Artificial Intelligence 311 (2022) 103765

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Risk-averse policy optimization via risk-neutral policy optimization [☆]

Lorenzo Bisi [a,*], Davide Santambrogio [a,1], Federico Sandrelli [a,1], Andrea Tirinzoni [a,2], Brian D. Ziebart [b], Marcello Restelli [a]

[a] *Politecnico di Milano, Milan, Italy*
[b] *University of Illinois, Chicago, USA*

ARTICLE INFO

Article history:
Received 16 June 2021
Received in revised form 13 March 2022
Accepted 8 July 2022
Available online 14 July 2022

Keywords:
Reinforcement learning
Risk-aversion
Risk-sensitivity

ABSTRACT

Keeping risk under control is a primary objective in many critical real-world domains, including finance and healthcare. The literature on risk-averse reinforcement learning (RL) has mostly focused on designing ad-hoc algorithms for specific risk measures. As such, most of these algorithms do not easily generalize to measures other than the one they are designed for. Furthermore, it is often unclear whether state-of-the-art risk-neutral RL algorithms can be extended to reduce risk. In this paper, we take a step towards overcoming these limitations, proposing a single framework to optimize some of the most popular risk measures, including conditional value-at-risk, utility functions, and mean-variance. Leveraging recent theoretical results on state augmentation, we transform the decision-making process so that optimizing the chosen risk measure in the original environment is equivalent to optimizing the expected cost in the transformed one. We then present a simple risk-sensitive meta-algorithm that transforms the trajectories it collects from the environment and feeds these into any risk-neutral policy optimization method. Finally, we provide extensive experiments that show the benefits of our approach over existing ad-hoc methodologies in different domains, including the Mujoco robotic suite and a real-world trading dataset.

1. Introduction

Recent advances have enabled Reinforcement Learning (RL) [58] to achieve milestone results in many complex problems, including playing games [36,56,8] and controlling robotic systems [28,29,24]. A wide variety of powerful policy optimization algorithms are now available in the literature, such as TRPO [52], PPO [54], DDPG [30], and SAC [23], to name a few. These approaches are designed to efficiently minimize the expected total cost (or, equivalently, maximize the expected total reward). However, in many real-world domains, and particularly in critical applications such as finance or healthcare, a primary objective, besides minimizing the expected costs, is to keep *risk* under control [33,38,34,32]. There are many ways

[☆] This article belongs to Special Issue: Risk-aware Autonomous Systems: Theory and Practice.
[*] Corresponding author.
E-mail address: lorenzo.bisi@polimi.it (L. Bisi).
[1] Participated to this work during a double degree program between University of Illinois and Politecnico di Milano.
[2] Now at INRIA.

https://doi.org/10.1016/j.artint.2022.103765
0004-3702/© 2022 Elsevier B.V. All rights reserved.

to express this notion of risk, primarily split into two categories [20]: *model uncertainty* and *inherent uncertainty*. The former arises from the imperfect knowledge of the underlying parameters of the environment and is typically addressed by using robust [46,41,65,69,31] or safe [21,20] RL techniques. The latter, which will be the primary focus of this paper, is related to the intrinsic stochasticity of the environment and is addressed by optimizing specific *risk measures*, i.e., different objective functions than the classic expected cost.

Several risk criteria have been taken into consideration in the RL literature, including utility functions [37,55,40], conditional value-at-risk [61,13–15], variance-related objectives [18,48,59], and the general class of coherent measures [60]. Since these risk criteria possess very different properties, a common approach consists in developing ad-hoc RL algorithms to optimize each risk measure (or class of risk measures), i.e., algorithms that are highly-specialized to the chosen objective function. While this enables a full understanding of the problem at hand, in practice it can be limited for at least two reasons: (1) Given a risk-averse RL algorithm for a specific risk measure, it is often not clear whether the algorithm can be easily adapted to optimize a different measure; and (2) Given any state-of-the-art (risk-neutral) RL algorithm, it is often non-trivial to adapt the algorithm to optimize some desired risk measure instead of the expected return. Intuitively, overcoming these two limitations is highly desirable from a practical viewpoint. Ideally, we would like an algorithm that enables optimization of a multitude of risk measures in an almost transparent manner and which, at the same time, can leverage recent advances in risk-neutral RL to improve learning efficiency.

In this paper, we take a step forward in this direction by proposing a single framework to optimize some of the most popular risk measures, including conditional value-at-risk, entropic risk measure, and mean-variance, by adopting *any* risk-neutral RL algorithm. Instead of focusing on deriving algorithms for optimizing each single risk measure, we transform the underlying Markov decision process (MDP) so that optimizing the chosen risk measure in the original MDP is equivalent to optimizing the expected cost in the transformed one. We achieve this by leveraging previous theoretical results on state augmentation [4,5], which we use to unify the optimization problem for the considered measures. The price we have to pay for this generality is the addition of one or two extra state variables and one extra optimization variable to the original problem, which we show can be easily handled in practice. Overall, our framework enables practitioners to learn risk-averse policies with minimal additional effort beyond learning risk-neutral ones. We believe this to be a significant step towards applying risk-sensitive RL algorithms to complex real-world problems.

Our detailed contributions are as follows.

1. Using recent results on state augmentation [5], we derive a unified objective for the considered risk measures (Section 3). In addition to reducing the conditional-value-at-risk and the entropic risk measure to an ordinary MDP, as originally shown by Bäuerle and Rieder [5], we also show that mean-variance can be treated analogously, by exploiting the decomposition provided in Xie et al. [68].
2. We propose a very simple meta-algorithm, Risk-averse policy Optimization by State Augmentation (ROSA), to optimize the unified objective by exploiting any available risk-neutral RL algorithm (Section 4).
3. We propose extensive empirical results that demonstrate:
   (i) the benefits of our single meta-algorithm over existing ad-hoc methodologies
   (ii) the scalability of our approach
   (iii) its performance on a real-world trading dataset (Section 6).

2. Preliminaries

A discounted Markov decision process (MDP) [49] is a tuple $M = (\mathcal{S}, \mathcal{A}, p, c, \mu, \gamma)$, where $\mathcal{S}$ is a measurable state space, $\mathcal{A}$ is a measurable action space, $p : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition kernel, $c : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is a measurable cost function, $\mu : \mathcal{S} \to \Delta(\mathcal{S})$ is the initial-state distribution, and $\gamma \in [0, 1)$ is the discount factor. Here $\Delta(\mathcal{S})$ denotes the set of probability measures over $\mathcal{S}$.

Without loss of generality,[3] we shall suppose that costs are positive and bounded by $c_{\max} > 0$ almost surely. The decision process works as follows. First, the initial state S_0 is drawn from μ. Then, the agent takes an action A_0, it transitions to a new state $S_1 \sim p(\cdot|S_0, A_0)$, it receives the cost $C_1 = c(S_0, A_0, S_1)$, and so on. Let $H_t := (S_0, A_0, S_1, C_1, \dots, S_t)$ denote a t-step history and $\mathcal{H}_t$ the set of such histories. The agent's actions are governed by a possibly history-dependent policy $\pi = \{\pi_t\}_{t\geq 0}$, i.e., a sequence of mappings $\pi_t : \mathcal{H}_t \to \Delta(\mathcal{A})$. We denote by Π^{HR} the set of history-dependent policies and by $\Pi^{\text{MD}} \subset \Pi^{\text{HR}}$ the set of Markovian deterministic policies, which are mappings $\pi : \mathcal{S} \to \mathcal{A}$. Each policy $\pi \in \Pi^{\text{HR}}$ induces a sequence of states and costs $\{(S_t^\pi, C_{t+1}^\pi)\}_{t\geq 0}$ and we denote by $\mathbb{P}^\pi, \mathbb{E}^\pi$ the resulting probability measure and expectation operator, respectively. We suppose there exists a subset $\bar{\mathcal{S}} \subset \mathcal{S}$ of absorbing (or terminal) states, such that, for all $s, s' \in \bar{\mathcal{S}}$ and $a \in \mathcal{A}$, $p(\bar{\mathcal{S}}|s, a) = 1$ and $c(s, a, s') = 0$. Let't define the random variable $T_\pi := \inf\{t \in \mathbb{N}^+ : S_t^\pi \in \bar{\mathcal{S}}\}$ to be the hitting time of an absorbing state when executing policy π. We need the following assumption.

Assumption 1 *(Bounded hitting times).* There exists $T < \infty$ such that, for any $\pi \in \Pi^{\text{HR}}$, $T_\pi \leq T$ almost surely.

[3] Any MDP can be reduced to this setting by translating the cost function and properly modifying the cost of the terminal absorbing state.

This assumption is common, e.g., in the policy gradient literature [45,17], where the trajectories collected by the agent terminate almost surely, no matter what policy is executed. Finally, the total (discounted) cost is denoted by $G := \sum_{t=0}^{\infty} \gamma^t c(S_t, A_t, S_{t+1}) = \sum_{t=0}^{T-1} \gamma^t c(S_t, A_t, S_{t+1})$, where the second equality follows from Assumption 1.

2.1. Risk measures

Let $(\mathcal{X}, \mathcal{F})$ be a measurable space. A risk measure over $\mathcal{X}$ is a function $\rho : \mathcal{X} \to \mathbb{R}$ that maps uncertain outcomes $X \in \mathcal{X}$ to the real line. In this paper, we are interested in optimizing a risk measure of the total discounted cost:

$$\min_{\pi \in \Pi^{\mathrm{HR}}} \rho^{\pi}(G), \tag{1}$$

where we add a dependency on π to emphasize that the underlying probability measure is induced by the chosen policy. The simplest example is $\rho^{\pi} = \mathbb{E}^{\pi}$, for which (1) reduces to the standard risk-neutral RL problem. We now introduce the risk measures considered in this work.

Conditional value-at-risk The Value-at-Risk (VaR) at level $\alpha \in (0, 1)$ of G is $\rho^{\pi}_{\mathrm{VaR}}(G; \alpha) := \inf\{x : \mathbb{P}^{\pi}\{G \leq x\} \geq \alpha\}$. VaR is a popular risk measure, e.g., in finance, but it is often unstable and it does not handle losses suffered beyond the threshold amount it indicates. The Conditional Value-at-Risk (CVaR) has been defined in Rockafellar et al. [50] as:

$$\rho^{\pi}_{\mathrm{CVaR}}(G; \alpha) = \min_{\eta \in \mathbb{R}} \{\eta + \frac{1}{1-\alpha} \mathbb{E}^{\pi}\left[(G - \eta)^{+}\right]\},$$

and it has several advantages over VaR: it allows to quantify the losses encountered in the tail, it can be expressed as a minimization, and it is a coherent risk measure [2].

Utility functions and entropic risk measure A popular approach to incorporate risk sensitivity into objective functions makes use of utility theory [22]. For a convex function $u : \mathbb{R} \to \mathbb{R}$, the utility-based risk measure is $\rho^{\pi}_{\mathrm{U}}(G; u) := \mathbb{E}^{\pi}[u(G)]$. Among them, exponential utility functions have been the first type of risk-aversion employed in MDPs [26]. Let $\beta > 0$ be the level of risk aversion. The entropic risk measure (ERM) of G is $\rho^{\pi}_{\mathrm{ERM}}(G; \beta) := \frac{1}{\beta} \log \mathbb{E}^{\pi}\left[e^{\beta G}\right]$. Optimizing ERM is equivalent to optimizing an exponential utility function (EU), $\rho^{\pi}_{\mathrm{EU}}(G; \beta) := \mathbb{E}^{\pi}\left[e^{\beta G}\right]$.

Mean-variance For $\lambda \geq 0$, the mean-variance (MV) of G is $\rho^{\pi}_{\mathrm{MV}}(G; \lambda) := \mathbb{E}^{\pi}[G] + \lambda \mathbb{V}\mathrm{ar}^{\pi}[G]$. Optimizing MV trades off between minimizing the expected total cost and its variance, where λ is a tunable parameter controlling the level of risk aversion. We note that a common alternative formulation is to minimize the expected cost subject to the variance not exceeding a given threshold. The methods we shall propose generalize straightforwardly to this constrained problem by optimizing its Lagrangian.

Mean-volatility The mean-volatility (MVo) risk measure [9] is an alternative to MV where the variance is taken w.r.t. the single-step cost under the discounted state-occupancy measure induced by π. For $\lambda \geq 0$, the MVo is $\rho^{\pi}_{\mathrm{MVo}}(G; \lambda) := \mathbb{E}^{\pi}[G] + \lambda \nu^{\pi}[G]$, where $\nu^{\pi}[G] := (1-\gamma)\mathbb{E}^{\pi}\left[\sum_t \gamma^t \left(C_{t+1} - \mathbb{E}^{\pi}[G]\right)^2\right]$ is called reward-volatility.

3. A unified perspective

Our first step is to reduce the main optimization problem (1) to a single objective that unifies all the risk measures introduced in Section 2.1. Later on, we shall see how to design a common framework that optimizes it by leveraging any risk-neutral RL algorithm as a sub-routine.

Definition 1 *(Unified objective).* Let ρ be a risk measure and $f_{\rho} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, $g_{\rho} : \mathbb{R} \to \mathbb{R}$ be two functions. The unified optimization problem is:

$$\min_{\eta \in \mathbb{R}} \left\{ \min_{\pi \in \Pi^{\mathrm{HR}}} \mathbb{E}^{\pi}\left[f_{\rho}(G, \eta)\right] + g_{\rho}(\eta) \right\}. \tag{2}$$

The explicit definition of the quantities involved depend on the chosen risk measure and its parameters, as specified in the following proposition.

Proposition 1. *For any of the risk measures of Section 2.1, an optimal policy computed by solving (1) is also optimal for (2) and vice versa, where*

- *for CVaR at level α, $f_{\text{CVaR}}(G,\eta)=\frac{1}{1-\alpha}(G-\eta)^+$ and $g_{\text{CVaR}}(\eta)=\eta$ [50];*
- *for a utility function u, we have no external parameter η, $f_{\text{U}}(G)=u(G)$, and $g=0$. In particular, for ERM with parameter β, $f_{\text{ERM}}(G)=e^{\beta G}$ and $g=0$;*
- *for MV with parameter λ, $f_{\text{MV}}(G,\eta)=(1-2\eta\lambda)G+\lambda G^2$ and $g_{\text{MV}}(\eta)=\lambda\eta^2$. MVo is analogous with the one-step cost instead of G and the expectation under the state-action occupancy measure.*

Proof. From the definition

$$\rho^{\pi}_{\text{CVaR}}(G;\alpha)$$

we have:

$$\begin{aligned}\min_{\pi\in\Pi^{\text{HR}}}\rho^{\pi}_{\text{CVaR}}(G;\alpha)&=\min_{\pi\in\Pi^{\text{HR}}}\min_{\eta\in\mathbb{R}}\left\{\eta+\frac{1}{1-\alpha}\mathbb{E}^{\pi}\left[(G-\eta)^+\right]\right\}\\&=\min_{\eta\in\mathbb{R}}\left\{\eta+\min_{\pi\in\Pi^{\text{HR}}}\mathbb{E}^{\pi}\left[\frac{1}{1-\alpha}(G-\eta)^+\right]\right\}.\end{aligned}$$

In the case of utility functions, the objective in (1) is actually equivalent to the one in (2) with no outer variable η. For ERM, the result simply follows by noting that the problem is equivalent to optimizing the exponential utility, which in fact does not require any extra variable. For MV with parameter λ, we use the same trick as in Xie et al. [68]. Starting from $\mathbb{V}\text{ar}^{\pi}[G]=\mathbb{E}^{\pi}\left[G^2\right]-\mathbb{E}^{\pi}[G]^2$, we use Legendre-Fenchel duality to reduce the squared expectation to a standard expectation, $\mathbb{E}^{\pi}[G]^2=-\min_{\eta\in\mathbb{R}}\left\{\eta^2-2\eta\mathbb{E}^{\pi}[G]\right\}$. Thus,

$$\begin{aligned}\min_{\pi\in\Pi^{\text{HR}}}\rho^{\pi}_{\text{MV}}(G;\alpha)&=\min_{\pi\in\Pi^{\text{HR}}}\left\{\mathbb{E}^{\pi}[G]+\lambda\left(\mathbb{E}^{\pi}\left[G^2\right]-\mathbb{E}^{\pi}[G]^2\right)\right\}\\&=\min_{\pi\in\Pi^{\text{HR}}}\left\{\mathbb{E}^{\pi}[G]+\lambda\mathbb{E}^{\pi}\left[G^2\right]+\lambda\min_{\eta\in\mathbb{R}}\left\{\eta^2-2\eta\mathbb{E}^{\pi}[G]\right\}\right\}\\&=\min_{\eta\in\mathbb{R}}\left\{\min_{\pi\in\Pi^{\text{HR}}}\mathbb{E}^{\pi}\left[(1-2\eta\lambda)G+\lambda G^2\right]+\lambda\eta^2\right\}.\quad\square\end{aligned}$$

The unified objective (2), together with Proposition 1, reveals that computing the optimal risk-averse policy in (1) can be reduced to computing the policy minimizing the *expected value* of some (non-linear) function of the total cost. One of the consequences due to moving from the risk operator to the expectation one is the introduction of a single additional optimization variable η. We now discuss how to optimize (2) by considering the two variables, π and η, separately. Specifically, in Section 3.1, we show that optimizing for π when η is fixed (inner objective) can be reduced to an ordinary MDP and thus solved by any RL algorithm. In Section 3.2, we show that the optimal value of η given π (outer objective) can be conveniently estimated for all the considered risk measures. Hence, a natural approach to solve (2) is an alternating optimization method, also known as block coordinate descent [67]. We present such an approach for our policy optimization framework in Section 4.

3.1. Inner objective as an ordinary MDP

Fix some value $\eta\in\mathbb{R}$ for the outer variables. The inner problem in (2) seeks a policy $\pi\in\Pi^{\text{HR}}$ that minimizes $\mathbb{E}^{\pi}\left[f_{\rho}(G,\eta)\right]$. Computing its solution is non-trivial for at least two reasons. First, when f_{ρ} is a non-linear function of the total cost, as for our risk measures, we may lose some guarantees about the optimal solution. In fact, provided that such optimal solution exists, Markovian deterministic policies may not be sufficient anymore to ensure optimality [49] and we need to look for history-dependent ones. Second, the optimization depends on the specific choice of f_{ρ}, i.e., on the underlying risk measure ρ. Instead of designing ad-hoc methodologies, we address these two complications by reducing the inner objective to an ordinary MDP via state-space augmentation. This will enable the application of any RL algorithm to its optimization. To achieve this, we borrow the state-augmentation proposed by Bäuerle and Rieder [5]. The key intuition is that making an optimal decision at each time only requires keeping track of the cumulative discounted cost suffered so far, instead of the whole sequence of states and actions. Formally, we define the augmented MDP as follows.

Algorithm 1 Trajectory-based State Augmentation.

```
Require: Trajectories $\{\tau_1, \tau_2, \dots, \tau_n\}$
    where $\tau_i = (S_{0,i}, A_{0,i}, S_{1,i}, C_{1,i} \dots, S_{T_i,i}, C_{T_i,i})$,
    function $f_\rho : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, parameter $\eta$
Ensure: Augmented trajectories $\{\tilde{\tau}_1, \tilde{\tau}_2, \dots, \tilde{\tau}_n\}$
1: for $i = 1, 2, \dots, n$ do
2:     for $t = 0, 1, \dots, T_i$ do
3:         $\tilde{S}_{t,i} \leftarrow (S_{t,i}, \sum_{h=0}^{t} \gamma^h C_{h,i+1}, \gamma^t)$
4:     end for
5:     $\tilde{C}_{T_i,i} \leftarrow f_\rho(\sum_{t=0}^{T_i-1} \gamma^t C_{t,i+1}, \eta)$
6:     $\tilde{\tau}_i = (\tilde{S}_{0,i}, A_{0,i}, \tilde{S}_{1,i}, 0 \dots, \tilde{S}_{T_i,i}, \tilde{C}_{T_i,i})$
7: end for
```

Definition 2 *(Augmented MDP [5]).* Let $M = (\mathcal{S}, \mathcal{A}, p, c, \mu, \gamma)$ be the original MDP. The augmented MDP for optimizing $\mathbb{E}^\pi \left[f_\rho(G, \eta)\right]$ is $\tilde{M} = (\tilde{\mathcal{S}}, \mathcal{A}, \tilde{p}, \tilde{c}, \tilde{\mu}, \tilde{\gamma})$, where:

- $\tilde{\mathcal{S}} := \mathcal{S} \times [0, c_{\max}/(1-\gamma)] \times (0, 1]$;
- for $\tilde{s} = (s, v, w)$, $a \in \mathcal{A}$, and $\tilde{s}' = (s', v', w')$, the transition kernel $\tilde{p}$ is such that[4] $\tilde{p}(\tilde{s}'|\tilde{s}, a) = p(s'|s, a)\delta\big(\gamma w - w'\big)\delta\big(v + wc(s, a, s') - v'\big)$;
- the cost function is $\tilde{c}(\tilde{s}, a, \tilde{s}') = f_\rho(v', \eta)$ if $s' \in \bar{\mathcal{S}}$, the terminal states set, and $s \notin \bar{\mathcal{S}}$, zero otherwise;
- the initial state-distribution is $\tilde{\mu}(\tilde{s}) = \mu(s)\delta(v)\,\delta(w-1)$
- and the discount factor is $\tilde{\gamma} = 1$.

Intuitively, the state-space is augmented with two scalar variables, while the action-space remains unchanged. We denote each augmented state by a tuple $\tilde{s} = (s, v, w)$, where s is the original state of our system, v keeps track of the running cumulative cost, and w keeps track of the discounting.[5] The transition kernel is such that the first state variable evolves according to the original transition dynamics, while the remaining two evolve deterministically. If $(\tilde{s}_0, a_0, \dots, \tilde{s}_T)$ is a trajectory in the augmented MDP, we have $w_{t+1} = \gamma w_t$ and $v_{t+1} = v_t + w_t c(s_t, a_t, s_{t+1})$. Since $v_0 = 0$ and $w_0 = 1$, unrolling this dynamics, it is easy to see that $w_{t+1} = \gamma^t$ and $v_{t+1} = \sum_{h=0}^{t} \gamma^h c(s_h, a_h, s_{h+1})$, i.e., the two extra state variables have the intended meaning. If a transition to an absorbing state of the original MDP occurs at time t, $\tilde{c}(\tilde{s}_t, a_t, \tilde{s}'_{t+1}) = f_\rho(v_{t+1}, \eta)$. Since the cost function is zero everywhere except when entering an absorbing state and $\tilde{\gamma} = 1$, $\sum_{t=0}^{T-1} \tilde{\gamma}^t \tilde{c}(\tilde{S}_t, A_t, \tilde{S}_{t+1}) = f_\rho\left(\sum_{t=0}^{T-1} \gamma^t c(S_t, A_t, S_{t+1}), \eta\right) = f_\rho(G, \eta)$, that is, the cumulative cost of the augmented trajectories is the same as the application of f_ρ to the cumulative cost of the original ones. This implies that we can solve the augmented MDP as an ordinary one, in which we seek a policy $\tilde{\pi} : \tilde{\mathcal{S}} \to \mathcal{A}$ that minimizes $\mathbb{E}^{\tilde{\pi}}\left[\sum_{t=0}^{T-1} \tilde{\gamma}^t \tilde{c}(\tilde{S}_t, A_t, \tilde{S}_{t+1})\right]$. Provided that standard compactness and continuity assumptions hold [5], an optimal Markov deterministic policy exists for this (augmented) problem. Furthermore, there is always a corresponding non-Markovian policy that is (at least locally) optimal for the original (non-augmented) problem. Here Markov refers to the fact that $\tilde{\pi}$ directly maps augmented states to actions, though these states depend on the history of the original process. The key result is, thus, that we can solve the inner objective in (2) by first augmenting the state space and then applying *any* (risk-neutral) RL algorithm. Clearly, we cannot directly build the augmented MDP as in Bäuerle and Rieder [5] since the dynamics are unknown. However, we note that it is simple to perform this augmentation on given samples, such as trajectories collected by the chosen RL method. In fact, all we have to do is keep track of the running costs and discounting and progressively add them to the original state samples. This procedure is summarized in Algorithm 1.

3.2. Optimizing the outer objective

Now that we have a convenient way to solve the inner objective in (2) for any fixed η, it only remains to specify how to compute the optimal value of η for any fixed policy. We now see that this is actually simple and can be done in closed-form for all the risk measures that we consider, or, if the model is not available, it can be efficiently estimated from samples. Formally, let $\pi \in \Pi^{\mathrm{HR}}$ be any policy and $\eta^\star(\pi) := \operatorname{argmin}_{\eta \in \mathbb{R}} \left\{\mathbb{E}^\pi\left[f_\rho(G, \eta)\right] + g_\rho(\eta)\right\}$. Starting from the definitions of the functions f_ρ, g_ρ for the various risk measures, we have what follows.

- For CVaR:

$$\eta^\star_{\mathrm{CVaR}}(\pi) = \operatorname*{argmin}_{\eta \in \mathbb{R}} \left\{\frac{1}{1-\alpha}\mathbb{E}^\pi\left[(G-\eta)^+\right] + \eta\right\}.$$

[4] Here $\delta(x)$ denotes the Dirac delta function at x.

[5] Keeping track of the discounting can be avoided, but the augmented MDP would have a non-stationary transition kernel. Clearly, when $\gamma = 1$ this extra state variable can be neglected.

This was shown by Rockafellar et al. [50] to be exactly the value-at-risk at level α, i.e., $\eta^\star_{\text{CVaR}}(\pi) = \rho^\pi_{\text{VaR}}(G;\alpha)$.

- For mean-variance, we have:

$$\eta^\star_{\text{MV}}(\pi) = \operatorname*{argmin}_{\eta\in\mathbb{R}} \left\{ \mathbb{E}^\pi\left[(1-2\eta\lambda)G + \lambda G^2\right] + \lambda\eta^2 \right\}.$$

This is a simple quadratic function of η. Taking its derivative and equating it to zero, we obtain $\eta^\star_{\text{MV}}(\pi) = \mathbb{E}^\pi[G]$, i.e., $\eta^\star_{\text{MV}}(\pi)$ is the expected total cost of π. It is easy to see that the same holds for the mean-volatility, $\eta^\star_{\text{MVo}}(\pi) = \mathbb{E}^\pi[G]$.

- Finally, the ERM has no outer parameter and thus it can be optimized by solving exclusively the inner objective via state-augmentation.

4. Policy optimization

We now present our general approach to risk-averse RL. Our meta-algorithm, called ROSA (Risk-averse policy Optimization by State Augmentation), is shown in Algorithm 2. ROSA takes as input a risk measure ρ among those of Section 2.1 and a risk-neutral RL algorithm A (hence the name meta-algorithm). No requirement on A is imposed and, in principle, it can be any RL algorithm. We shall elaborate more on its choice later on. Before learning starts, ROSA casts ρ into the unified objective (2) by finding the corresponding functions f_ρ and g_ρ. At each iteration $j = 1,\dots,k$, ROSA collects a batch of n trajectories using the current policy π_j. Then, it estimates the next value of η_j by using the closed-form expression as mentioned in Section 3.2. Since this involves the expected value of $f(G;\eta_j)$ under π_j, an unbiased estimator is built using the collected trajectories. Finally, ROSA augments the trajectories by using Algorithm 1 and feeds them into the policy optimization algorithm A. The latter performs one or more updates to the current policy. The overall procedure is an incremental block coordinate descent method [67], whose convergence to a local optimum has been proven for general convex [7] and non-convex [64] settings.

Algorithm 2 Risk-averse policy Optimization by State Augmentation (ROSA).

```
Require: Risk measure $\rho$, risk-neutral RL algorithm A (e.g., PPO, TRPO, etc.), batch size $n$, number of iterations $k$
1: Compute functions $f_\rho$ and $g_\rho$ as in Proposition 1
2: Initialize policy $\pi_1$
3: for $j = 1, 2, \dots, k$ do
4:     Collect a batch $\{\tau_i\}_{i=1}^n$ of $n$ trajectories using $\pi_j$
5:     Compute $\eta_j \leftarrow \operatorname{argmin}_{\eta\in\mathbb{R}} h_j(\eta)$ where
       $h_j(\eta) := \frac{1}{n}\sum_{i=1}^n f_\rho\left(\sum_{t=0}^{T_i-1}\gamma^t C_{t+1,i}, \eta\right) + g_\rho(\eta)$
6:     Get augmented trajectories $\{\tilde{\tau}_i\}_{i=1}^n$ with Algorithm 1
7:     Feed $\{\tilde{\tau}_i\}_{i=1}^n$ into A, optimize $\pi_j$ and obtain $\pi_{j+1}$
8: end for
```

The possibility to adopt any risk-neutral RL algorithm A to optimize a risk measure is the key component of ROSA. Such algorithm can be freely chosen among those available in the literature. For instance, it can be an on-policy [51, 54] or off-policy [23,39] policy search algorithm or even a value-based method [36]. In our experiments, we shall indeed combine ROSA with both on-policy and off-policy methods. Regardless of the chosen algorithm A, ROSA interacts with the environment in an online on-policy fashion, collecting, at each step, a batch of trajectories under the current policy and updating the latter by means of A. This is required to compute the outer variables, whose update requires the estimation of some statistics of the current policy (e.g., the expected return). While this is the solution that we consider in this paper, we note that it is not restrictive and ROSA can be generalized to a fully off-policy setting by employing, e.g., importance sampling [43].

Additional component A possible concern in using the proposed approach regards the state augmentation's negative impact to the underlying RL problem. While it is true that adding state variables might increase the sample complexity, we note that this augmentation has been shown as a sufficient condition for representing optimal *Markov* policies for CVaR and concave utilities criteria [4,5]. On the other hand, existing ad-hoc approaches typically consider only Markov policies in the original state space and, thus, while solving simpler problems, might not necessarily converge to near-optimal risk-averse behavior.

Sparse costs The augmented MDP features a modified cost function, which is zero valued in all the steps apart from the last one. This sparse cost function may make the problem harder from a credit assignment viewpoint. Moreover, it can make more challenging the use of importance sampling techniques, whose variance grows exponentially with the time at which rewards occur. In tasks in which these issue are more severe, one could prefer, for instance, on-policy methods employing eligibility traces [57], to better deal with credit assignment.

Sensitivity to the outer variable estimation While the outer variable η can always be computed in closed form knowing the model, when the MDP is unknown we need to estimate it from samples. This estimation is easier in the MV case, where η is the expected return, while it is more challenging in the CVaR case, where a quantile estimate needs to be computed instead. The errors in these estimates may influence the optimization path, by altering the sequence of the augmented problems to be solved. However, convergence to some local optimum of the original CVaR objective is not affected from these errors, and policy initialization is more likely to have a critical role in determining which local optimum will be obtained at the end.

Non-stationary costs We note that, when using alternated incremental updates (in a block-coordinate descent fashion) as in ROSA, the reward function optimized by the risk-neutral RL algorithm becomes non-stationary. This is due to the fact that the reward at each iteration depends on η through f, and the value of η is repeatedly updated by the outer optimizer. Such non-stationarity could become problematic for value-based methods, where a moving target may cause instability. In order to avoid non-stationary costs, it is possible to explicitly inform the agent of the new context, by including the outer variable η as part of the state or, more simply, as an input to the policy and/or value function. In this way costs become stationary, at the price of introducing some non-stationarity into the initial state distribution. However, in a function approximation regime, this approach could allow to generalize what the agent has learnt for previous value of η to new ones. More refined techniques can be adopted, by taking inspiration from non-stationary [44], multi-task [3,12] and lifelong learning literature [1]. Luckily this is not an issue for policy gradient algorithms, whose convergence could still be guaranteed using analyses from the block-coordinate literature.

To conclude, we summarized the main challenges that the ROSA framework poses, together with some possible solutions. We argue that, while the framework allows the employment of any RL algorithm, state-of-the-art trust-region approaches as TRPO [51] and PPO [54] are likely to be exempt from most of the aforementioned issues, since they are on-policy policy gradient approaches, naturally including mechanisms to deal with credit assignment issues [53].

5. Related works

In recent years, there has been a growing interest in designing practical and efficient risk-sensitive RL algorithms, with the risk measures introduced in Section 2.1 being among the most popular. Our approach sheds a new light on this topic, while allowing the unification of some existing methods into the same framework. A general treatment of these popular risk measures under a constrained-optimization perspective can be found in Prashanth and Fu [47]. For what concerns CVaR, an actor-critic algorithm was proposed by Chow and Ghavamzadeh [13], while a value-based perspective was considered by Chow et al. [14]. In a slightly different fashion, Chow et al. [15] designed a method to minimize the expected cost subject to CVaR constraints. More recently, CVaR was optimized with distributional approaches [16,62]. Tamar et al. [60] considered the class of coherent risk measures, generalizing a previous work on CVaR [61], and obtaining algorithms for its optimization. Interestingly, the same algorithm is equivalent to a particular case of ROSA applied to CVaR when using REINFORCE [66] for the inner optimization. The only difference is that the methods do not optimize in the same state-space since ROSA includes the augmentation variables.

For what concerns ERM, a modified optimal Bellman equation has been obtained by Howard and Matheson [27], and furthered studied in successive works [11,10,35]. A policy-search method for optimizing this objective has been recently proposed by Nass et al. [40]. The authors derived the exact risk-sensitive policy gradient and proposed an actor-only algorithm on top of it. This approach also turns out to be similar to our solution: it is, indeed, equivalent to instantiating ROSA for ERM using GPOMDP [6] as the inner optimizer, in which augmented returns are weighted by the current ERM. The optimization of this risk measure can also be seen as a particular KL-constrained robust MDP [42].

Di Castro et al. [18] proposed policy gradient algorithms for optimizing MV and sharpe-ratio. Actor-critic approaches for the same problem were proposed by Prashanth and Ghavamzadeh [48] and Tamar and Mannor [59]. Xie et al. [68] decomposed the mean-variance objective in the same form as (2) and designed an incremental block-coordinate algorithm to optimize it. The inner objective is directly optimized by gradient descent, while ROSA can use any RL algorithm for this purpose and reduces to their approach when choosing REINFORCE.

The mean-volatility was introduced by Bisi et al. [9], who also proposed a trust-region algorithm for its optimization. Recently, Zhang et al. [70] showed that optimizing this risk measure can be stated as a double optimization problem. Their block-coordinate method is similar to ROSA but can only be applied to mean-volatility, while ours generalizes to any of the risk measures of Section 2.1.

6. Experiments

We conducted an empirical analysis of the proposed approach on three domains: two toy problems (a multi-armed bandit problem and a more complex MDP problem), a trading environment based on real financial data, and standard robotics benchmarks, with the last two being contexts where risk aversion plays a fundamental role. The purpose of our experiments is three-fold:

1. to show that ROSA can be successfully combined with different risk-neutral RL algorithms;

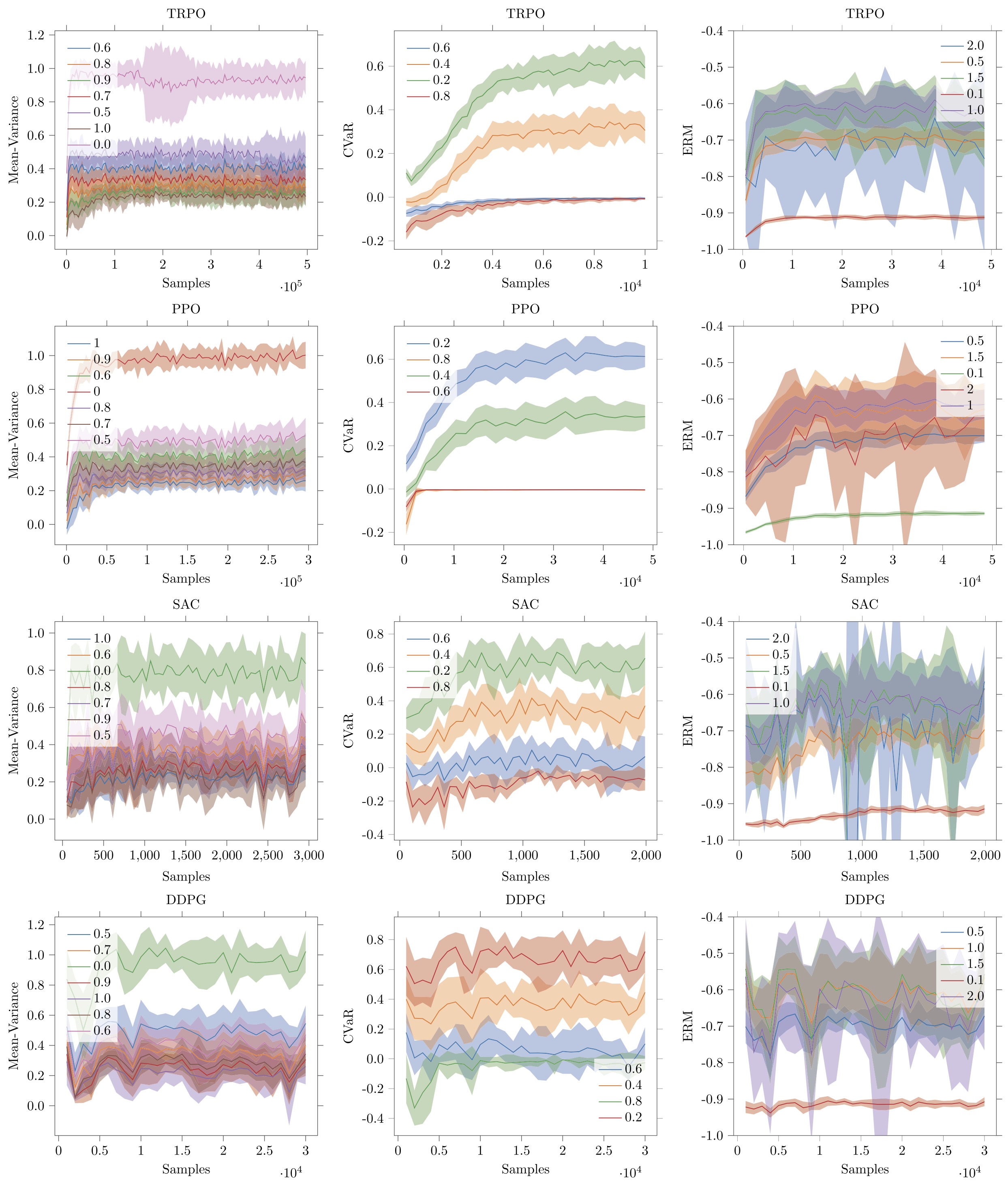

Fig. 1. Learning curves for ROSA optimizing MV, CVaR, and ERM when combined with TRPO, PPO, SAC, and DDPG.

2. to show that ROSA outperforms existing ad-hoc methodologies;
3. to show that ROSA scales to high-dimensional continuous domains which have received little to no attention in the risk-averse literature.

We focused on three risk measures: mean-variance, ERM and CVaR, which are representative of all the transformations we propose. ERM is indeed a particular case of utility function, while we did not test the mean-volatility since thorough

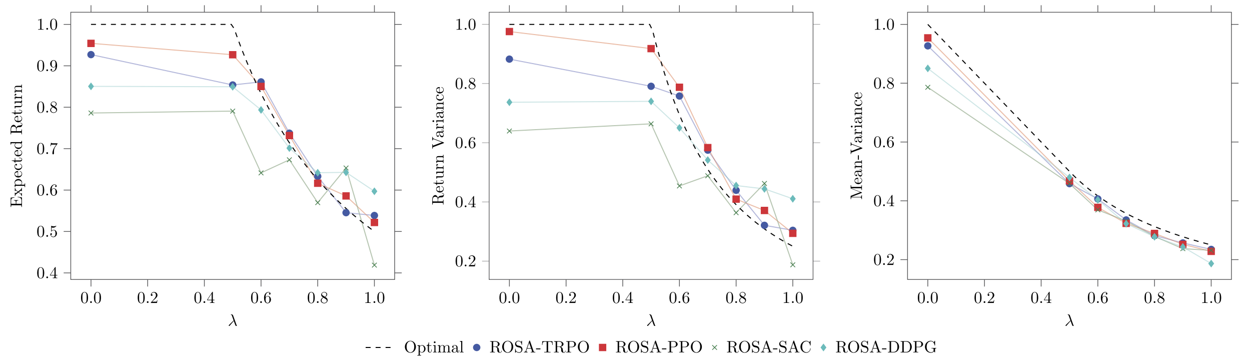

Fig. 2. Results of ROSA optimizing mean-variance when combined with different risk-neutral algorithms compared to the optimal solutions.

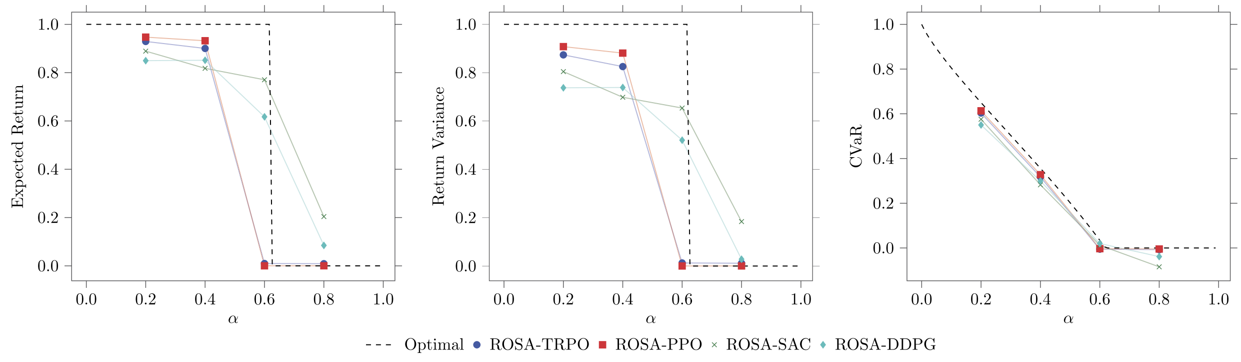

Fig. 3. Results of ROSA optimizing CVaR when combined with different risk-neutral algorithms compared to the optimal solutions.

experiments, for an algorithm that is conceptually equivalent to ROSA, have been recently provided by Zhang et al. [70]. We compared our algorithm with baselines from the risk-averse RL literature for each of the chosen risk measures. We employ, respectively, a policy search approach [40] for ERM, a block-coordinate approach [68] for mean-variance, and GCVaR Tamar et al. [61] for CVaR. The implementation details, together with additional results, can be found in the appendix.

6.1. Multi-armed bandit

We consider a multi-armed bandit problem with a continuous space of actions. More precisely, the agent can take any action in the interval $[-1, 1]$. When taking an action $a \in [-1, 1]$, the agent receives a reward R distributed as $\mathcal{N}(1 - |a|, (1 - |a|)^2)$. Clearly, the optimal risk-neutral policy is to take action $a = 0$ (which has maximum expected value equal to 1). However, this action has also the largest variance and is thus risky. Therefore, depending on the chosen risk measure, the agent needs to trade off between taking small actions (in absolute value) to maximize the expected return, and taking large actions to reduce risk. Since the reward is Gaussian, we are able to compute the optimal trade-off for mean-variance and CVaR in closed form, which allows us to perfectly evaluate the solutions learned by ROSA.

Results We combine ROSA with four different risk-neutral algorithms: TRPO [51], PPO [54], SAC [23], and DDPG [30]. We report the results for mean-variance in Fig. 2. More precisely, the three figures show the optimal values of expected return, return variance, and mean-variance as function of the risk-aversion parameter λ compared with the solutions learned by ROSA combined with the four base algorithms. Among these, the right-most plot is clearly the most indicative since it reports the actual objective function optimized by ROSA. Notably, all the algorithms almost perfectly learn the optimal mean-variance curve. The fact that expected returns and return variances of the learned policies are not as close to the optimal curve seems to indicate that mean-variance objective function is nearly flat in a neighborhood of the optimal points. Moreover, the slight sub-optimality of the learned risk-neutral solutions (for $\lambda = 0$) is probably due to the fact that online RL algorithms tend to be sensible to return variances (especially when learning with small batch sizes), thus converging to slightly risk-averse solutions.

The results for CVaR are shown in Fig. 3 in the same format as those for MV. Consistently with MV, ROSA learns almost perfectly the optimal CVaR curve when combined with all algorithms. We report the results for ERM in Fig. 4. Differently from before, for ERM we cannot compute the optimal solution in closed-form, so we simply report the learned pareto frontiers. We can appreciate that ROSA combined with all algorithms achieves very clear pareto frontiers, where solutions with high expectation/variance correspond to low risk-aversion parameters and viceversa.

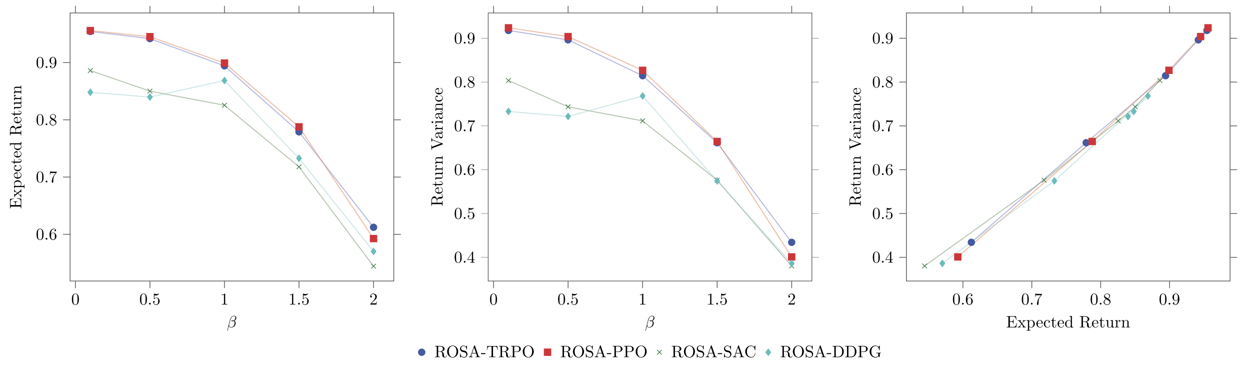

Fig. 4. Pareto frontiers for ROSA optimizing ERM when combined with different risk-neutral algorithms.

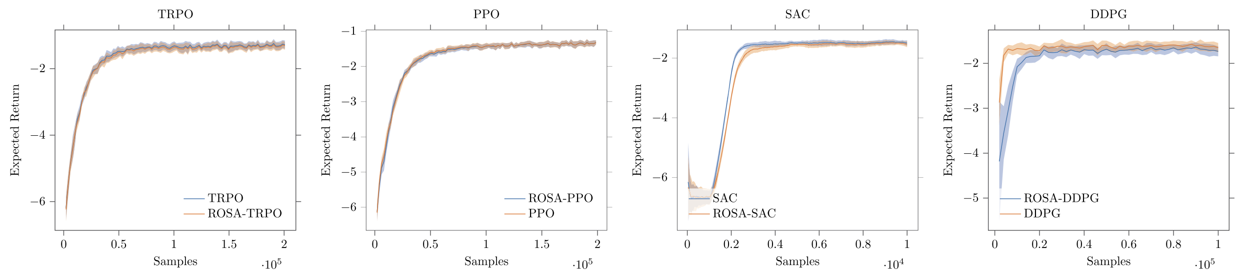

Fig. 5. Comparison between the base risk-neutral RL algorithms with and without ROSA's state augmentation in the Point Reacher domain.

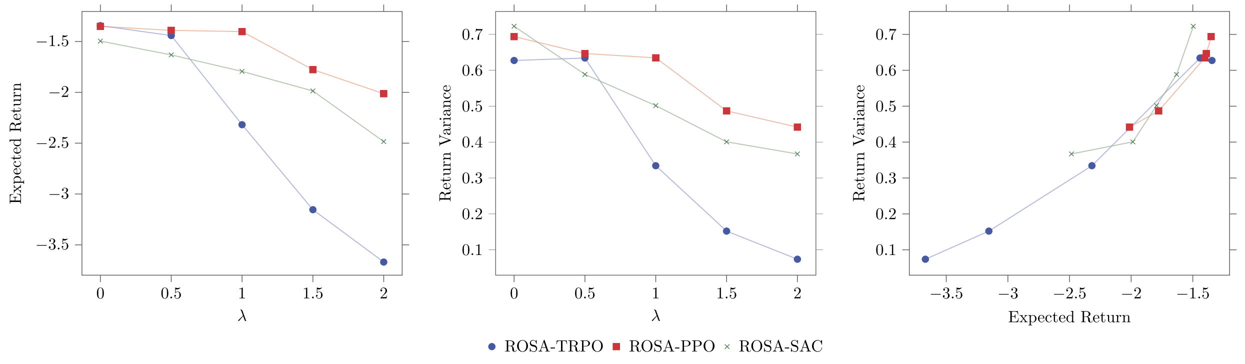

Fig. 6. Results of ROSA optimizing MV when combined with different risk-neutral algorithms on the Point Reacher domain.

Fig. 1 report the learning curves, for Multi-armed bandit for all risk measures and base algorithms. Here we notice that MV seems the simplest risk measure to optimize as all algorithms converge quickly with a stable learning behavior. On the other hand, ERM seems the most difficult and, due to its exponentiated nature, makes some algorithms (especially the off-policy ones) more unstable. Nonetheless, all curves converge to good solutions as we have already seen in the previous plots.

6.2. Point Reacher

In the second toy problem, the agent controls a point mass that moves along the real line in order to bring it to a target location in the minimum number of steps. The state of the system is described by the position of the mass in the interval $[-10, 10]$, while the agent chooses (continuous) actions in $[-2, 2]$. If the system is in state s and the agent takes action a, the new state is $s' \sim \mathcal{N}(s + a, a^2)$ and the immediate reward is $r = -0.1|s'| + a^2$. The goal is the ball of radius 0.05 around the origin. Episodes have length at most 10 and terminate whenever the agent reaches a goal state. The initial state is drawn uniformly in $[-5.1, -5] \cup [5, 5.1]$.

Results First, we investigate the effects of the state augmentation on the learning process of the standard risk-neutral objective function. To this purpose, we run the original version of our four base algorithms for optimizing the expected return

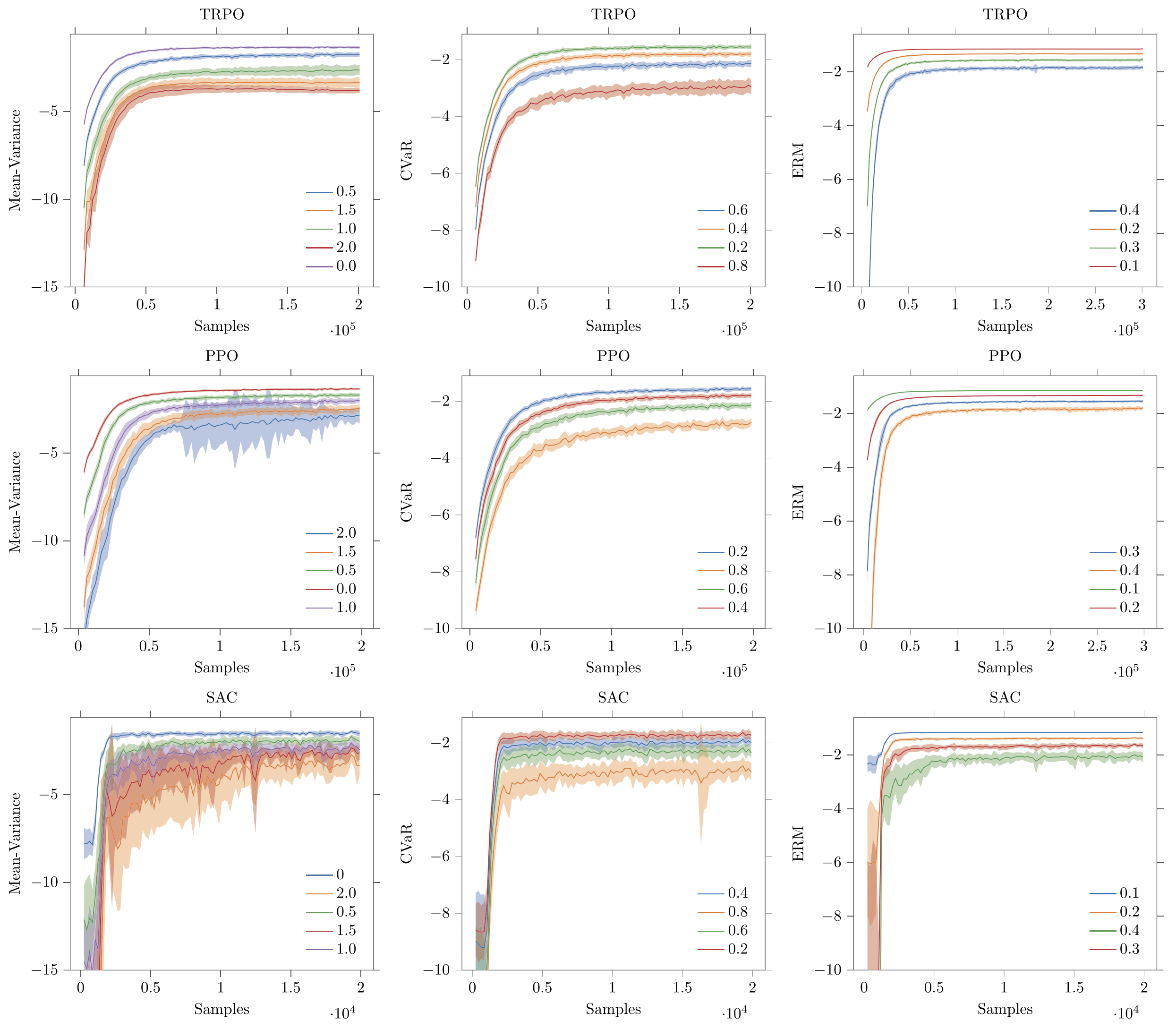

Fig. 7. Learning curves for ROSA optimizing MV, CVaR, and ERM when combined with different risk neutral algorithms on the Point Reacher domain.

and compare it to their ROSA counterparts with state augmentation (i.e., with rewards delayed to the end of the episode and state augmented by the running cumulative return). Fig. 5 shows the results, each point is obtained by evaluating 20 different policies (learned on 20 independent runs) on 1000 samples. While it is true that the state augmentation slightly slows down the learning process (especially for the off-policy algorithms), we notice that this performance degradation is never too severe. Moreover, convergence seems unaffected. This ablation experiment seems to confirm the intuitions of Section 4: in this multi-step problem, TRPO and PPO suffer less from the sparser reward signal, arguably thanks to the presence of a mechanism to explicitly tackle credit assignment issues, namely, the generalized advantage estimator [53].

The results of ROSA optimizing the different risk measures are shown in Fig. 6 for MV, Fig. 8 for CVaR, and Fig. 9 for ERM. Since we cannot evaluate the optimal solutions in closed-form as before, here we plot the mean-variance Pareto frontier achieved by the learned policies for all risk measures. As expected, all the algorithms achieve a clear approximated Pareto frontier when optimizing the mean-variance. However, in terms of hyper-volume, the Pareto frontier obtained by TRPO seems to be the best one. Good results are also obtained for the CVaR, while a clear frontier is not achieved in ERM. The latter result is probably due to the fact that the adopted risk aversion parameters are all very similar and do not encode much risk aversion. All the algorithms seem to keep an almost constant expected return but, interestingly, they manage to slightly reduce variance with higher levels of risk aversion.

By inspecting the learning curves in Fig. 7, it can be observed that TRPO plots are less noisy than PPO and SAC ones, especially when optimizing MV with higher level of risk-aversion. For what concerns SAC, an increased variability in the performance is also observed while optimizing the other objectives. This can be due to either its entropy regularization component, which may collide with risk minimization, or to its being off-policy. This latter feature, in particular, may conflict with the non-stationary nature of the costs, hampering the algorithm to correctly track them. A way to circumvent

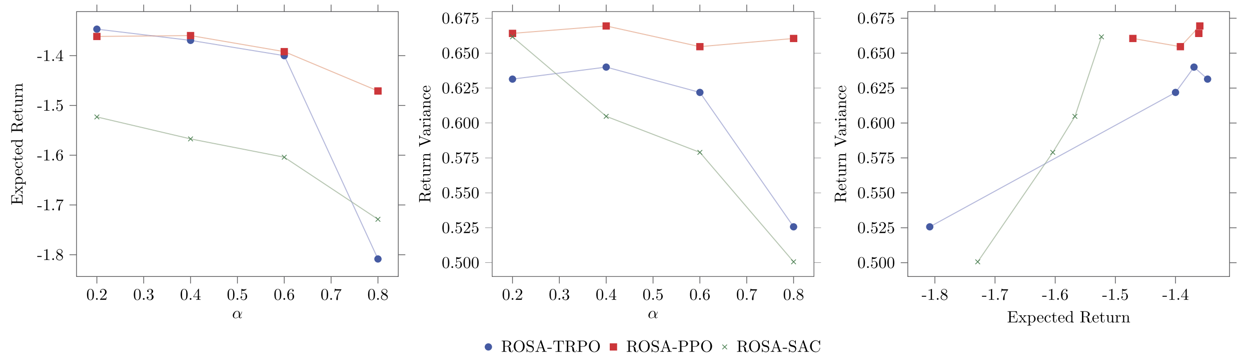

Fig. 8. Results of ROSA optimizing CVaR when combined with different risk-neutral algorithms on the Point Reacher domain.

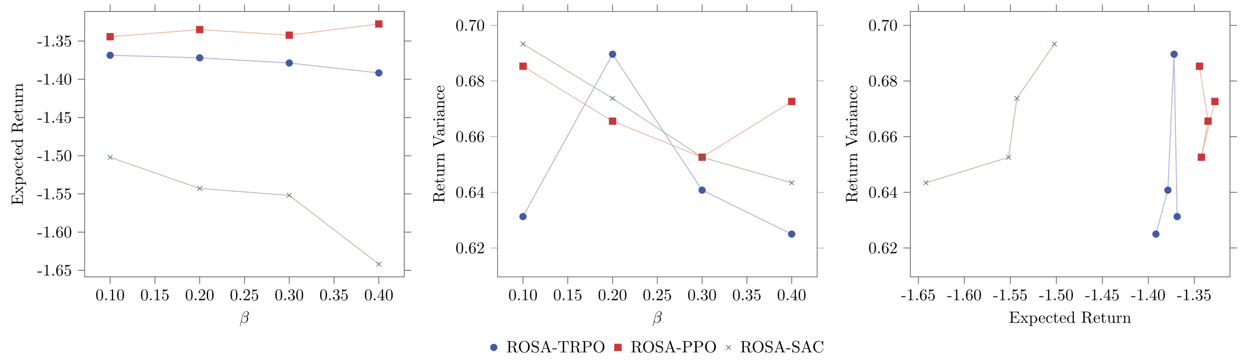

Fig. 9. Results of ROSA optimizing ERM when combined with different risk-neutral algorithms on the Point Reacher domain.

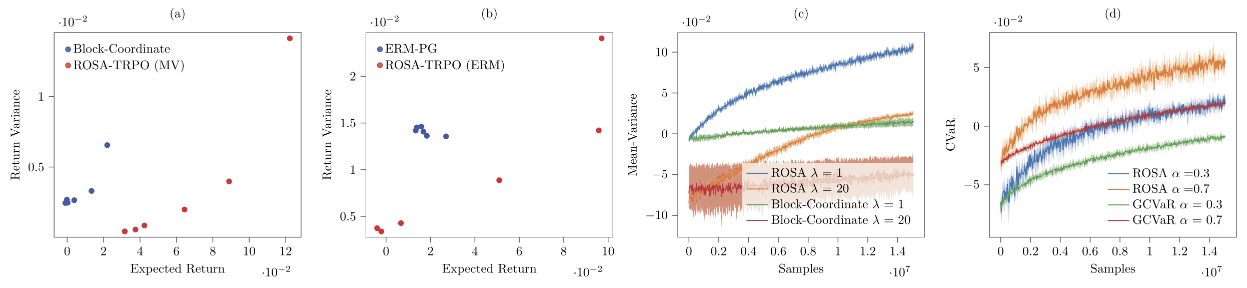

Fig. 10. Figures report the results obtained on the Trading environment instantiating ROSA for Mean-Variance, ERM, and CVaR. The optimization was performed employing TRPO Schulman et al. [51]. Figs. 1a-b show the Mean-Variance trade-off for, respectively, Mean-Variance and ERM optimization tasks. Pareto Frontiers for both our approach and baselines are included. Figs. 1c-d reports instead the learning curve for Mean-Variance and CVaR, respectively, comparing the convergence of our approach and the corresponding baselines, with two different values of risk aversion.

the problem could consist in saving in the replay buffer the original samples instead of the transformed ones, in order to transform them w.r.t. the current η by the time the are re-sampled from the buffer. This allows to avoid re-sampling rewards from previous tasks, and, for the CVaR case, to decrease the number of discarded samples per trajectory. We indeed expect η to grow with the iterations, since the policy is improving.

6.3. *Trading environment*

The S&P trading environment simulates a trading scenario in which an agent has to trade a single asset, whose price follows the daily S&P index values from the '80s until 2019. In each episode, the agent starts its trajectory from a random day of the S&P time-series and observes the ordered sequence of historical prices for 49 steps (two months). Its state is composed of its current portfolio, the price, and the time left until the end of the episode, plus the 10 previous prices. The action space in this task is discrete, and the three possible actions are: buy, sell or stay flat. The reward is equal to $R_t = a_t(p_t - p_{t-1}) - f|a_t - a_{t-1}|$, where the first term is the profit or loss given by action a_t, and the second term represents the transaction costs, where f is set to $7 \cdot 10^{-5}$. See Bisi et al. [9] for further details.

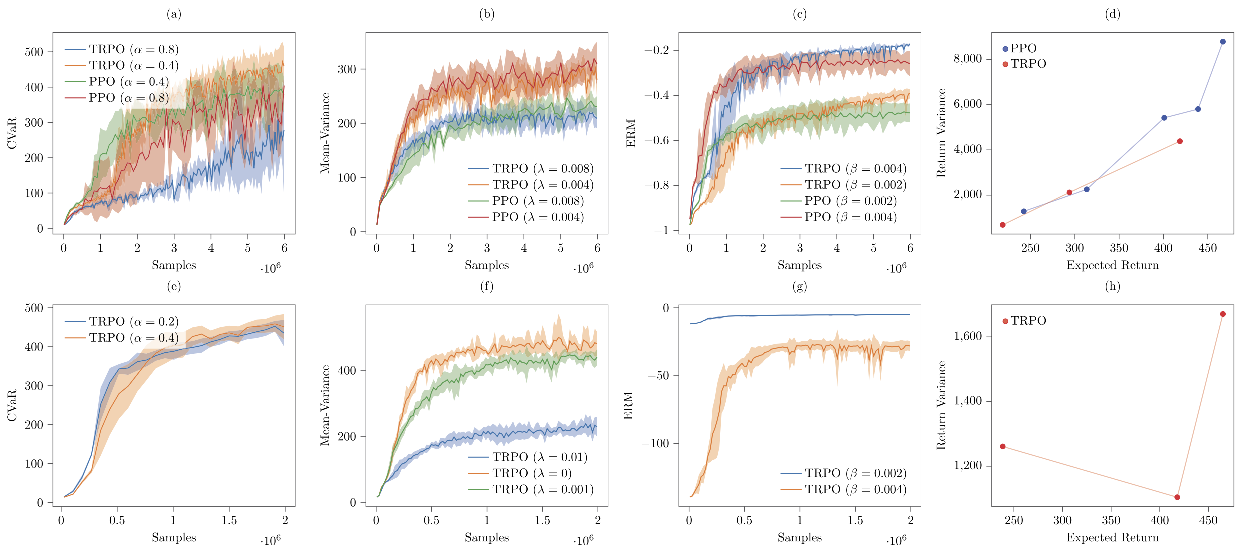

Fig. 11. The figures report the results obtained for the Walker and the Hopper environments on the first and the second row, respectively. Figures (a-c) and (e-g) display the learning curves obtained by employing ROSA to optimize, respectively, mean-variance, ERM, and CVaR. For each risk-measure, two levels of risk-aversion are shown. The inner optimization was performed by employing both TRPO and PPO for the Walker case, and only TRPO for the Hopper one. Shaded areas represent the standard deviation between 5 independent runs, while solid lines represent their means. Figures (d) and (h) show the trade-off between expected return and return variance obtained when optimizing mean-variance.

Results In Fig. 10, we report the results obtained in the Trading environment for all the selected risk measures. For this task, we instantiated ROSA with TRPO. A mean-variance Pareto frontier is plotted for both our approach and the corresponding baseline when optimizing mean-variance and ERM.[6] The algorithms were trained with the same budget of 15M samples. It can be noticed that ROSA learns solutions that dominate those of the baselines. For ERM (Fig. 10b) the baselines cannot even obtain a clear frontier, while in Fig. 10a it is clear that the learning process is still far from convergence. The improved learning speed for mean-variance and CVaR can be noticed from Fig. 10c-d, which show the learning curve for two levels of risk aversion. Notably, ROSA achieves faster and more stable learning behavior.

6.4. Robotic locomotion

For the robotic setting, two challenging environments from the MuJoCo simulator [63] were evaluated: Walker and Hopper.[7] The state of the robot is composed by its generalized position and velocity, while the controls are torques applicable to various joints. Both the state and the action spaces are continuous and high-dimensional. The reward is a linear combination of the following components: (1) a bonus for being alive, (2) a penalization for large action torques, (3) a bonus for moving forward, and (4) a bonus for high speed. Since these environments have deterministic dynamics, it can be difficult to understand the meaning of a risk-averse optimization. Therefore, we modified the task by introducing a perturbation to the action chosen by the agent. In particular, we added a white Gaussian *noise* to each action, with zero mean and a standard deviation proportional to the action magnitude. Intuitively, this models the fact that high torques have typically more unpredictable effects on the resulting system states. These environments, presenting high-dimensional continuous actions and states, are out of reach for the aforementioned baselines which performed very poorly in all our experiments. Their results have thus been neglected from our plots to ease readability. The experiments were run with a fixed budget of 6M and 2M of samples for the Walker and the Hopper environments, respectively. All the reported results are the average of 5 independent runs with shaded areas representing plus-minus standard deviation.

Results Fig. 11(a-d) shows the results we obtained on the Walker environment, where we optimized the three risk measures under consideration while instantiating ROSA with both PPO an TRPO. In particular, we report the learning curves of both algorithms for two of the risk-aversion coefficients we trained the agents with. It can be noticed that the learning process is stable and improving for all the objectives and for both risk-neutral algorithms. This empirically demonstrates that ROSA successfully optimizes the considered risk measures even in high-dimensional tasks when combined with state-of-the-art RL approaches. As expected, the most critical risk measure to be optimized seems to be the CVaR, which is known to pose many estimation issues [47]. In fact, both PPO and TRPO seems to struggle in optimizing CVaR with the higher level of risk

[6] We recall that ERM is an approximation to the mean-variance objective, so it makes sense to plot the same Pareto frontier.

[7] We employed the refined version of these environments from Pybullet [19]. Moreover we set the maximum length of each episode to 500 instead of 1000.

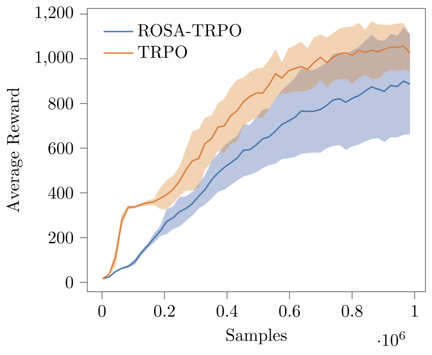

Fig. 12. Comparison between TRPO with and without ROSA's state augmentation on the Hopper environment.

aversion, while they perform well with the lower level. In Fig. 11(d), we report the approximated Pareto frontier obtained for the mean-variance criterion. It can be noticed that, independently from the base algorithm chosen, ROSA obtains nice trade-offs between expected return and return variance by varying the risk-aversion coefficient.

In Fig. 11 (e-h), we reported the results of ROSA optimizing the three risk measures on the Hopper environment, obtained using TRPO as base risk-neutral algorithm. Consistently with the Walker environment, ROSA successfully optimizes the different objectives with a stable and improving learning process. The mean-variance Pareto frontier in Fig. 11*(h)*, generated from the policies learned with three different values of λ, is less clear than before since the solution associated with $\lambda = 0.01$ is dominated by the one of $\lambda = 0.001$. This is probably due to the fact that the optimization process of the former did not reach convergence in 2M steps. However, as desired, both risk-averse solutions achieve a clear variance reduction with respect to the risk-neutral counterpart.

Finally, as illustrated in Section 4, we recall that the state-augmentation could increase the sample-complexity for learning. In our experiments, we indeed noticed that a greater number of samples is required to solve augmented tasks. However, this increase revealed to be reasonable in all our experiments, even in the most complex robot locomotion tasks. In Fig. 12, we compare the learning curves of TRPO optimizing the expected return on the Hopper environment with and without ROSA's state augmentation. We note that, despite the state augmentation, the learning process remains reasonably close to the one in the original state space, confirming what we have already noticed in the ablation study for the Point-Reacher environment.

7. Conclusions

We presented a unified framework for risk-averse RL which captures many of the most popular risk measures. Our simple meta-algorithm, ROSA, allows to optimize risk-sensitive policies by using any risk-neutral RL algorithm. We tested our approach on both a financial and a robotic setting. The empirical results presented reveal that our method combined with state-of-the-art policy optimization approaches scales to complex problems and outperforms ad-hoc risk-sensitive algorithms, while requiring minimal additional efforts, both in terms of computation and implementation, with respect to learning risk-neutral policies. A relevant direction for future work is to extend ROSA to the batch RL setting, which would further increase its applicability. Moreover, we could investigate whether our framework generalizes to a larger class of risk measures, such as the coherent ones. Empirically, we noticed that risk-averse agents tend to under-explore the environment, occasionally converging to poor local optima. As a possible workaround, it would be interesting to run ROSA starting from some good pre-trained risk-neutral policy.

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. Reproducibility details

Here we provide the configurations and hyperparameters that we adopted for all the considered algorithms in our experiments. We implemented ROSA on top of Stable Baselines [25]. For each algorithm and domain, we used the hyperparameters suggested in the library or slight variations of them.

A.1. Multi-armed bandit

TRPO We used the default MLP policy of Stable Baselines. The main parameters are: γ: 0.999, generalized advantage estimation factor: 0.95, maximum KL: 0.01, batch size: 200, entropy coefficient: 0.

PPO We used the default MLP policy of Stable Baselines. The main parameters are: γ: 0.999, clip range: 0.2, generalized advantage estimation factor: 0.95, learning rate: 0.003, batch size: 200, entropy coefficient: 0, number of mini-batches: 1.

SAC We used the custom SAC policy from Stable Baselines. The main parameters are: γ: 0.999, learning rate: 0.001, batch size: 50, buffer size: 1000, entropy coefficient: automatically learned, number of gradient steps: 5. Every 500 time steps, we collected 50 samples under the current policy to update the outer variables.

DDPG We used the default MLP policy of Stable Baselines. The main parameters are: γ: 0.999, batch size: 50, memory limit: 30000, number of rollouts per iteration: 10, number of training steps per iteration: 5, noise type: Ornstein-Uhlenbeck with 0.1 std. Every 500 time steps, we collected 50 samples under the current policy to update the outer variables.

A.2. Point Reacher

TRPO We used the default MLP policy of Stable Baselines. The main parameters are: γ: 0.999, generalized advantage estimation factor: 0.95, maximum KL: 0.01, batch size: 2048, entropy coefficient: 0.

PPO We used the default MLP policy of Stable Baselines. The main parameters are: γ: 0.999, clip range: 0.2, generalized advantage estimation factor: 0.95, learning rate: 0.005, batch size: 2048, entropy coefficient: 0, number of mini-batches: 1.

SAC We used the custom SAC policy from Stable Baselines. The main parameters are: γ: 0.999, learning rate: 0.001, batch size: 100, buffer size: 20000, entropy coefficient: automatically learned, number of gradient steps: 5. Every 500 time steps, we collected 50 episodes (i.e., 500 additional steps) under the current policy to update the outer variables.

A.3. Trading

For this environment we used a Boltzmann policy on top of a neural-network architecture composed of 2 layers with 64 hidden neurons each. We used the following parameters for TRPO: γ: 1, generalized advantage estimation factor: 1, maximum KL: 0.001, batch size: 700, entropy coefficient: 0, conjugate-gradient iterations: 10, conjugate-gradient damping: 0.01, value-function step size: 0.0003, value-function update iterations: 3. The episodes in this setting have a fixed length of 49 steps. The total number of iterations was 400.

The risk-aversion coefficients used for the two risk measures are $\lambda \in \{1, 5, 10, 20, 25, 30\}$ for mean-variance and $\beta \in \{-0.1, -2, -3, -3.5, -4, -5\}$ for ERM.

A.4. Walker and Hopper

We used a Gaussian policy on top of a neural-network architecture composed of 2 layers with 64 hidden neurons each. We used the following parameters for TRPO: γ: 0.999, generalized advantage estimation factor: 0.95, maximum KL: 0.01, batch size: 2048, entropy coefficient: 0. The episodes in this setting have a maximum length of 500. For PPO instead, the main parameters were: γ: 0.999, generalized advantage estimation factor: 0.95, batch size: 4096, minibatches: 32 and a learning rate starting from 0.0002 and decreasing with a linear schedule.

References

[1] D. Abel, D. Arumugam, L. Lehnert, M. Littman, State abstractions for lifelong reinforcement learning, in: International Conference on Machine Learning, PMLR, 2018, pp. 10–19.
[2] P. Artzner, F. Delbaen, J.-M. Eber, D. Heath, Coherent measures of risk, Math. Finance 9 (3) (1999) 203–228.
[3] A. Barreto, W. Dabney, R. Munos, J.J. Hunt, T. Schaul, H.P. van Hasselt, D. Silver, Successor features for transfer in reinforcement learning, Adv. Neural Inf. Process. Syst. 30 (2017).
[4] N. Bäuerle, J. Ott, Markov decision processes with average-value-at-risk criteria, Math. Methods Oper. Res. 74 (3) (2011) 361–379.
[5] N. Bäuerle, U. Rieder, More risk-sensitive Markov decision processes, Math. Oper. Res. 39 (1) (2014) 105–120.
[6] J. Baxter, P.L. Bartlett, Infinite-horizon policy-gradient estimation, J. Artif. Intell. Res. 15 (2001) 319–350.
[7] A. Beck, L. Tetruashvili, On the convergence of block coordinate descent type methods, SIAM J. Optim. 23 (4) (2013) 2037–2060.
[8] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Dębiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, et al., Dota 2 with large scale deep reinforcement learning, arXiv preprint, arXiv:1912.06680, 2019.
[9] L. Bisi, L. Sabbioni, E. Vittori, M. Papini, M. Restelli, Risk-averse trust region optimization for reward-volatility reduction, in: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, Special Track on AI in FinTech, IJCAI-20, 2020, pp. 4583–4589.
[10] V.S. Borkar, Q-learning for risk-sensitive control, Math. Oper. Res. 27 (2) (2002) 294–311.
[11] V.S. Borkar, S.P. Meyn, Risk-sensitive optimal control for Markov decision processes with monotone cost, Math. Oper. Res. 27 (1) (2002) 192–209.
[12] E. Brunskill, L. Li, Sample complexity of multi-task reinforcement learning, arXiv preprint, arXiv:1309.6821, 2013.
[13] Y. Chow, M. Ghavamzadeh, Algorithms for cvar optimization in mdps, in: Advances in Neural Information Processing Systems, 2014, pp. 3509–3517.
[14] Y. Chow, A. Tamar, S. Mannor, M. Pavone, Risk-sensitive and robust decision-making: a cvar optimization approach, in: Advances in Neural Information Processing Systems, 2015, pp. 1522–1530.
[15] Y. Chow, M. Ghavamzadeh, L. Janson, M. Pavone, Risk-constrained reinforcement learning with percentile risk criteria, J. Mach. Learn. Res. 18 (1) (2017) 6070–6120.

[16] W. Dabney, G. Ostrovski, D. Silver, R. Munos, Implicit quantile networks for distributional reinforcement learning, in: International Conference on Machine Learning, PMLR, 2018, pp. 1096–1105.
[17] M.P. Deisenroth, G. Neumann, J. Peters, A Survey on Policy Search for Robotics, 2013, Now publishers.
[18] D. Di Castro, A. Tamar, S. Mannor, Policy gradients with variance related risk criteria, arXiv preprint, arXiv:1206.6404, 2012.
[19] Y.B.E. Coumans, Pybullet, a Python Module for Physics Simulation for Games, Robotics and Machine Learning, 2016.
[20] J. Garcıa, F. Fernández, A comprehensive survey on safe reinforcement learning, J. Mach. Learn. Res. 16 (1) (2015) 1437–1480.
[21] M. Ghavamzadeh, M. Petrik, Y. Chow, Safe policy improvement by minimizing robust baseline regret, in: Advances in Neural Information Processing Systems, 2016, pp. 2298–2306.
[22] C. Gollier, The Economics of Risk and Uncertainty, Edward Elgar Publishing Limited, 2018.
[23] T. Haarnoja, A. Zhou, P. Abbeel, S. Levine, Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor, arXiv preprint, arXiv:1801.01290, 2018.
[24] N. Heess, D. TB, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. Eslami, et al., Emergence of locomotion behaviours in rich environments, arXiv preprint, arXiv:1707.02286, 2017.
[25] A. Hill, A. Raffin, M. Ernestus, A. Gleave, A. Kanervisto, R. Traore, P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, Y. Wu, Stable baselines, https://github.com/hill-a/stable-baselines, 2018.
[26] R.A. Howard, J.E. Matheson, Risk-sensitive Markov decision processes, Manag. Sci. 18 (7) (1972) 356–369.
[27] R.A. Howard, J.E. Matheson, Risk-sensitive Markov decision processes, Manag. Sci. 18 (7) (1972) 356–369, Publisher: INFORMS.
[28] J. Kober, J.A. Bagnell, J. Peters, Reinforcement learning in robotics: a survey, Int. J. Robot. Res. 32 (11) (2013) 1238–1274.
[29] S. Levine, C. Finn, T. Darrell, P. Abbeel, End-to-end training of deep visuomotor policies, J. Mach. Learn. Res. 17 (1) (2016) 1334–1373.
[30] T.P. Lillicrap, J.J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, Continuous control with deep reinforcement learning, arXiv preprint, arXiv:1509.02971, 2015.
[31] S.H. Lim, H. Xu, S. Mannor, Reinforcement learning in robust Markov decision processes, in: Advances in Neural Information Processing Systems, 2013, pp. 701–709.
[32] A. Majumdar, M. Pavone, How should a robot assess risk? Towards an axiomatic theory of risk in robotics, in: Robotics Research, Springer, 2020, pp. 75–84.
[33] J.G. March, Learning to be risk averse, Psychol. Rev. 103 (2) (1996) 309.
[34] O. Mihatsch, R. Neuneier, Risk-sensitive reinforcement learning, Mach. Learn. 49 (2–3) (2002) 267–290.
[35] O. Mihatsch, R. Neuneier, Risk-sensitive reinforcement learning, Mach. Learn. 49 (2–3) (2002) 267–290, Publisher: Springer.
[36] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533.
[37] T.M. Moldovan, P. Abbeel, Risk aversion in Markov decision processes via near optimal Chernoff bounds, in: Advances in Neural Information Processing Systems, 2012, pp. 3131–3139.
[38] J. Moody, M. Saffell, Learning to trade via direct reinforcement, IEEE Trans. Neural Netw. 12 (4) (2001) 875–889.
[39] R. Munos, T. Stepleton, A. Harutyunyan, M. Bellemare, Safe and efficient off-policy reinforcement learning, in: Advances in Neural Information Processing Systems, 2016, pp. 1054–1062.
[40] D. Nass, B. Belousov, J. Peters, Entropic risk measure in policy search, arXiv preprint, arXiv:1906.09090, 2019.
[41] A. Nilim, L. El Ghaoui, Robust control of Markov decision processes with uncertain transition matrices, Oper. Res. 53 (5) (2005) 780–798.
[42] T. Osogami, Robustness and risk-sensitivity in Markov decision processes, in: Advances in Neural Information Processing Systems, 2012, pp. 233–241.
[43] A.B. Owen, Monte Carlo Theory, Methods and Examples, 2013.
[44] S. Padakandla, A survey of reinforcement learning algorithms for dynamically varying environments, ACM Comput. Surv. 54 (6) (2021) 1–25.
[45] J. Peters, S. Schaal, Policy gradient methods for robotics, in: 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2006, pp. 2219–2225.
[46] L. Pinto, J. Davidson, R. Sukthankar, A. Gupta, Robust adversarial reinforcement learning, in: Proceedings of the 34th International Conference on Machine Learning, vol. 70, 2017, pp. 2817–2826, JMLR. org.
[47] L.A. Prashanth, M. Fu, Risk-sensitive reinforcement learning: a constrained optimization viewpoint, arXiv preprint, arXiv:1810.09126, 2018.
[48] L.A. Prashanth, M. Ghavamzadeh, Actor-critic algorithms for risk-sensitive mdps, in: Advances in Neural Information Processing Systems, 2013, pp. 252–260.
[49] M.L. Puterman, Markov Decision Processes: Discrete Stochastic Dynamic Programming, John Wiley & Sons, 2014.
[50] R.T. Rockafellar, S. Uryasev, et al., Optimization of conditional value-at-risk, J. Risk 2 (2000) 21–42.
[51] J. Schulman, S. Levine, P. Abbeel, M. Jordan, P. Moritz, Trust region policy optimization, in: International Conference on Machine Learning, 2015, pp. 1889–1897.
[52] J. Schulman, S. Levine, P. Abbeel, M. Jordan, P. Moritz, Trust region policy optimization, in: International Conference on Machine Learning, 2015, pp. 1889–1897.
[53] J. Schulman, P. Moritz, S. Levine, M. Jordan, P. Abbeel, High-dimensional continuous control using generalized advantage estimation, arXiv preprint, arXiv:1506.02438, 2015.
[54] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms, arXiv preprint, arXiv:1707.06347, 2017.
[55] Y. Shen, R. Huang, C. Yan, K. Obermayer, Risk-averse reinforcement learning for algorithmic trading, in: 2014 IEEE Conference on Computational Intelligence for Financial Engineering & Economics, CIFEr, IEEE, 2014, pp. 391–398.
[56] D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al., Mastering the game of go with deep neural networks and tree search, Nature 529 (7587) (2016) 484.
[57] S.P. Singh, R.S. Sutton, Reinforcement learning with replacing eligibility traces, Mach. Learn. 22 (1) (1996) 123–158.
[58] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, MIT Press, 2018.
[59] A. Tamar, S. Mannor, Variance adjusted actor critic algorithms, arXiv preprint, arXiv:1310.3697, 2013.
[60] A. Tamar, Y. Chow, M. Ghavamzadeh, S. Mannor, Policy gradient for coherent risk measures, in: Advances in Neural Information Processing Systems, 2015, pp. 1468–1476.
[61] A. Tamar, Y. Glassner, S. Mannor, Optimizing the cvar via sampling, in: Twenty-Ninth AAAI Conference on Artificial Intelligence, 2015.
[62] Y.C. Tang, J. Zhang, R. Salakhutdinov, Worst cases policy gradients, in: Conference on Robot Learning, PMLR, 2020, pp. 1078–1093.
[63] E. Todorov, T. Erez, Y. Tassa, Mujoco: a physics engine for model-based control, in: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2012, pp. 5026–5033.
[64] P. Tseng, Convergence of a block coordinate descent method for nondifferentiable minimization, J. Optim. Theory Appl. 109 (3) (2001) 475–494.
[65] W. Wiesemann, D. Kuhn, B. Rustem, Robust Markov decision processes, Math. Oper. Res. 38 (1) (2013) 153–183.
[66] R.J. Williams, Simple statistical gradient-following algorithms for connectionist reinforcement learning, Mach. Learn. 8 (3–4) (1992) 229–256.
[67] S.J. Wright, Coordinate descent algorithms, Math. Program. 151 (1) (2015) 3–34.
[68] T. Xie, B. Liu, Y. Xu, M. Ghavamzadeh, Y. Chow, D. Lyu, D. Yoon, A block coordinate ascent algorithm for mean-variance optimization, in: Advances in Neural Information Processing Systems, 2018, pp. 1065–1075.
[69] H. Xu, S. Mannor, Distributionally robust Markov decision processes, in: Advances in Neural Information Processing Systems, 2010, pp. 2505–2513.
[70] S. Zhang, B. Liu, S. Whiteson, Per-step reward: a new perspective for risk-averse reinforcement learning, arXiv preprint, arXiv:2004.10888, 2020.