply distributional RL in risk-averse domains. Even in traditional RL problems, distributional RL algorithms show better sample efficiency and ultimate performance compared to the standard expectation-based approach, but the state-of-the-art techniques have not been applied to safety-constrained RL with separate reward and safety signals.

Quantile regression, one of the main techniques in distributional RL, is widely used to estimate the return distribution, which has been combined with DQN (Mnih et al., 2015) to generate distributional variants such as QR-DQN (Dabney et al., 2018b), IQN (Dabney et al., 2018a), and FQF (Yang et al., 2019). In these methods, the difference between the distributions is measured by 1-Wasserstein distance:

$$W_1(u,v) \doteq \int_0^1 |F_u^{-1}(\chi) - F_v^{-1}(\chi)| d\chi, \tag{6}$$

where u and v are random variables (e.g., the return or cost-return), and F is the cumulative distribution function (CDF). In these methods, we learn the inverse CDF of the return distribution, i.e., mapping quantile fraction $\tau \in [0,1]$ to the corresponding quantile function value Z^τ,[2] which can be expressed as $Z^\tau = F_Z^{-1}(\tau)$. QR-DQN, IQN, and FQF differ in how to generate the quantile fractions during training. Compared to fixing the quantile fractions (QR-DQN) and random sampling (IQN), we can theoretically better approximate the real distribution by using a proposal network (FQF) that generates appropriate quantile fractions for each state-action pair. However, IQN has been found to perform better in experiments and has fewer parameters to tune in complex environments (Ma et al., 2020). The quantile values of IQN are learned based on the Huber quantile regression loss (Huber, 1964):

$$\rho_\tau^\kappa(\delta_{ij}) = |\tau - \mathbb{I}\{\delta_{ij} < 0\}| \frac{\mathcal{L}_\kappa(\delta_{ij})}{\kappa}, \text{ where } \mathcal{L}_\kappa(\delta_{ij}) = \begin{cases} \frac{1}{2}\delta_{ij}^2, & \text{if} \quad |\delta_{ij} \le \kappa| \\ \kappa(|\delta_{ij} - \frac{1}{2}\kappa|), & \text{otherwise} \end{cases}, \tag{7}$$

where κ is the threshold to make the loss within an interval $[-\kappa, \kappa]$ quadratic but a regular quantile loss if outside the interval. Based on the distributional Bellman operator (Morimura et al., 2010; Sobel, 1982; Tamar et al., 2016)

$$\mathcal{B}^\pi Z(s,a) \doteq r(s,a) + \gamma Z(s',a'), \tag{8}$$

[2] In this section, Z stands for the return $Z_\pi^r(s,a)$, but this method can easily be adapted to estimate the cost-return distribution.

we can get the TD error δ_{ij} between the quantile values at quantile fractions τ_i and τ'_j, i.e.,

$$\delta_{ij} = r(s,a) + \gamma Z^{\tau'_j}(s', \pi(s')) - Z^{\tau_i}(s,a), \tag{9}$$

where (s, a, r, a') is sampled from the replay buffer $\mathcal{D}$, and $\pi(s) = \arg\max_{a \in \mathcal{A}} Q^r(s,a)$. We can approximate $Q^r(s,a)$ using K i.i.d. samples of $\tilde{\tau} \sim U([0,1])$:

$$Q^r(s,a) \doteq \frac{1}{K} \sum_{k=1}^{K} Z^{\tilde{\tau}_k}(s,a).$$

It is important to note that τ, τ', and $\tilde{\tau}$ are sampled from continuous and independent distributions in IQN. τ' is for the TD target (average quantile values at several τ'), and τ is the given quantile we aim to estimate.

3 Risk-averse constrained RL

Traditional expectation-based safe RL methods maximize the return under the premise that the expected cost-return remains below the safety threshold d. In this way, RL agents are not aware of the potential risks because of the randomness in cost-return, which is generated by the stochastic policy and the dynamics of the environment. In expectation-based cases, if a safe policy has higher returns and higher variance in safety costs, it will be preferred over another safe policy with lower returns and lower variance in safety costs. In safety-critical domains, the optimal policies are expected to be more robust, i.e., to have a lower risk of hazardous events even for stochastic or heavy-tailed cost-return.

In Fig. 2, the x-axis depicts the cost-return C (Eq. 2). The y-axis depicts the density of its probability distribution. The expectation-based algorithm focuses on the average performance in safety when optimizing policies. Thus, π, Q^c_π, and the shape of the cost-return distribution $p^\pi(C \mid s,a)$ will be changed during the training process until Q^c_π (blue line) is shifted to the left side of the boundary (red line). After that, there is still a strong likelihood that the constraint value d is exceeded. For a policy π, Q^c_π can only be used as the evaluation of average performance in safety, however, in safety-critical domains, the worst-case performance in safety is preferred over the average performance. Therefore, we replace the expected value with the Conditional Value-at-Risk (CVaR; Rockafellar & Uryasev, 2000), using the upper α of the distribution to assess the safety of a policy. In the right panel of Fig. 2, we set the constraint on CVaR. Thus we optimize policies that will move the tail-end of $p^\pi(C \mid s,a)$ (blue line) to the left side of the boundary d (red line).

Definition 2 (*Risk level.*) A positive scalar $\alpha \in (0,1]$ is used to define the risk level in WCSAC. A WCSAC with smaller α ($\alpha \to 0$) is expected to be more pessimistic and risk-averse. Conversely, a larger value of α leads to a less risk-averse behavior, with $\alpha = 1$ corresponding to the risk-neutral case.

Considering the probability distribution of cost-returns $p^\pi(C)$ induced by the aleatoric uncertainty of the environment and the policy π, we model the safety-constrained RL problem in a more risk-averse way than the traditional formulation (1). We focus on the α-percentile $F_C^{-1}(1-\alpha)$, where F_C is the CDF of $p^\pi(C \mid s,a)$, so we can get the CVaR:

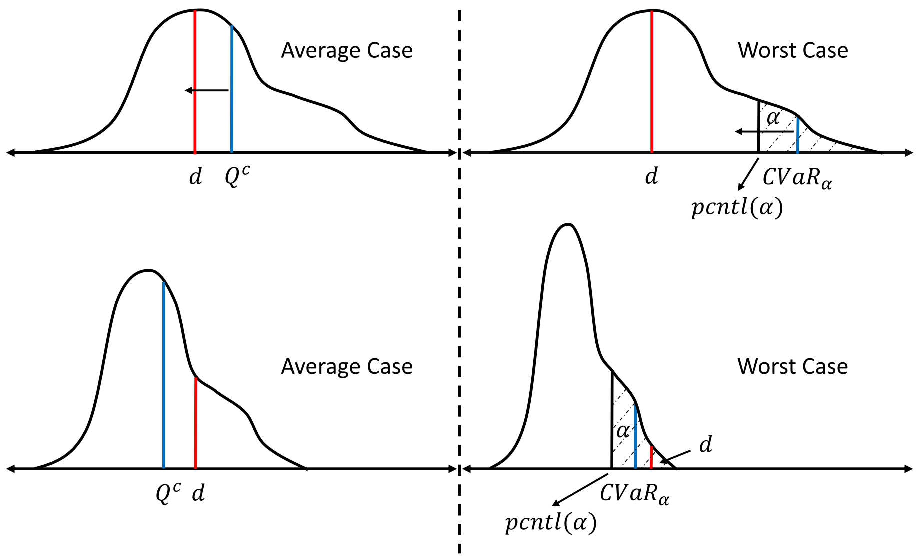


Fig. 2 In the average case (left panel), the policies are optimized to ensure Q^c (blue line) is moved to the left side of the fixed boundary d (red line). In the worst case (right panel), we optimize policies to ensure the $CVaR_\alpha$ (blue line) measure on the left side of the fixed boundary d (red line) (Color figure online)

$$\Gamma_\pi(s, a, \alpha) \doteq \mathrm{CVaR}^\alpha_\pi(C) = \mathop{\mathbb{E}}_{p^\pi}[C \mid C \geq F_C^{-1}(1-\alpha)]. \tag{10}$$

The following definition gives us a new constraint to learn risk-averse policies, which differs from the traditional constraint (1).

Definition 3 (*Safety based on CVaR*) Given the risk level α, a policy π is safe if it satisfies $\Gamma_\pi(s_t, a_t, \alpha) \leq \overline{d} \quad \forall t$, where $(s_t, a_t) \sim \mathcal{T}_\pi$ and $s_0 \sim \iota$.

Now we can generalize the framework from Sect. 2.2, using maximum entropy RL with the above risk-sensitive safety constraints. That is, the optimal policy in a constrained RL problem might be stochastic therefore it is reasonable to seek a policy with some entropy (Eq. 3). So, the policy is optimized to satisfy

$$\begin{aligned} \max_\pi \quad & \mathbb{E}\left[Z^r_\pi\right] \\ \text{s.t.} \quad & \mathrm{CVaR}^\alpha_\pi(C) \leq \overline{d} \quad \text{and} \quad \mathop{\mathbb{E}}_{(s_t,a_t)\sim\mathcal{T}_\pi}\left[-\log(\pi_t(a_t \mid s_t))\right] \geq h \quad \forall t. \end{aligned} \tag{11}$$

With (11) it is possible to solve safe RL problems using the Soft Actor Critic (SAC; Haarnoja et al., 2018a) framework, maintaining a minimum expected entropy (Haarnoja et al., 2018b).

4 Worst-case soft actor critic

To solve the *risk-averse constrained RL* problem (11), we design the Worst-Case Soft Actor Critic (WCSAC) algorithm. WCSAC generalizes SAC-Lag (Sect. 2.2), because SAC-Lag can be regarded as WCSAC with $\alpha = 1$, such that $\Gamma_\pi(s, a, 1) = Q^c_\pi(s, a)$ (10). In this section, we start by describing a safety critic that assumes the cost-return distribution is Gaussian, then we show how to handle the case where the cost-return distribution is not Gaussian using a quantile regression approach. Finally, we show how to optimize the actor with the new safety critics and present an overview of the full algorithm.

4.1 Gaussian safety critic

In this section, we present how to obtain a Gaussian approximation of the safety critic. We will refer to the WCSAC with a Gaussian safety critic as WCSAC-GS in the following parts of the paper.

4.1.1 Gaussian approximation

WCSAC-GS uses a separate Gaussian safety critic (parallel to the reward critic for the return) to estimate the distribution of C instead of computing a point estimate of the expected cost-return, as the SAC-Lag algorithm. To obtain the cost-return distribution, $p^\pi(C \mid s, a)$ is approximated with a Gaussian, i.e.,

$$Z^c_\pi(s, a) \sim \mathcal{N}(Q^c_\pi(s, a), V^c_\pi(s, a)), \tag{12}$$

where $V^c_\pi(s, a) = \mathbb{E}_{p^\pi}[C^2 \mid s, a] - (Q^c_\pi(s, a))^2$ is the variance of the cost-return.

Given the Gaussian approximation, the CVaR measure is easily computed (Khokhlov, 2016; Tang et al., 2020). At each iteration, $Q^c_\pi(s, a)$ and $V^c_\pi(s, a)$ can be estimated. Thus, the new safety measure for risk level α is computed by

$$\Gamma_\pi(s, a, \alpha) \doteq Q^c_\pi(s, a) + \alpha^{-1}\phi(\Phi^{-1}(\alpha))\sqrt{V^c_\pi(s, a)}, \tag{13}$$

where $\phi(\cdot)$ and $\Phi(\cdot)$ denote the probability distribution function (PDF) and the cumulative distribution function (CDF) of the standard normal distribution (Khokhlov, 2016).

WCSAC-GS learns the mean and variance of $p^\pi(C)$, as shown in Fig. 1. To estimate Q^c_π, we can use the standard Bellman function:

$$Q^c_\pi(s, a) = c(s, a) + \gamma \sum_{s' \in S} p(s' \mid s, a) \sum_{a' \in A} \pi(a' \mid s') Q^c_\pi(s', a'). \tag{14}$$

The projection equation for estimating $V^c_\pi(s, a)$ is:

$$
\begin{aligned}
V_\pi^c(s,a) = {} & c(s,a)^2 - Q_\pi^c(s,a)^2 \\
& + 2\gamma c(s,a) \sum_{s' \in S} p(s' \mid s,a) \sum_{a' \in A} \pi(a' \mid s') Q_\pi^c(s',a') \\
& + \gamma^2 \sum_{s' \in S} p(s' \mid s,a) \sum_{a' \in A} \pi(a' \mid s') V_\pi^c(s',a') \\
& + \gamma^2 \sum_{s' \in S} p(s' \mid s,a) \sum_{a' \in A} \pi(a' \mid s') Q_\pi^c(s',a')^2.
\end{aligned} \tag{15}
$$

We refer the reader to Tang et al. (2020) for the proof of (15).

4.1.2 Gaussian safety critic learning

WCSAC-GS uses two neural networks parameterized by θ_C^μ and θ_C^σ, respectively, to estimate the safety critic, i.e.,

$$Q^c_{\theta_C^\mu}(s,a) \rightarrow \hat{Q}_\pi^c(s,a) \text{ and } V^c_{\theta_C^\sigma}(s,a) \rightarrow \hat{V}_\pi^c(s,a).$$

In order to learn the safety critic, the distance between value distributions is measured by the 2-Wasserstein distance (Bellemare et al., 2017; Olkin & Pukelsheim, 1982): $W_2(u,v) \doteq \left(\int_0^1 |F_u^{-1}(\chi) - F_v^{-1}(\chi)|^2 d\chi \right)^{1/2}$, where $u \sim \mathcal{N}(Q_1, V_1)$, $v \sim \mathcal{N}(Q_2, V_2)$. WCSAC-GS uses the simplified 2-Wasserstein distance (Tang et al., 2020) to estimate the safety critic loss:

$$W_2(u,v) = \|Q_1 - Q_2\|_2^2 + trace(V_1 + V_2 - 2(V_2^{1/2} V_1 V_2^{1/2})^{1/2}). \tag{16}$$

The 2-Wasserstein distance can be computed as the Temporal Difference (TD) error based on the projection Eqs. (14) and (15) to update the safety critic, i.e., WCSAC-GS minimizes the following values:

$$
\begin{aligned}
J_C^\mu(\theta_C^\mu) &= \mathop{\mathbb{E}}_{(s_t,a_t)\sim\mathcal{D}} \|\Delta Q(s_t,a_t,\theta_C^\mu)\|_2^2, \text{ and} \\
J_C^\sigma(\theta_C^\sigma) &= \mathop{\mathbb{E}}_{(s_t,a_t)\sim\mathcal{D}} \text{trace}(\Delta V(s_t,a_t,\theta_C^\sigma)),
\end{aligned} \tag{17}
$$

where $J_C^\mu(\theta_C^\mu)$ is the loss function of $Q^c_{\theta_C^\mu}$, and $J_C^\sigma(\theta_C^\sigma)$ is the loss function of $V^c_{\theta_C^\sigma}$. So,

$$\Delta Q(s_t,a_t,\theta_C^\mu) = \overline{Q}^c_{\theta_C^\mu}(s_t,a_t) - Q^c_{\theta_C^\mu}(s_t,a_t), \tag{18}$$

where $\overline{Q}^c_{\theta_C^\mu}(s_t,a_t)$ is the TD target from (14), and

$$
\begin{aligned}
\Delta V(s_t,a_t,\theta_C^\sigma) = {} & \overline{V}^c_{\theta_C^\sigma}(s_t,a_t) + V^c_{\theta_C^\sigma}(s_t,a_t) \\
& - 2(V^c_{\theta_C^\sigma}(s_t,a_t)^{1/2} \overline{V}^c_{\theta_C^\sigma}(s_t,a_t) V^c_{\theta_C^\sigma}(s_t,a_t)^{1/2})^{1/2},
\end{aligned} \tag{19}
$$

where $\overline{V}^c_{\theta_C^\sigma}(s_t,a_t)$ is the TD target from (15).

Unfortunately, this Gaussian approximation can be coarse in many domains. In the next section, we will investigate methods that provide precise estimates of the return distribution, which later we apply to estimate the cost-return distribution.

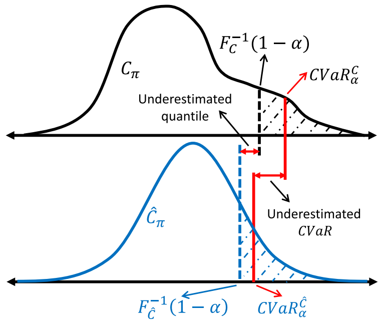


Fig. 3 Unreliability of Gaussian approximation. The top curve depicts the true cost distribution, while the bottom curve depicts the estimated Gaussian distribution, based on the correct mean and standard deviation. In this case, the $(1-\alpha)$-quantile and corresponding CVaR are underestimated

4.2 Safety critic with quantile regression

Although the Gaussian approximation leverages distributional information to attain more risk-averse policies, only an additional variance is estimated compared to regular constrained RL methods. This means the information of the experiences collected are only used to a limited extent. Thus, the Gaussian approximation does not possess the general advantages of distributional RL algorithms.

Besides, it is not always appropriate to approximate the cost-return by a Gaussian distribution, as shown in Fig. 3, since the contribution from the tail of the cost distribution might be underestimated. In this case, the agent might converge to an unsafe policy, according to (11). In this section, we present a distributional safety critic modeled by IQN, as illustrated on Fig. 1, which provides a more precise estimate of the upper tail part of the distribution. Henceforth, we refer to WCSAC with a safety critic modeled by IQN as WCSAC-IQN.

4.2.1 Estimating safety critic with IQN

We propose the *implicit quantile network* to model the cost-return distribution (safety-IQN), regarded as the safety critic of the SAC method. Safety-IQN maps the samples from a base distribution (usually $\tau \sim U([0,1])$) to the corresponding quantile values of the cost-return distribution. In theory, by adjusting the capacity of the neural network, safety-IQN can fit the cost-return distribution with arbitrary precision, which is essential for safety-critical problems.

We denote $F_C^{-1}(\tau)$ as the quantile function for the cost-return C and, for clarity of exposition, we define $C^\tau = F_C^{-1}(\tau)$. We use θ_C to parameterize the safety-IQN. The approximation is implemented as $\hat{C}^\tau(s,a) \leftarrow f_{IQN}(s,a,\tau \mid \theta_C)$, which also takes the quantile fraction τ as the input of the model, so that it uses the neural network to fit the entire continuous distribution. When training f_{IQN}, two quantile fraction samples $\tau, \tau' \sim U([0,1])$ at time step t are used to get the sampled TD error:

$$\delta_t^{\tau,\tau'} = c_t + \gamma C^{\tau'}(s_{t+1}, a_{t+1}) - C^\tau(s_t, a_t). \tag{20}$$

Following (7), we can get the loss function for safety-IQN, i.e.,

$$J_C(\theta_C) = \mathop{\mathbb{E}}_{(s_t,a_t,c_t,s_{t+1})\sim\mathcal{D}} \mathcal{I}_C(s_t, a_t, c_t, s_{t+1} \mid \theta_C), \tag{21}$$

where

$$
\begin{aligned}
\mathcal{I}_C(s_t, a_t, c_t, s_{t+1} \mid \theta_C) &\underset{(a)}{=} \sum_{i=1}^{N} \mathop{\mathbb{E}}_{\mathcal{B}^\pi C}[\rho^\kappa_{\tau_i}(\mathcal{B}^\pi C(s_t, a_t) - C^{\tau_i}(s_t, a_t))] \\
&\underset{(b)}{=} \sum_{i=1}^{N} \mathop{\mathbb{E}}_{C}[\rho^\kappa_{\tau_i}(c_t + \gamma C(s_{t+1}, a_{t+1}) - C^{\tau_i}(s_t, a_t))] \\
&\underset{(c)}{\doteq} \frac{1}{N'} \sum_{i=1}^{N} \sum_{j=1}^{N'} \rho^\kappa_{\tau_i}(c_t + \gamma C^{\tau'_j}(s_{t+1}, a_{t+1}) - C^{\tau_i}(s_t, a_t)) \\
&\underset{(d)}{=} \frac{1}{N'} \sum_{i=1}^{N} \sum_{j=1}^{N'} \rho^\kappa_{\tau_i}(\delta_t^{\tau_i, \tau'_j}).
\end{aligned}
\tag{22}
$$

In (22): (a) indicates that the total loss of all the target quantiles $\tau_i, i = 1, \cdots, N$ is computed at once, and applies the distributional Bellman operator $\mathcal{B}$ (Bellemare et al., 2017), (b) expands the Bellman operator, taking an action for the next state sampled from the current policy $a_{t+1} \sim \pi(\cdot \mid s_{t+1})$, (c) introduces τ_j to estimate the TD target, and (d) uses (20). We point out that for the estimation of quantiles, the quantile loss is replaced by the Huber loss to ease training, as in the regular IQN method (Dabney et al., 2018a). However, this may lead to a bias in the safety distribution (Rowland et al., 2019), especially for larger values of κ. The imputation approach proposed in the work by Rowland et al. (2019) can be combined with the proposed method to reduce this bias. Investigation of the extent of the bias and the efficacy of the correction in risk-averse RL is a subject for future research.

4.2.2 CVaR safety measure based on sample mean

Since we base our estimate of the distribution of cost-return on a quantile-parameterized approximation, we approximate the CVaR based on the expectation over the values of the quantile τ as $\Gamma_\pi(s, a, \alpha) \doteq \mathop{\mathbb{E}}_{\tau \sim U([1-\alpha, 1])} \left[C^\tau_\pi(s, a)\right]$. This allows us to estimate $\Gamma_\pi(s, a, \alpha)$ at each update step using K i.i.d. samples of $\widetilde{\tau} \sim U([1 - \alpha, 1])$:

$$
\Gamma_\pi(s, a, \alpha) \doteq \frac{1}{K} \sum_{k=1}^{K} C^{\widetilde{\tau}_k}_\pi(s, a). \tag{23}
$$

While the Gaussian approximation leverages a closed-form approach to estimate the CVaR, which is inherently limited by the Gaussian assumption, our method efficiently estimates the CVaR using a sampling approach. This can attain higher accuracy due to the quantile regression framework. We also highlight that this method still estimates the full distribution, sampling τ, τ' from $U([0, 1])$ to compute the safety critic loss. We use (23) only when estimating the CVaR to compute the Lagrangian safety loss J_s. In the next section, we describe how the actor uses the estimates of the CVaR described so far.

4.3 Worst-case actor

For a certain risk level α, we optimize the policy π until it satisfies the safety criterion $\Gamma_\pi(s_t, a_t, \alpha) \leq \overline{d} \quad \forall t$ according to Definition 3. In the policy improvement step, we update the policy towards the exponential of the new policy evaluation $X^\pi_{\alpha,\omega}(s, a) = Q^r_\pi(s, a) - \omega\Gamma_\pi(s, a, \alpha)$, based on the balance between safety and performance.

This particular choice of update can be guaranteed to result in an improved policy in terms of this new evaluation (Haarnoja et al., 2018a). The role of safety changes over the training process. As the policy becomes safe, the influence of the safety term wanes, then the return optimization will play a greater role in our formulation.

Since in practice we prefer tractable policies, we will additionally restrict the policy to some set of policies Π, which can correspond, for example, to a parameterized family of distributions such as Gaussians. To account for the constraint that $\pi \in \Pi$, we project the improved policy into the desired set of policies. In principle, we could choose any projection, but it turns out to be convenient to use the information projection defined in terms of the KL divergence (Dabney et al., 2018b; Kullback & Leibler, 1951). Similar to the work by Haarnoja et al. (2018b), for each state s_t, we try to minimize the following KL divergence to update the policy:

$$\begin{aligned} &\min_{\pi \in \Pi} D_{KL}\left(\pi(\cdot \mid s_t) \middle\| \frac{\exp\left(\frac{1}{\beta}(Q^r_\pi(s_t, \cdot) - \omega \Gamma_\pi(s_t, \cdot, \alpha))\right)}{\Xi^\pi(s_t)}\right) \\ &= \min_{\pi \in \Pi} D_{KL}(\pi(\cdot \mid s_t) \| \exp(\frac{1}{\beta} X^\pi_{\alpha,\omega}(s_t, \cdot) - \log(\Xi^\pi(s_t)))), \end{aligned} \tag{24}$$

where $\Xi^\pi(s_t)$ is the partition function to normalize the distribution. β and ω are the adaptive entropy and safety weights, respectively. A loss function can be constructed by averaging the KL divergence over all states in the sample buffer and approximating the KL divergence using a single sampled action, resulting in

$$\begin{aligned} &\mathop{\mathbb{E}}_{\substack{s_t \sim \mathcal{D} \\ a_t \sim \pi}} \left[-\log\left(\frac{\pi(a_t \mid s_t)}{\exp\left(\frac{1}{\beta} X^\pi_{\alpha,\omega}(s_t, a_t) - \log(\Xi^\pi(s_t))\right)}\right) \right] \\ &= \mathop{\mathbb{E}}_{\substack{s_t \sim \mathcal{D} \\ a_t \sim \pi}} \left[\log \pi(a_t \mid s_t) - \frac{1}{\beta} X^\pi_{\alpha,\omega}(s_t, a_t) + \log(\Xi^\pi(s_t)) \right]. \end{aligned} \tag{25}$$

$\Xi^\pi(s_t)$ has no influence on updating θ, thus it can be omitted. The resulting actor loss is

$$J_\pi(\theta) = \mathop{\mathbb{E}}_{\substack{s_t \sim \mathcal{D} \\ a_t \sim \pi}} \left[\beta \log \pi(a_t \mid s_t) - Q^r_\pi(s_t, a_t) + \omega \Gamma_\pi(s_t, a_t, \alpha) \right]. \tag{26}$$

The main difference to Eq. 4 is that we replace the expected cost by the CVaR estimate.

We update the reward critic Q^r and entropy weight β in the same way as the SAC method. The reward critic (including a bonus for the policy entropy) is trained to minimize

$$\begin{aligned} J_R(\theta_R) = &\mathop{\mathbb{E}}_{(s_t, a_t) \sim \mathcal{D}} \left[\frac{1}{2}(Q^r_{\theta_R}(s_t, a_t) - (r(s_t, a_t) \right. \\ &\left. + \gamma(Q^r_{\theta_R}(s_{t+1}, a_{t+1}) - \beta \log(\pi(a_{t+1} \mid s_{t+1})))))^2 \right], \end{aligned} \tag{27}$$

where Q^r is parameterized by θ_R, and $a_{t+1} \sim \pi(\cdot \mid s_{t+1})$. Based on the new safety measure, the safety weight ω can be learned by minimizing the loss function:

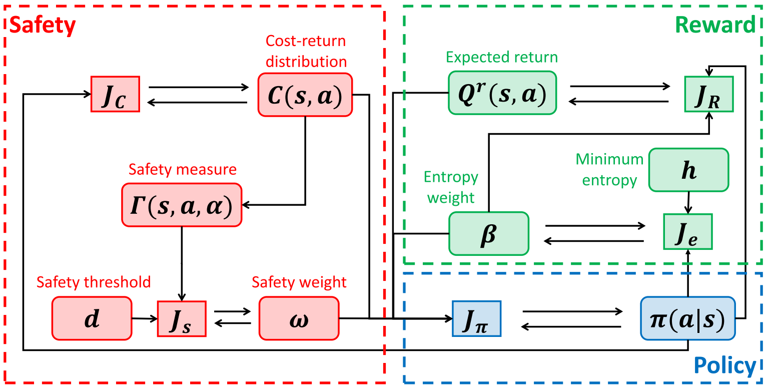


Fig. 4 WCSAC 's structure. The elements are grouped by function: Safety, Reward, and Policy

$$J_s(\theta_\omega) = \mathop{\mathbb{E}}_{\substack{s \sim \mathcal{D} \\ a \sim \pi}} \left[\omega(\overline{d} - \Gamma_\pi(s, a, \alpha)) \right], \tag{28}$$

so ω will be decreased if $\overline{d} \geq \Gamma_\pi(s, a, \alpha)$, otherwise ω will be increased to emphasize safety more. The main difference to how SAC-Lag optimizes its safety weight, is the use of the CVaR estimate, in opposite to the mean estimate (5). We note that in (28), we sample from the replay buffer $\mathcal{D}$, whereas (2) suggests that the constraint applies to the initial state distribution. This replacement is certainly valid in the strongly discounted regime, or when episodes are very long. In this case, each visited state can be considered an initial state for the cost calculation. Although the replay buffer may initially be strongly off-policy, this deviation reduces over time. Moreover, this replacement also turns out to work well in practice when these conditions do not apply.

Figure 4 shows a general overview of the proposed algorithm, indicating the relations between safety, reward, and policy components. The arrows depict the relations between all terms in the method, i.e., the element at the beginning of an arrow influences the element at its end. We may notice that the safety and reward terms only influence each other through the policy.

4.4 Complete algorithm

The complete algorithm WCSAC is presented in Algorithm 1, where we list the input of the algorithm and all initialization objects in lines 0-2. Under a certain safety requirement α, we input $\langle d, h \rangle$ for the constraints. With the WCSAC-IQN, we also need the hyperparameters $\langle N, N' \rangle$ for updating the safety-IQN, K for computing the new safety measure Γ_π. For the environment steps (lines 4-7), we sample actions from the policy to attain experience for the replay buffer $\mathcal{D}$, which allows us to get batches for updating all parameters at each gradient descent step (lines 8-21). After line 22, we list all the optimized parameters of the algorithm.

In standard maximum entropy RL, the entropy of the policy is expected to be as large as possible. However, relatively deterministic policies are preferred over stochastic policies in safe exploration, even though it is essential to encourage exploration during the early steps of learning. In SAC, the entropy of the policy is constrained to ensure that the final optimal policy is more robust (Haarnoja et al., 2018b). Therefore, for safety-critical domains, it is preferred to set a relatively low minimum requirement h for the entropy, or omit this constraint altogether.

With the Gaussian safety critic, we use two separate neural networks to estimate the mean function and variance function respectively. The size of each network can be smaller than using one network to estimate the mean function and variance together, so it does not add more parameters to be trained. In addition, it is much easier to compare the distributional safety critic of WCSAC-GS to the regular safety critic of SAC-Lag, which can be seen as an ablation of WCSAC-GS. As to the neural network structure of safety-IQN, we use the same function as in IQN for return (Dabney et al., 2018a), i.e., a DQN-like network with an additional embedding for the quantile fraction τ.

For the reward critic, to avoid overestimation and reduce the positive bias during the policy improvement process, we also learn two soft Q-functions independently, which are parameterized by θ_{R1} and θ_{R2}. The minimum Q-function is used in each gradient step. We leverage target networks (parameterized by $\overline{\theta}_R$ and $\overline{\theta}_C$) to achieve stable updating, a common technique used in DQN (Mnih et al., 2015) and DDPG (Lillicrap et al., 2015). Specifically, the parameters of target networks (including safety critic and reward critic) are updated by moving averages (lines 19-20), where hyperparameter $\eta \in [0, 1]$ is used to reduce fluctuations.

```
Algorithm 1 Worst-Case Soft Actor-Critic (WCSAC)
Require: Inputs: Hyperparameters $\alpha$, $d$, $h$, $\eta$
 1: Inputs: Hyperparameters $\langle N, N', K\rangle$ if WCSAC-IQN
 2: Initialize: $\theta_\pi$, $\theta_R$, $\theta_C$, $\theta_\beta$, $\theta_\omega$, $\langle\overline{\theta}_R, \overline{\theta}_C\rangle \leftarrow \langle\theta_R, \theta_C\rangle$, $\mathcal{D} \leftarrow \emptyset$
 3: for each iteration do
 4:     for each environment step do
 5:         $a_t \sim \pi(a_t \mid s_t)$, $s_{t+1} \sim \mathcal{P}(s_{t+1} \mid s_t, a_t)$
 6:         $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_t, a_t, r(s_t, a_t), c(s_t, a_t), s_{t+1})\}$
 7:     end for
 8:     for each gradient step do
 9:         Sample experience from replay buffer $\mathcal{D}$
10:         if WCSAC-IQN then
11:             Sample $\tau_i \sim U([0,1])$, $\tau'_j \sim U([0,1])$, and $\tilde{\tau}_k \sim U([\alpha,1])$
12:         end if
13:         $\theta_R \leftarrow \theta_R - \lambda_R \hat{\nabla}_{\theta_R} J_R(\theta_R)$            ▷ Reward critic (27)
14:         $\theta_C \leftarrow \theta_C - \lambda_C \hat{\nabla}_{\theta_C} J_C(\theta_C)$         ▷ Safety critic (17) ∨ (21)
15:         Compute $\Gamma_{\pi_\theta}$                           ▷ CVaR estimate (13) ∨ (23)
16:         $\theta_\pi \leftarrow \theta_\pi - \lambda_\pi \hat{\nabla}_{\theta_\pi} J_\pi(\theta_\pi)$                      ▷ Actor (26)
17:         $\theta_\beta \leftarrow \theta_\beta - \lambda_\beta \hat{\nabla}_{\theta_\beta} J_e(\theta_\beta)$        ▷ Exploration weight (5)
18:         $\theta_\omega \leftarrow \theta_\omega - \lambda_\omega \hat{\nabla}_{\theta_\omega} J_s(\theta_\omega)$            ▷ Safety weight (28)
19:         $\overline{\theta}_R \leftarrow \eta\theta_R + (1-\eta)\overline{\theta}_R$
20:         $\overline{\theta}_C \leftarrow \eta\theta_C + (1-\eta)\overline{\theta}_C$
21:     end for
22: end for
Output: Optimized parameters $\theta_\pi$, $\theta_R$, $\theta_C$, $\theta_\beta$, $\theta_\omega$
```

When selecting the learning rate for the neural networks (λ_R, λ_C, λ_π, λ_β, and λ_ω) which are used to minimize the corresponding loss functions (J_R, J_C, J_π, J_β, and J_ω), we usually make λ_ω larger than the others to enhance the safety constraint. A relatively low learning rate for the safety weight does not converge fast enough to improve the safety of the actor's policy, but the practical learning rate should be set according to the environment. Typically, the disparity between λ_ω and the remaining learning rates (λ_R, λ_C, λ_π, and λ_β) will be more pronounced in more complex and safety-critical environments.

5 Empirical analysis

In this section, we evaluate our methods WCSAC-GS and WCSAC-IQN on the tasks with different difficulties, i.e., two SpyGame environments and Safety Gym benchmark (Ray et al., 2019). This section has three goals: (i) test the hypothesis that WCSAC can achieve good risk control in an environment with a *Gaussian* cost-return distribution; (ii) test the hypothesis that WCSAC can find a safe policy on environments with a *non-Gaussian* cost-return distribution when equipped with an appropriate estimate of the distribution; and (iii) evaluate the performance of the proposed method in *highly-complex environments*.

5.1 SpyGame environments

To test whether the two WCSAC algorithms can achieve safe behavior in environments with a Gaussian cost-return, and whether WCSAC-IQN indeed has better performance in environments with non-Gaussian cost-return distribution, we designed two SpyGame environments: Spy-Unimodal and Spy-Bimodal. For any policy, Spy-Unimodal leads to a unimodal cost-return distribution (approximately Gaussian), while Spy-Bimodal has a bimodal cost-return distribution (non-Gaussian). The SpyGame is a toy model, meant to give rise to unimodal (approximately Gaussian) and bimodal cost distributions. For this model, we consider an agency that trains spies to go on covert missions. On each mission, the spy gets a random amount of useful information (the reward) and leaves some traces (the cost). If too many clues to the spy's identity are left across missions, the spy is likely to get discovered. In order to control the risk of discovery, a safety constraint is implemented on cumulative cost. For each mission, the spy has a choice of low-risk, low-reward and high-risk, high-reward approaches, parameterized by the action $a \in [0, 1]$. For a choice of a, random rewards and costs are drawn from uniform distributions as follows (Fig. 5):

$$r(a) \sim U(-0.25 + a, 0.75 + a + 0.5a^2) \tag{29}$$

$$c(a) \sim U(0.5a, 1.5a). \tag{30}$$

Two variants of the game are implemented in the `SpyEnv` environment, which are named *Spy-Unimodal* and *Spy-Bimodal* according to the shape of their cost distributions.[3]

Spy-Unimodal: Each spy executes 100 missions until retirement. The aim is to maximise expected reward subject to a cost constraint (in expectation or CVaR). The cumulative costs are a linear sum of a large number of independent random variables, so they are approximately normally distributed.

[3] The code of `SpyEnv` will be made available online.

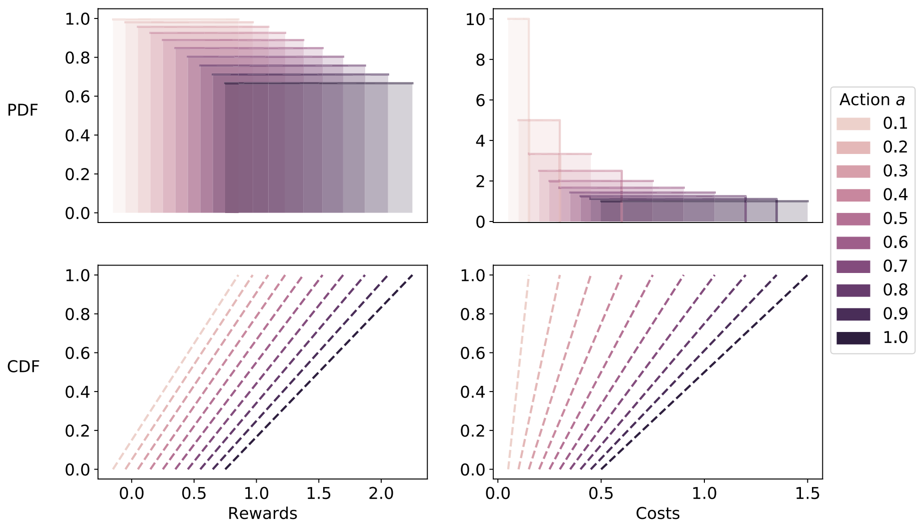


Fig. 5 The PDF and CDF of the reward and cost functions under different actions

Spy-Bimodal: In this variant, spies face early retirement if they do not gain sufficiently useful information. After 5 missions, a stopping criterion is evaluated that terminates the game unless the *average* reward per mission exceeds 0.15. This results in a significant fraction of spies retiring early, which is reflected in a bimodal cost distribution.

We set the safety thresholds $d = 25$ for Spy-Unimodal, and $d = 15$ for Spy-Bimodal. We use WCSAC-GS and WCSAC-IQN with risk-neutral and risk-averse constraints (cost-CVaR-α) to solve both variations of the SpyGame. Each algorithm uses small neural networks (2 layers with 16 units) and trains for 30000 steps. After training, we run each of the final policies for 10000 episodes to evaluate the cost-returns of our algorithms.

5.1.1 Cost-return distribution evaluation

In Fig. 6, we compare the two algorithms with risk-neutral ($\alpha = 1$) and risk-averse ($\alpha = 0.1$) constraints on both versions of the SpyGame. We report the distribution of cost-returns. This gives a clear overview of the full cost-return distribution, allowing us to evaluate the frequency the safety constraint is violated. We also report the metric used as safety constraints to verify when each agent can reach the designated safety requirements.

At the top of Fig. 6 (risk-neutral case), we can see that the two WCSAC algorithms approximately attain a constraint-satisfying expected cost-return in both environments, and the realised values are very close. So, in the average case, WCSAC-GS and WCSAC-IQN have similar performance independently of the underlying distribution.

At the bottom of Fig. 6 (risk-averse case), first we notice that in the Spy-Unimodal environment (Gaussian) both WCSAC algorithms attain a cost-CVaR-0.1 below the threshold. We can also notice that WCSAC-IQN is closer to the bound showing a slightly better control over the cost-CVaR-0.1. On Spy-Bimodal (non-Gaussian), WCSAC-GS is unable to satisfy the safety constraint, attaining a cost-CVaR-0.1 larger than the bound. This indicates that the Gaussian approximation cannot control the risk level in this domain.

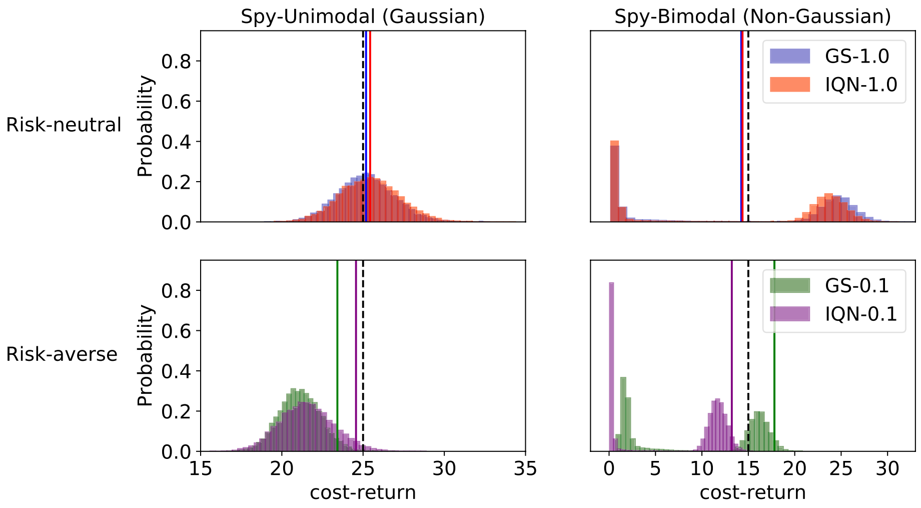


Fig. 6 Cost-return distribution in Spy-Unimodal (left) and Spy-Bimodal (right) considering risk-neutral constraints with CVaR-1.0 (top) and risk-averse constraints with CVaR-0.1 (bottom). Dashed lines show the imposed safety threshold d and solid lines the realised values for each method

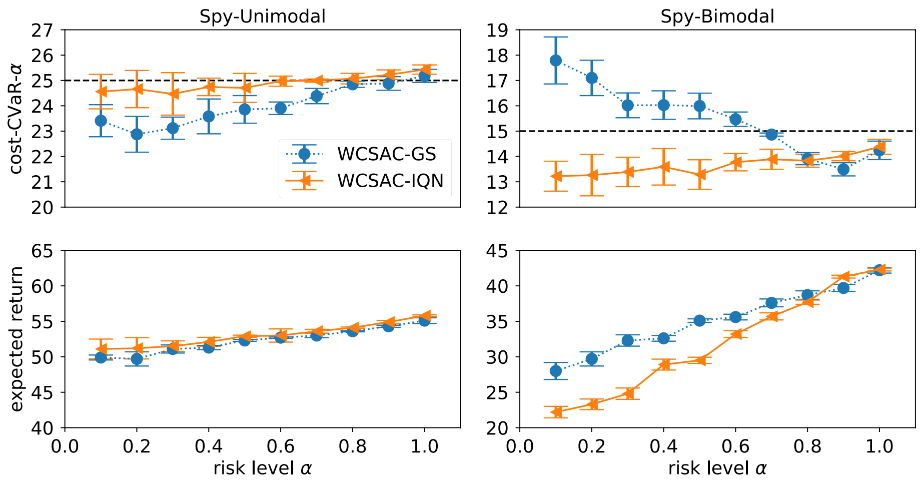
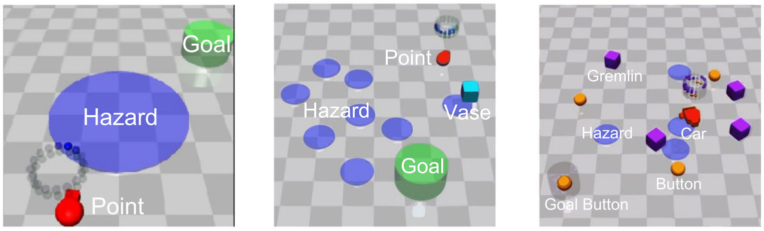


Fig. 7 Performance of WCSAC-GS and WCSAC-IQN in terms of different risk levels. The error bars are the standard deviation over 5 runs. Dashed lines show the imposed safety threshold d

Overall, comparing the top and bottom plots in Fig. 6, we can see that both WCSAC algorithms can attain a more risk-averse behavior by setting the risk level α to be a small value, reducing significantly the probability that a trajectory violates the safety constraints.

5.1.2 Varying level of safety constraint

To get a better overview of when the safety constraints are violated or not, we consider the same environment setting different risk level constraints. In Fig. 7, the x-axis depicts the



(a) StaticEnv **(b)** PointGoal **(c)** CarButton

Fig. 8 Robot navigation environments. The three environments differ in complexity, i.e., the action space, the type of the obstacles, and the task. The Point or Car agent should avoid Hazards (dangerous areas), Vases (fragile objects), and Gremlins (moving objects) in the above environments, then move to the Goal position or press the Goal Button

risk level α, under which the agents are trained. The y-axis depicts the corresponding cost-CVaR-α (Fig. 7 top) and expected return (Fig. 7 bottom) generated by the final policies and the standard deviation over 5 repetitions.

At the bottom of Fig. 7, we can see that, in the more risk-averse settings (lower value of α), WCSAC-GS and WCSAC-IQN will both have lower expected returns. In general, the changes in the cost-CVaR-α and expected return under different risk levels show the same trend, i.e., a larger cost-CVaR-α corresponds to a larger expected return at the risk level α.

When we have a unimodal cost-return distribution (left panel of Fig. 7), we can see that the WCSAC algorithms attain safe performance with different risk levels α. But when we have higher safety requirements, both WCSAC algorithms generate a greater variance, and the distance between the corresponding cost-CVaR-α and the safety bound d becomes larger. Compared to WCSAC-IQN, WCSAC-GS is more over-conservative with lower α.

In the right panel of Fig. 7, we show the results with a bimodal cost-return distribution, where a Gaussian approximation can underestimate the CVaR, as we have seen in the previous section. In this case, WCSAC-IQN is safe in all different α, and WCSAC-GS can also obtain safe performance for values closer to the risk-neutral constraint ($\alpha \in [0.7, 1.0]$). However, WCSAC-GS starts to become increasingly unsafe for lower values of α (more risk-averse constraints).

No matter in the unimodal or bimodal cases, both WCSAC algorithms approach the safety boundary better with higher α (more risk-neutral). But we have a greater deviation with lower α, especially in the bimodal case. Even with WCSAC-IQN, our safety performance is becoming more pessimistic when we decrease α. Based on the experimental results in the work by (Théate et al., 2021), it appears to be the case that the quantile regression methods may result in more approximation errors for higher-order moments compared to the first-order moment.

5.2 SafetyGym environments

Next, we evaluate our method in three domains from the *Safety Gym* benchmark suite (Ray et al., 2019), where a robot navigates in a 2D map to reach target positions while trying to avoid dangerous areas, with different complexity levels (Fig. 8). The first one is StaticEnv with one fixed hazard and one fixed goal, but the initial position of the Point agent is randomly generated at the beginning of each episode. The second is PointGoal (`Safexp-PointGoal1-v0` in *Safety Gym*) with one Point agent, several hazards, and one vase.

Table 1 Performance of policies after training in terms of expected return (ER), expected cost-return (EC), and cost-CVaR-0.5 (C0.5)

Env	StaticEnv ($d = 8$)			PointGoal ($d = 25$)			CarButton ($d = 25$)		
	ER	EC	C0.5	ER	EC	C0.5	ER	EC	C0.5
SAC	0.87	18.53	19.11	28.85	62.77	71.59	21.36	201.06	247.13
CPO	-0.63	**9.25**	14.71	21.48	**42.99**	50.39	3.62	**80.16**	116.31
PPO-Lag	-0.76	**6.31**	8.93	17.81	**24.47**	30.11	0.45	**26.18**	35.09
GS-1.0	0.75	**8.04**	13.42	23.08	**40.39**	61.19	11.45	**65.03**	80.86
IQN-1.0	0.46	**7.83**	9.61	15.46	**23.41**	28.74	1.40	**24.02**	32.34
GS-0.5	0.66	6.16	**7.30**	19.50	37.23	**38.29**	7.06	58.51	**62.97**
IQN-0.5	0.40	5.10	**7.18**	9.23	20.80	**22.60**	0.64	15.95	**20.13**

The metric of the safety constraint used by each agent are in bold

The third and more complex environment is CarButton (`Safexp-CarButton1-v0` in *Safety Gym*) where a Car robot (higher dimensional action space than Point) is navigating to press a goal button while trying to avoid hazards and moving gremlins, and not pressing any of the wrong buttons. These tasks are particularly complex due to the observation space, instead of observing its location, the agent has a lidar that indicates the distance to other objects. All experiments are performed with 10 random seeds. In all environments, $c = 1$ if an unsafe interaction happens, otherwise $c = 0$. We use the original reward signal in *Safety Gym*, i.e., the absolute distance towards the goal plus a constant for finishing the task, e.g., press the goal button.

We evaluate four versions of the WCSAC: GS-1.0 (WCSAC-GS with $\alpha = 1.0$), GS-0.5 (WCSAC-GS with $\alpha = 0.5$), IQN-1.0 (WCSAC-IQN with $\alpha = 1.0$), and IQN-0.5 (WCSAC-IQN with $\alpha = 0.5$). For comparison, we used SAC (Haarnoja et al., 2018b), CPO (Achiam et al., 2017), and PPO-Lagrangian (PPO-Lag Ray et al., 2019; Schulman et al., 2017) as baselines. In this experiment, we use the discount factor $\gamma = 0.99$ and $\kappa = 1$ for the Huber loss in WCSAC-IQN. The safety thresholds are set to be $d = 8$ for StaticEnv, and $d = 25$ for PointGoal and CarButton. We train each agent for 50 epochs in StaticEnv, and for 150 epochs in PointGoal and CarButton. The epoch length is 30000 environment steps, and the maximal episodic length is 1000 environment steps.[4]

To evaluate the performance of the algorithms, we use the following metrics: CVaR-0.5 of the cost-return (cost-CVaR-0.5), expected cost (AverageEpCost), and expected reward (AverageEpRet). Table 1 shows the performance of the policies returned by the algorithms after training. We use 1000 episodes (100 runs for each random seed) to evaluate the final policy of each method; the expected cost and expected return are estimated by the average of all runs, while the cost-CVaR-0.5 is estimated by the average of the worst 500 runs. In Fig. 9, we visualize the distribution by plotting the PDF and CDF histograms of sampled episodic costs in PointGoal and CarButton. Finally, Fig. 10 shows the behavior of the algorithms during training. We provide a collection of videos of the execution of the final policies on the following webpage: https://sites.google.com/view/wcsac.

[4] The code of WCSAC will be made available online.

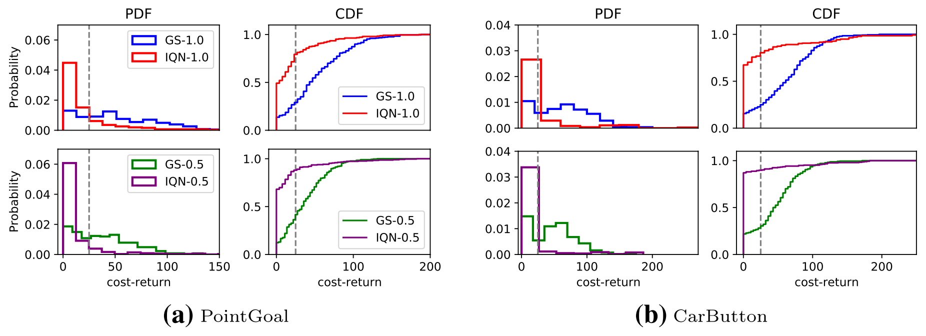


Fig. 9 Cost-return distributions in PointGoal and CarButton (based on the same data as Table 1). The x-axis depicts the cost-return, while the y-axis depicts the binned probability density and the cumulative density functions. The gray dashed line indicates the safety threshold

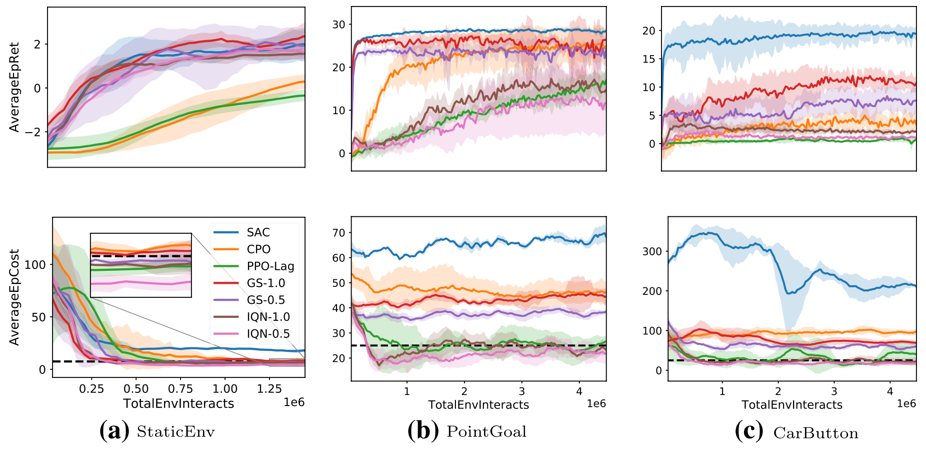


Fig. 10 Performance of the algorithms during training in terms of mean (solid lines) and ±1 standard deviation (shaded area) of the runs within an epoch. The black dashed lines indicate the safety thresholds

5.2.1 Final behavior

We start our analysis by considering the behavior of the final policy. In Table 1, we can see that only IQN-1.0 and IQN-0.5 can be considered safe, because they satisfy the cost constraint with which they trained in all environments. In particular, only IQN-0.5 satisfies the risk-averse threshold on cost-CVaR-0.5, demonstrating its suitability for risk-averse agents. PPO-Lag has competitive safety performance in all the environments, but fails to achieve a high return in StaticEnv and CarButton. Compared to the safe RL methods (CPO, PPO-Lag, and WCSAC), SAC has an excellent performance in expected return, but, naturally it does not satisfy the safety constraint, this shows that a safe agent must find a tradeoff between safety and performance. Although the final policies of the remaining algorithms CPO, GS-1.0 (SAC-Lag), and GS-0.5 may show better expected returns, these methods are not safe in PointGoal and CarButton.

In Fig. 9, we can observe that the distributions in PointGoal and CarButton are not Gaussian, which justifies the use of a quantile regression algorithm and the safe behavior of the WCSAC-IQN algorithms in these more complex environments. Compared to GS-1.0, GS-0.5, and IQN-1.0, the distribution of IQN-0.5 displays a smaller range of costs, most of which are within the safety bound. Although the policies from GS-0.5 still generate some unsafe trajectories, the likelihood is much lower.

5.2.2 Behavior during training

Figure 10 shows the behavior of the agents during training. The top row shows the expected return, while the bottom row shows the expected cost-return.

We can see that all safe RL methods manage to make some safety improvement, while SAC has better and stable performance in average episodic return obviously across all the environments since it ignores the safety constraints.

In StaticEnv (Fig. 10a), we notice that all safe RL algorithms converge toward the optimal policy. However, compared to the off-policy WCSAC, the on-policy baselines CPO and PPO-Lag take more time to do so. When we look closely, we notice that CPO and GS-1.0 exceed slightly the cost bound at the end of the training, while PPO-Lag, GS-0.5, and IQN-1.0 end slightly below the cost bound. In particular, we highlight that IQN-0.5 achieves a lower expected cost without sacrificing much performance in terms of return.

In PointGoal (Fig. 10b), we see a different behavior: only the PPO-Lag and WCSAC-IQN algorithms manage to find a satisfactory policy. Although WCSAC-GS and CPO manage to find policies with high returns, they fail to achieve safe behavior.

Finally, in the most complex environment, CarButton (Fig. 10c), we see that the cost constraints severely limit the ability to find high-reward policies: PPO-Lag, IQN-1.0, and IQN-0.5 manage to find safe policies, however, they cannot improve significantly in terms of return; GS-1.0 and GS-0.5 also approach a safe policy and manage to get some improvements in terms of return; and CPO does not find a safe policy whilst simultaneously struggling to improve returns.

As to the unsafe performance of CPO in PointGoal and CarButton, the approximation errors in CPO may prevent it from fully satisfying constraints in these environments, which are harder than ones where CPO has previously been tested. PPO-Lag has competitive performance in safety, but it converges slowly compared to the off-policy baselines. This phenomenon is even more obvious in relatively simple StaticEnv. For WCSAC-GS in the relatively more complex environments PointGoal and CarButton (Fig. 10b and c), we can see that the return and cost-return of WCSAC-GS start to stabilize near a certain value, instead of making continuous improvements until satisfying the constraint. However, in Fig. 12 (Appendix B), the safety weights of GS-1.0 and GS-0.5 quickly converge to a small value. It appears to be the case that the algorithm mistakenly takes the policy as safe, which means we get a convergent safety approximation (CVaR or expectation) that is below the safety threshold, but the safety of the policy is not truly reflected. Then, the algorithm stops making any progress in safety. Compared to the Gaussian approximation, the safety weights of IQN-1.0 and IQN-0.5 have drastic changes at the beginning of training (Fig B12b and Bc), but they finally converge to a safe policy according to the training process in Fig. 10. We hypothesize that WCSAC-IQN benefits from the quantile regression to enhance exploration and avoid overfitting. That may also explain why distributional RL can converge to a better policy than traditional RL (Dabney et al., 2018a).

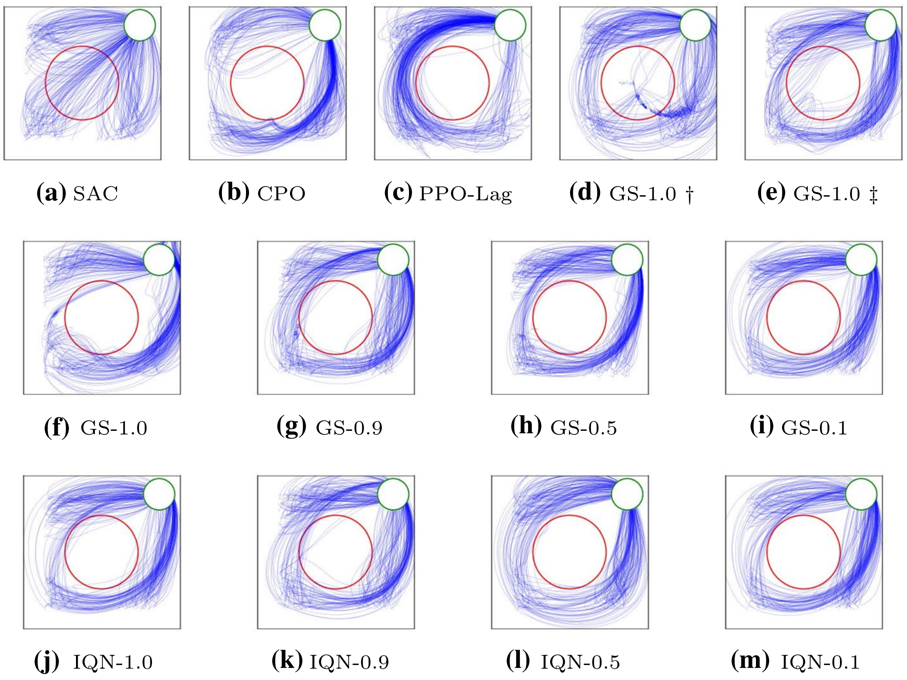


Fig. 11 Trajectory analysis. **c–e** show the trajectories generated by policies from GS-1.0 (SAC-Lag) at the beginning, middle stage and end of training respectively. **a, b**, and **f–l** show the final trajectories from SAC, CPO, PPO-Lag, and WCSAC separately

5.2.3 Trajectory analysis

Finally, we execute a trajectory analysis for each algorithm in StaticEnv, see Fig. 11. Specifically, we will compare SAC, CPO, PPO-Lag, WCSAC with different risk levels in safety, i.e., $\alpha = 0.1$ (highly risk-averse), $\alpha = 0.5$, $\alpha = 0.9$, and $\alpha = 1.0$ (risk-neutral).

The behavior of the SAC agent is presented in Fig. 11a, the agent chooses the shortest path to reach the target directly since the safety constraint is not considered.

We also consider the training process of GS-1.0 at different stages.[5] At the beginning of learning (Fig. 11d), it is possible that the agent cannot get out of the hazard, and gets stuck before arriving at the goal area. We can observe the number of constraint violations being reduced over time (Fig. 11e and f).

The final policies from CPO, PPO-Lag, GS-1.0, GS-0.9, IQN-1.0, and IQN-0.9 perform better than before, but still prefer to take a risk within the budget to get a larger return (Fig 11f, b, g, j and k). Conversely, the agents GS-0.5 and IQN-0.5 are more risk-averse (Fig 11h and l). Finally, in Fig 11i and m, we can see that the agents GS-0.1 and IQN-0.1 prefer to avoid the hazardous area more strictly given its risk level setting.

[5] Other methods show similar behavior during training.

Overall, we observe that with a higher risk level α (around 1.0), WCSAC can attain risk-neutral performance similar to expectation-based methods. Both WCSAC algorithms can become more risk-averse by setting lower risk level α.

6 Related work

Risk-averse methods have commonly been used in RL problems with a single signal (reward or cost). Although CVaR in the context of IQN was used in the work by Dabney et al. (2018a) (with only a reward signal) to get risk-sensitive policies, the implementation is significantly different from ours. We deploy IQN in the safety-constrained RL problems (with separate reward and cost signals) with continuous action spaces, instead of problems with a discrete action space.

Based on the work by Bellemare et al. (2017), Keramati et al. (2020) propose to perform strategic exploration to quickly obtain the optimal risk-averse policy. Following Dabney et al. (2018a), Urpí et al. (2021) propose a new actor critic method to optimize a risk-averse criterion in terms of return, where only samples previously collected by safe policies are available for training. Although their paper provides the off-policy algorithm version, it is not clear how the exploration-exploitation tradeoff is handled, while we explicitly define a SAC-based method with an entropy-related mechanism for exploration. Chow et al. (2017) propose efficient RL algorithms for risk-constrained MDPs, but their goal is to minimize an expected cumulative cost while keeping the cost CVaR below a given threshold, instead of maintaining reward and cost signals independently. To some extent, the way they update the Lagrange multiplier inspired our use of adaptive safety weights. However, in the real world, safe RL problems typically involve multiple objectives, some of which may be contradictory, like collision avoidance and speed on an autonomous driving task (Kamran et al., 2020). Therefore, the setting with an explicit safety signal can be more practical (Dulac-Arnold et al., 2021).

The safe RL setting with separate reward and cost signals has also been studied in several works (Achiam et al., 2017; Bharadhwaj et al., 2021; Liu et al., 2020; Yang et al., 2020). Specifically, Achiam et al. (2017); Liu et al. (2020); Yang et al. (2020) propose a series of on-policy constrained policy optimization methods with trust-region property, where the worst case performance is bounded at each update. However, they do not present a clear risk aversion mechanism for the intrinsic uncertainty, captured by the distribution over the cost-return. In addition, on-policy methods (with worse sample efficiency compared to off-policy methods) are usually not favored in safe RL domains. Under a similar problem setting, Bharadhwaj et al. (2021) work on a conservative estimate of the expected cost-return (Kumar et al., 2020) for each candidate state-action tuple, which is used in both safe-exploration and policy updates. With the conservative safety estimate, their proposed method can learn effective policies while reducing the rate of catastrophic failures. However, they only focus on the parametric uncertainty over the value estimate instead of the intrinsic uncertainty. On the other hand, their paper focuses on catastrophic events, which is a binary signal. While our paper considers safety according to the accumulated cost in a trajectory. Overall, our approach gives more freedom to the designer of the system to indicate which behaviors are more or less desirable.

Finally, we did not find state-of-the-art distributional RL techniques had been used in safety-constrained RL with separate reward and safety signals prior to our work.

7 Conclusion

In this paper, we propose an actor critic method, WCSAC , to achieve control of risks for safety-constrained RL problems. We employ a Gaussian distribution or an implicit quantile network as the safety critic to overcome considerable risks caused by the randomness in cost-return. The experiments show that both WCSAC-GS and WCSAC-IQN can attain better risk control compared to expectation-based methods. In complex environments, WCSAC-GS does not show improvements in safety, where the safety weight is not updated fast enough to truly reflect the current policy. However, WCSAC-IQN has a strong performance with the benefits from IQN, which provides a stronger safety signal than the one from the Gaussian approximation. The novel use of IQN for safety constraints can potentially be extended to other safe RL methods.

7.1 Limitations

Without any knowledge about the environment, it is hard to strictly fulfill the safety constraint during exploration. Thus, our algorithm still focus more on the performance of the final policy. While our method has good risk control for the safety-constrained RL problems, one limitation is that we cannot ensure a safe training process. Also, although our method shows good performance in practice, our work has not established theoretical proofs of convergence.

7.2 Future work

In safety-critical problems, the sample efficiency and adaptation to new tasks are both particularly crucial, so off-policy RL and meta-RL are natural approaches to solve safe-RL problems. We will further explore meta-RL with safe exploration tasks in the future (Finn et al., 2017; Rakelly et al., 2019). Another direction is to leverage the epistemic uncertainty about the safety dynamics to ensure safety also during training (Simão et al., 2021; Yang et al., 2022; Zheng & Ratliff, 2020).

Appendix A Approximate safety constraint

A discounted version of d is defined as $\overline{d} = [(1-\gamma^T)d]/[(1-\gamma)T_{\max}]$, where we assume equal cost accumulated at each step, and $T_{\max}$ is the maximum length of the episode. The assumption is not strictly correct, since we do not have equal cost at each step of a real episode, and often no costs are incurred early in the episode. However, since our algorithm optimizes the discounted infinite horizon from each state-action pair in the replay buffer, we should be approximately correct here.

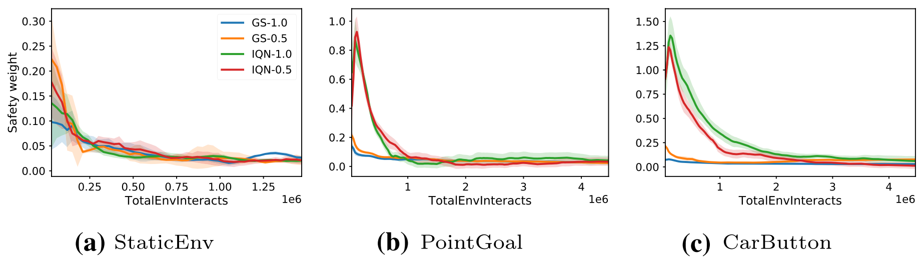


Fig. 12 Change of the adaptive safety weights in WCSAC during training

Appendix B Safety weights

We present the change of adaptive safety weights ω in WCSAC (GS-1.0, GS-0.5, IQN-1.0, and IQN-0.5) during training in Fig. 12, where a small stable weight means that the constraint is approximately satisfied. In StaticEnv (Fig. 12a), we can see that the safety weights in the four WCSAC algorithms change similarly, which accord with the convergence process of cost-return in Fig. 10a. In PointGoal and CarButton (Fig. 12b and c), compared to IQN-1.0 and IQN-0.5, the safety weights of GS-1.0 and GS-0.5 quickly converge to a small value, but they fail to obtain a constraint-satisfying policy from the results in Fig. 10b and c. Although the safety weights of IQN-1.0 and IQN-0.5 have drastic changes at the beginning of training (Fig. 12b and c), they finally converge to a safe policy, according to the training process in Fig. 10.

Author Contributions Q.Y. conceived of the presented idea, implemented the algorithms, carried out the experiments. Q.Y. and T.S. conceived and planned the experiments, contributed to the analysis of the results. S.T. proposed the spy game environment. S.T. and M.S. supervised the project. All authors discussed the results and contributed to the final manuscript, helping with writing, reviewing and editing.

Funding This research is funded by the Netherlands Organisation for Scientific Research (NWO), as part of the Energy System Integration: planning, operations, and societal embedding program and the grant NWA.1160.18.238: "PrimaVera". Qisong Yang is supported by Xidian University.

Availability of data and materials Not applicable.

Code availability Code to reproduce our experiments are available at https://github.com/AlgTUDelft/WCSAC.

Declarations

Conflict of interest/Competing interests 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.

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

References

Achiam, J., Held, D., Tamar, A., & Abbeel, P. (2017). Constrained policy optimization. *Proceedings of the 34th international conference on machine learning (pp. 22-31). PMLR.*

Altman, E. (1999). *Constrained Markov decision processes* (Vol. 7). CRC Press.

Bellemare, M. G., Dabney, W., & Munos, R. (2017). A distributional perspective on reinforcement learning. *Proceedings of the 34th international conference on machine learning* (pp. 449-458). PMLR.

Bertsekas, D. P. (1982). *Constrained optimization and Lagrange multiplier methods* (Vol. 1). Academic press.

Bharadhwaj, H., Kumar, A., Rhinehart, N., Levine, S., Shkurti, F., & Garg, A. (2021). Conservative safety critics for exploration. *9th international conference on learning representations* (pp. 1-9).

Borkar, V. S. (2005). An actor-critic algorithm for constrained Markov decision processes. *Systems & Control Letters, 54*(3), 207–213.

Chow, Y., Ghavamzadeh, M., Janson, L., & Pavone, M. (2017). Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research, 18*(1), 6070–6120.

Dabney, W., Ostrovski, G., Silver, D., & Munos, R. (2018). Implicit quantile networks for distributional reinforcement learning. *Proceedings of the 35th international conference on machine learning* (pp. 1096-1105).

Dabney, W., Rowland, M., Bellemare, M. G., & Munos, R. (2018). Distributional reinforcement learning with quantile regression. *Thirty-Second AAAI Conference on Artificial Intelligence* (pp. 2892-2901). AAAI Press.

Duan, J., Guan, Y., Li, S. E., Ren, Y., & Cheng, B. (2020). Distributional soft actor-critic: Off-policy reinforcement learning for addressing value estimation errors. arXiv preprint arxiv:2001.02811.

Dulac-Arnold, G., Levine, N., Mankowitz, D. J., Li, J., Paduraru, C., Gowal, S., & Hester, T. (2021). Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. *Machine Learning*, 2419-2468.

Finn, C., Abbeel, P., & Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. *Proceedings of the 34th international conference on machine learning* (pp. 1126-1135). PMLR.

García, J., & Fernández, F. (2015). A comprehensive survey on safe reinforcement learning. *The Journal of Machine Learning Research, 16*(1), 1437–1480.

Ha, S., Xu, P., Tan, Z., Levine, S., & Tan, J. (2020). Learning to walk in the real world with minimal human effort. arXiv preprint arxiv:2002.08550.

Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). Soft actor-Critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. Proceedings of the 35th international conference on machine learning (pp. 1861-1870). PMLR.

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., & Levine, S. (2018). Soft actor-critic algorithms and applications. arXiv preprint arxiv:1812.05905.

Huber, P. J. (1964). Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 73-101.

Kamran, D., Lopez, C. F., Lauer, M., & Stiller, C. (2020). Risk-aware highlevel decisions for automated driving at occluded intersections with reinforcement learning. *IEEE intelligent vehicles symposium, IV* (pp. 1205-1212). IEEE.

Keramati, R., Dann, C., Tamkin, A., & Brunskill, E. (2020). Being optimistic to be conservative: Quickly learning a cvar policy. *Proceedings of the AAAI conference on artificial intelligence* (pp. 4436-4443).

Khokhlov, V. (2016). Conditional value-at-risk for elliptical distributions. *Evropskỳ časopis ekonomiky a managementu, 2*(6), 70–79.

Kullback, S., & Leibler, R. A. (1951). On information and sufficiency. *The Annals of Mathematical Statistics, 22*(1), 79–86.

Kumar, A., Zhou, A., Tucker, G., & Levine, S. (2020). Conservative Q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems, 33*, 1179–1191.

Lillicrap, T., Hunt, J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Wierstra, D. (2015). Continuous control with deep reinforcement learning. *4th international conference on learning representations* (pp. 1-10). ICLR.

Liu, Y., Ding, J., & Liu, X. (2020). IPO: Interior-point policy optimization under constraints. *Proceedings of the AAAI conference on artificial intelligence* (pp. 4940-4947).

Ma, X., Zhang, Q., Xia, L., Zhou, Z., Yang, J., & Zhao, Q. (2020). Distributional soft actor critic for risk sensitive learning. arXiv preprint arxiv:2004.14547.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature, 518*(7540), 529–533.

Morimura, T., Sugiyama, M., Kashima, H., Hachiya, H., & Tanaka, T. (2010). Parametric return density estimation for reinforcement learning. *Twenty-sixth conference on uncertainty in artificial intelligence* (pp. 368-375). AUAI Press.

Olkin, I., & Pukelsheim, F. (1982). The distance between two random vectors with given dispersion matrices. *Linear Algebra and its Applications, 48*, 257–263.

Pecka, M., & Svoboda, T. (2014). Safe exploration techniques for reinforcement learning–an overview. *First international workshop on modelling and simulation for autonomous systems* (pp. 357-375). Springer.

Rakelly, K., Zhou, A., Finn, C., Levine, S., & Quillen, D. (2019). Efficient off-policy meta-reinforcement learning via probabilistic context variables. *Proceedings of the 36th international conference on machine learning* (Vol. 97, pp. 5331-5340). PMLR.

Ray, A., Achiam, J., & Amodei, D. (2019). Benchmarking safe exploration in deep reinforcement learning. Retrieved from https://cdn.openai.com/safexp-short.pdf

Rockafellar, R. T., & Uryasev, S. (2000). Optimization of conditional value-at-risk. *Journal of Risk, 2*(3), 21–41.

Rowland, M., Dadashi, R., Kumar, S., Munos, R., Bellemare, M. G., & Dabney, W. (2019). Statistics and samples in distributional reinforcement learning. *Proceedings of the 36th international conference on machine learning* (pp. 5528-5536).

Roy, J., Girgis, R., Romoff, J., Bacon, P.-L., & Pal, C. (2021). Direct behavior specification via constrained reinforcement learning. arXiv preprint arxiv:2112.12228.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., & Moritz, P. (2015). Trust region policy optimization. *Proceedings of the 32nd international conference on machine learning* (pp. 1889-1897). JMLR.org.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal Policy optimization algorithms. arXiv preprint arxiv:1707.06347.

Simão, T. D., Jansen, N., & Spaan, M. T. J. (2021). AlwaysSafe: Reinforcement learning without safety constraint violations during training. *Proceedings of the 20th international conference on autonomous agents and multiagent systems* (AAMAS) (pp. 1226-1235). IFAAMAS.

Sobel, M. J. (1982). The variance of discounted markov decision processes. *Journal of Applied Probability, 19*(4), 794–802.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction* (Vol. 2). MIT press.

Tamar, A., Di Castro, D., & Mannor, S. (2016). Learning the variance of the reward-To-Go. *The Journal of Machine Learning Research, 17*(1), 361–396.

Tang, Y. C., Zhang, J., & Salakhutdinov, R. (2020). Worst cases policy gradients. *3rd annual conference on robot learning* (pp. 1078-1093). PMLR.

Théate, T., Wehenkel, A., Bolland, A., Louppe, G., & Ernst, D. (2021). Distributional reinforcement learning with unconstrained monotonic neural networks. arXiv preprint arxiv:2106.03228.

Urpí, N. A., Curi, S., & A. K. (2021). Risk-averse offline reinforcement learning. *9th international conference on learning representations*.

Yang, T.-Y., Rosca, J., Narasimhan, K., & Ramadge, P. J. (2020). Projection-based constrained policy optimization. *8th international conference on learning representations*.

Yang, Q., Simão, T. D., Jansen, N., Tindemans, S. H., & Spaan, M. T. J. (2022). Training and transferring safe policies in reinforcement learning. *AAMAS 2022 Workshop on Adaptive Learning Agents*.

Yang, Q., Simão, T. D., Tindemans, S. H., & Spaan, M. T. J. (2021). WCSAC: Worst-case soft actor critic for safety-constrained reinforcement learning. *Thirty-Fifth AAAI conference on artificial intelligence* (pp. 10639–10646). AAAI Press.

Yang, D., Zhao, L., Lin, Z., Qin, T., Bian, J., & Liu, T.-Y. (2019). Fully parameterized quantile function for distributional reinforcement learning. *Advances in Neural Information Processing Systems* 32 (pp. 6193-6202). Curran Associates, Inc.

Zheng, L., & Ratliff, L. (2020). Constrained upper confidence reinforcement learning. *Proceedings of the 2nd conference on learning for dynamics and control* (pp. 620-629). online: PMLR.

Authors and Affiliations

Qisong Yang[1,2] · Thiago D. Simão[1,3] · Simon H. Tindemans[2] · Matthijs T. J. Spaan[1]

Thiago D. Simão
thiago.simao@ru.nl

Simon H. Tindemans
s.h.tindemans@tudelft.nl

Matthijs T. J. Spaan
m.t.j.spaan@tudelft.nl

[1] Software Technology, Delft University of Technology, Van Mourik Broekmanweg, 2628 XE Delft, The Netherlands

[2] Electrical Sustainable Energy, Delft University of Technology, Mekelweg, 2628 CD Delft, The Netherlands

[3] Software Science, Radboud University, Toernooiveld, 6525 EC Nijmegen, The Netherlands