IEEE Control Systems Letters paper presented at
2023 American Control Conference (ACC)
San Diego, CA, USA. May 31 - June 2, 2023

Exponential TD Learning: A Risk-Sensitive Actor-Critic Reinforcement Learning Algorithm

Erfaun Noorani*, Christos N. Mavridis*, and John S. Baras

Abstract— **Incorporating risk in the decision-making process has been shown to lead to significant performance improvement in optimal control and reinforcement learning algorithms. We construct a temporal-difference risk-sensitive reinforcement learning algorithm using the exponential criteria commonly used in risk-sensitive control. The proposed method resembles an actor-critic architecture with the 'actor' implementing a policy gradient algorithm based on the exponential of the reward-to-go, which is estimated by the 'critic'. The novelty of the update rule of the 'critic' lies in the use of a modified objective function that corresponds to the underlying multiplicative Bellman's equation. Our results suggest that the use of the exponential criteria accelerates the learning process and reduces its variance, i.e., risk-sensitiveness can be utilized by actor-critic methods and can lead to improved performance.**

I. Introduction

The need for robust Reinforcement Learning (RL) algorithms is evident from the often brittle and non-robust (to disturbances and model perturbations) performance of classical (risk-neutral) RL algorithms [1]. Various approaches have been proposed to mitigate the shortcomings of classical RL algorithms; from risk-sensitive [2] and robust [3] RL algorithms, to various postulated regularized objectives [4], [5]. Among the various risk-sensitive objectives, the exponential performance criterion [6] has been widely used due to its mathematical convenience and firm theoretical foundations rooted in the large deviation theory [7].

The underlying motivation for this RL objective comes from the connection of risk-sensitive control to robust output feedback control and its dynamic game formulation, which established the robustness properties of the risk-sensitive exponential criterion [6], [8], [9]. In our prior work, we showed that the regularized objectives are closely related to risk-sensitive criteria, by using the dual representation of convex and coherent risk-measures [10]. These results further encourage the recent interest in risk-sensitive RL which has led to the development of more robust and efficient RL [2].

Using exponential performance criteria in a dynamic programming formulation results in the modification of the classical Hamilton-Jacobi-Bellman (HJB) equation to a multiplicative HJB [11]. As a result, the traditional Temporal-Difference (TD) rule in reinforcement learning, e.g., in Q-learning or actor-critic methods [12], which is rooted in the principles of the HJB equation, needs to be modified as well. A direct approach of a Q-learning algorithm with such an exponential objective has been suggested in [13], inspired by the Value-Iteration algorithm for exponential criteria [11]. In model-free policy optimization methods, however, taking the gradient of an exponential objective is challenging due to the nonlinearity of the exponential function. As a result, existing approaches often bypass this computation by using heuristic methods or making use of the known connection between risk-sensitive and robust control [14] to approximate the risk-sensitive optimization with a min-max robust adversarial RL [3], [15]. In our prior work, we extended the Monte Carlo policy gradient algorithm REINFORCE [4] and developed a risk-sensitive Monte Carlo method based on the exponential criteria, Risk-sensitive REINFORCE [16].

In this work, we further extend these results and propose a temporal-difference risk-sensitive actor-critic learning algorithm (see, e.g., [12], [17]) based on the exponential criterion. In particular, an 'actor' model is used to implement a policy gradient algorithm based on a function approximation of the exponential of the reward-to-go, which is estimated and updated by a 'critic' model with every observation. The difference lies in the update rule of the 'critic', as it makes use of a modified objective function that corresponds to the underlying multiplicative HJB equation of the dynamic programming problem. Our experimental results show that the use of the exponential criteria accelerates the learning process and reduces its variance, resulting in a policy method with a higher expected return, i.e., risk-sensitivity can be utilized by actor-critic methods and can lead to sample efficiency and improved performance.

The rest of this paper is organized as follows: Section II offers a summary of classical RL and defines our notation. In section III, we develop our risk-sensitive actor-critic algorithm based on the exponential criteria using a modified temporal-difference learning rule, and provide a game-theoretic interpretation of the robustness of the proposed methodology. Section IV illustrates our experimental results that show increased performance and reduced variance compared to risk-neutral RL algorithms. Finally, Section V concludes the paper.

II. Risk-Neutral RL

A reinforcement learning problem is typically modeled using a Markov Decision Process (MDP) which is represented by a tuple $\mathcal{M}=(\mathcal{S},\mathcal{A},p_0,P,r,\gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are, respectively, the state and action spaces. At each time-step t, starting with the initial state s_0 drawn from the distribution

*E. Noorani and C. Mavridis contributed equally to this work.

All authors are with the Department of Electrical and Computer Engineering and the Institute for System Research (ISR) at the University of Maryland, College Park, MD, USA. {enoorani, mavridis, baras}@umd.edu

Research partially supported by ONR grant N00014-17-1-2622, by a grant from the Army Research Lab and by the Clark Foundation.

p_0, the agent perceives the state of the environment s_t and executes an action a_t. The environment transitions to a successor state s_{t+1} with probability $p(s_{t+1}|s_t,a_t)$ given by the kernel P. The agent receives a reward $r_t := r(s_t,a_t)$; $\gamma \in (0,1]$ is a discounting factor.

The behavior of an RL agent is determined by its policy. Here, we consider randomized policies. A (randomized) policy $\pi(\cdot|s)$ is a probability distribution over action space given the state, which prescribes the probability of taking an action a when in state s. Here, we assume policies are differentiable and parameterized, e.g., a neural network, $\pi(\cdot|s_t;\theta)$ where $\theta \in \mathbb{R}^d$ is a vector of parameters. By following policy π, the agent generates trajectory τ (a sequence of states and actions). The agent's policy and the system transition probabilities induce a trajectory distribution given by

$$\rho_\pi(\tau) = p_0 \prod_{t=0}^{|\tau|-1} \pi(a_t|s_t;\theta) p(s_{t+1}|s_t,a_t) \tag{1}$$

The RL agent aims to find a policy that maximizes some desired performance criterion during an episode. In risk-neutral RL, the objective is to optimize some long-run average of some desired quantity. A common example of a risk-neutral objective in the RL literature is expected (discounted) cumulative reward, i.e.,

$$\max_\theta J(\theta) := \mathbb{E}_{\pi_\theta}\Big[R\Big] \tag{2}$$

where $R = \sum_{t=0}^{|\tau|-1} \gamma^t r(s_t,a_t)$ is the trajectory's total reward. The expectation is taken with respect to the trajectory distribution. That is, the expectation is taken over the space of trajectories $\mathcal{T}$ generated by following the policy, i.e., $s_0 \sim p_0$, $a_t \sim \pi(\cdot|s_t;\theta)$ and $s_{t+1} \sim p(\cdot|s_t,a_t)$. Risk-neutral RL as defined above has been extensively studied [18]–[20]. Below, we present a brief overview of the most commonly used RL algorithms that will set the base for our proposed algorithm in Section III.

A. Policy Gradient Algorithms

Policy gradient algorithms, such as the standard REINFORCE algorithm [21], are the most straightforward RL methods, using an iterative gradient ascent to find the optimal policy parameter

$$\theta_{t+1} = \theta_t + \alpha \widehat{\nabla J(\theta_t)} \tag{3}$$

where $\alpha \in \mathbb{R}$ is a step-size and is called learning rate, and $\widehat{\nabla J(\theta_t)} \in \mathbb{R}^d$ is an unbiased estimate of the gradient with respect to the policy parameter θ. The gradient of the (risk-neutral) objective of (2) with respect to the policy parameters is given by policy gradient theorem [22]

$$\nabla J(\theta) \propto \mathbb{E}_{\pi_\theta}\Big[R \sum_{t=0}^{|\tau|-1} \nabla \log \pi_\theta(a_t|s_t)\Big] \tag{4}$$

where we use $\pi_\theta(s) := \pi(a|s;\theta)$ as a shorthand notation. Policy gradient methods that are based on Monte Carlo estimation of the expectation in (4) suffer from high variance. To reduce variance, by taking advantage of the temporal structure of the problem and causality, it can be shown that the gradient could be re-written in terms of reward-to-go $R_t := \sum_{t'=t}^{|\tau|-1} \gamma^{t'-t} r(s_{t'},a_{t'})$ as follows:

$$\nabla J(\theta) \propto \mathbb{E}_{\pi_\theta}\Big[\sum_{t=0}^{|\tau|-1} R_t \nabla \log \pi_\theta(a_t|s_t)\Big] \tag{5}$$

Using (5), the update rule, in the standard REINFORCE algorithm, is given by

$$\theta_{k+1} = \theta_k + \alpha R_t \frac{\nabla \pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)}. \tag{6}$$

B. Using Baselines

To further reduce the variance associated with the gradient estimations of (4) and (5), which is a must in complex environments, various techniques have been employed. Baseline methods are among the most common and are based on subtracting an appropriately chosen baseline from the reward-to-go R_t to reduce the variance without introducing bias. Using baselines, we have [1]

$$\nabla J(\theta) \propto \mathbb{E}_{\pi_\theta}\Big[\sum_{t=0}^{|\tau|-1} \Big(R_t - b(s_t)\Big) \nabla \log \pi_\theta(a_t|s_t)\Big] \tag{7}$$

where $b(s_t)$ is a state-dependent function. The existence of an optimal state-dependent baseline has been shown, however, it is hard to find [23]. A common baseline in practice is the estimate of the value function, i.e., $b(s_t) = V^{\pi_\theta}(s_t)$, where $V^{\pi_\theta}(s_t) := \mathbb{E}_{\pi_\theta}[R_t|s_t]$. The effect of action-dependent baselines over state-dependent baselines is subject to debate. As we will discuss, a particularly convenient property of using exponential criteria is that it alleviates the need for such approaches [10].

C. Using Function Approximation

The reward-to-go $R_t := \sum_{t'=t}^{|\tau|-1} \gamma^{t'-t} r(s_{t'},a_{t'})$ in (5) needs to be approximated by Monte Carlo simulation with the rewards $r(s_t,a_t)$ being collected over an adequately long period of time, during which the RL agent cannot implement any policy updates. Function approximation can be used to estimate R_t by a value function $V^{\pi_\theta}(s_t) := \mathbb{E}_{\pi_\theta}[R_t|s_t]$, which can be shown to satisfy the Bellman's equation

$$V^{\pi_{\theta^*}}(s_t) = \mathbb{E}_{\pi_{\theta^*}}\Big[r(s_t,a_t) + \gamma V^{\pi_{\theta^*}}(s_{t+1}) \mid s_t\Big] \tag{8}$$

where $a_t \sim \pi_{\theta^*}(\cdot|s_t)$.

The fact that (8) is a contraction mapping has given rise to stochastic approximation algorithms that try to asymptotically minimize the mean-squared error

$$\min_\theta \ \mathbb{E}_{\pi_\theta}\Big[\|r(s_t,a_t) + \gamma V^{\pi_\theta}(s_{t+1}) - V^{\pi_\theta}(s_t)\|^2 \mid s_t\Big]$$

where $a_t \sim \pi_\theta(\cdot|s_t)$. A fact that is used by temporal-difference RL methods that employ learning models (e.g. neural networks [20] or other learning algorithms [24]–[26]) to learn the optimal value function.

D. Temporal-Difference Methods

Model-free temporal-difference RL methods are mainly represented by Actor-Critic (AC) methods. AC methods use two learning systems (e.g., neural networks) to estimate the optimal policy (actor) and reward-to-go (critic), given by

$$\begin{cases} \theta_{t+1} = \theta_t + \alpha\left(\hat{R}_t - V(s_t;w_t)\right)\frac{\nabla \pi_{\theta_t}(a_t|s_t)}{\pi_{\theta_t}(a_t|s_t)} \\ w_{t+1} = w_t - \bar{\alpha}\nabla J_c(s_t;w_t,\theta_t) \end{cases} \tag{9}$$

where the function V is parameterized with a vector of parameters $w_t \in \mathbb{R}^{d'}$, and $J_c(s_t;w_t,\theta_t) := \|\hat{R}_t - V(s;w_t)\|^2$. In this case, $\hat{R}_t$ is given by

$$\hat{R}_t := r(s_t,a_t) + \gamma V(s_{t+1},w_t) \simeq R_t$$

where $a_t \sim \pi_{\theta_t}(\cdot|s_t)$. The reward-to-go estimate and the form of the objective function J_c of the critic is what gives it the name temporal-difference learning. We note that not all actor-critic algorithms are temporal-difference methods, with many adhering to the batch learning approach described in Section II-A.

Special consideration needs to be given to the step sizes $\{\alpha,\bar{\alpha}\}$ as their choice heavily affects the learning process. In theory, the step sizes $\{\alpha,\bar{\alpha}\}$ should decrease with time according to the theory of stochastic approximation algorithms [26], [27], i.e., $\sum_n \alpha(n) = \infty$, $\sum_n \alpha^2(n) < \infty$, and $\alpha(n)/\bar{\alpha}(n) \to 0$, a fact that is often overlooked in practice.

III. Risk-Sensitive RL

Risk-sensitive RL incorporates some notion of risk to the agent's objective, e.g., higher moments of the return. Of particular interest is the exponential criterion, i.e.,

$$J_\beta(\theta) := \mathbb{E}_{\pi_\theta}\left[\beta e^{\beta R}\right], \tag{10}$$

which has been well-studied in the context of risk-sensitive control for decades. To see why the exponential criterion in (10) incorporates risk into the objective function, one can take its Taylor expansion which consists of an infinite sum of higher moments of the return with diminishing weights for small values of β, i.e.,

$$\mathbb{E}\left[\beta e^{\beta R}\right] = \beta + \beta^2\mathbb{E}\left[R\right] + \frac{\beta^3}{2}\mathbb{E}\left[R^2(\tau)\right] + \dots \tag{11}$$

Note that the optimization of such an objective corresponds to the optimization of the tail of the distribution of the return [9]. Also, one can see the incorporation of risk into the exponential criterion, by noting that, for β with small magnitude, the maximization of the exponential criterion is approximately a trade-off between the risk-neutral objective and the variance of the return. Note that as the risk parameter β approaches zero, the optimization of the exponential objective (10) is equivalent to the risk-neutral case (2).

A. Game-Theoretic Interpretation

Theorem 1 in [10] spells out the relationship between the risk-sensitive RL with exponential criteria and max-min optimization. It states that for a positive risk parameter β>0 (risk-seeking), the risk-sensitive exponential criterion has a dual representation given by

$$J_\beta(\pi) = \sup_{\hat{\pi}}\left\{\mathbb{E}_{\hat{\pi}}\left[R\right] - \frac{1}{\beta}D\left(\rho_{\hat{\pi}},\rho_\pi\right)\right\},\ \beta > 0 \tag{12}$$

and for a negative risk parameter β<0 (risk-aversion) the dual representation is given by

$$J_\beta(\pi) = \inf_{\hat{\pi}}\left\{\mathbb{E}_{\hat{\pi}}\left[R\right] - \frac{1}{\beta}D\left(\rho_{\hat{\pi}},\rho_\pi\right)\right\},\ \beta < 0 \tag{13}$$

where

$$D(Q,P) = \begin{cases} \mathbb{E}_Q\left[\log\frac{dQ}{dP}\right] & \text{if } Q \ll P \\ \infty & \text{otherwise} \end{cases}$$

is the relative entropy of Q w.r.t. P (Kullback–Leibler (KL) divergence of probability distribution Q from P), and the support of $\rho_{\hat{\theta}}$ is contained within the support of ρ_θ, that is to say, $\rho_{\hat{\theta}}$ is absolutely continuous with respect to ρ_θ.

In this sense, a risk-averse agent with exponential criteria tries to maximize over the policy class in (12) and (13), and implicitly plays a non-cooperative game against "nature" (a hypothetical second player). The KL divergence $D(\rho_{\hat{\pi}},\rho_\pi)$ defines a measure of how much the adversary player, controlling ρ_π, can deviate from the current policy distribution $\rho_{\hat{\pi}}$. The parameter β determines the weight of the KL divergence term, as well as the nature of the game which depends on the sign of β.

B. Risk-Sensitive Policy Gradient

In [16], we recently proposed a risk-sensitive policy gradient algorithm, called Risk-Sensitive REINFORCE. This is a Monte Carlo algorithm, similar to REINFORCE, that seeks to find the optimal policy for exponential criteria with the update rule

$$\theta_{t+1} = \theta_t + \alpha\beta e^{\beta R_t}\frac{\nabla\pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)} \tag{14}$$

Remark 1: Note that the update rule is not proportional to the reward-to-go $R_t := \sum_{t'=t}^{|\tau|-1}\gamma^{t'-t}r(s_{t'},a_{t'})$, but to the exponential

$$\beta e^{\beta R_t} = \beta\prod_{t'=t}^{|\tau|-1}\exp\{\gamma^{t'-t}\beta r(s_{t'},a_{t'})\} \tag{15}$$

which is a significant difference as shown in (11). Positive and negative risk parameter β result in a risk-seeking and risk-averse behaviour, respectively.

Remark 2: By substituting the exponential with its Taylor series expansion, shown in (11), the risk-sensitive objective can be understood to provide a natural baseline (Section II-B). This has been shown in [16] and holds for the temporal-difference case (Section IV) as well. Such a baseline will be empirically shown to lead to considerable variance reduction and acceleration of learning.

While the reward-to-go term $\beta e^{\beta R_t}$ can be approximated by Monte Carlo simulation with the rewards being collected over an adequately long period of time, this update rule is not appropriate for online updates, i.e., when the policy is updated with each observation. In the next section, we

introduce an online actor-critic algorithm that simultaneously estimates the reward-to-go and updates the current policy.

C. Risk-Sensitive Temporal-Difference RL

To develop a risk-sensitive temporal-difference RL algorithm, we use two learning systems (e.g., neural networks) to estimate the optimal policy (actor) and reward-to-go (critic), similar to Section II-D. According to the objective function J_β in (10), we define the risk-sensitive value function of a policy π as

$$V_\beta^\pi(s_t) := \beta \mathbb{E}\left[e^{\beta \sum_{l=t}^{\infty} \gamma^{l-t} r(s_l,a_l)} | s_t\right], \quad a_l \sim \pi(\cdot|s_l) \tag{16}$$

We further define:

$$\bar{V}_\beta^\pi(s_t) := \frac{1}{\beta} V_\beta^\pi(s_t) = \mathbb{E}\left[e^{\beta \sum_{l=t}^{\infty} \gamma^{l-t} r(s_l,a_l)} | s_t\right] \tag{17}$$

where $a_l \sim \pi(\cdot|s_l)$ and by definition, $\bar{V}_\beta^\pi(\cdot) \geq 0$. The following relationship holds:

$$V_\beta^*(s_t, w_t) = \max_a \; e^{\beta r(s_t,a)} \mathbb{E}\left[e^{(V_\beta^*(s_{t+1},w_t))^\gamma} \mid s_t\right]. \tag{18}$$

The following actor-critic learning approach can be constructed:

$$\begin{cases} \theta_{t+1} = \theta_t + \alpha|\beta|\left(R_t^\beta - \bar{V}_\beta(s_t;w_t)\right)\frac{\nabla \pi(a_t|s_t;\theta_t)}{\pi(a_t|s_t;\theta_t)} \\ w_{t+1} = w_t - \bar{\alpha}\nabla J_r(s_t;w_t,\theta_t) \end{cases} \tag{19}$$

where $R_t^\beta = \exp\left[\beta r(s_t,a_t) + \gamma \ln \bar{V}_\beta(s_{t+1};w_t)\right]$, and

$$J_r(s_t;w_t,\theta_t) = \|e^{\beta r(s_t,a_t)+\gamma \ln \bar{V}_\beta(s_{t+1};w_t)} - \bar{V}_\beta(s_t;w_t)\|^2.$$

Note that $a_t \sim \pi_{\theta_t}(\cdot|s_t)$. In contrast to the risk-neutral case, here

$$R_t^\beta = \exp\left[\beta r(s_t,a_t) + \gamma \ln \bar{V}_\beta(s_{t+1};w_t)\right].$$

This recursion is not a fixed-point iteration but rather a stochastic gradient descent scheme. The remarks on the stepsizes $\{\alpha, \bar{\alpha}\}$ hold similarly to Section II-D. The following remarks are also important.

Remark 3: Note that the update rule in (19) is now based on minimizing the objective function

$$\min_w \; \mathbb{E}\left[\|e^{\beta r(s_t,a_t)}(\bar{V}_\beta)^\gamma(s_{t+1};w) - \bar{V}_\beta(s_t;w)\|^2 \mid s_t\right], a_t \sim \pi_{\theta_t}$$

This is significant in two ways. First, because of the properties explained in Remark 1. Secondly, this implies that (19) uses a modified update rule that complies with the multiplicative Bellman's equation [11].

Remark 4: Note also that simply minimizing the error $\|\beta e^{\beta r(s_t,\pi_{\theta_t})} + \gamma V^{\theta_t}(s_{t+1};w) - V^{\theta_t}(s_t;w_t)\|$ is not equivalent to the above update rule, but it is rather equivalent to scaling the initial rewards r_t to $\beta e^{\beta r_t}$. This would result in the substitution of the product term with a summation in (15), as found in classical RL approaches.

IV. Experiments

To evaluate the effectiveness of the risk-sensitive temporal-difference RL algorithm proposed in Section III-C, we compare it against the standard temporal-difference actor-critic algorithm on two classic RL problems, namely the inverted pendulum (Cart-Pole) and the underactuated double pendulum (Acrobot).

We model the actor and the critic models as single-layer fully connected neural networks of $h = 16$ neurons for Cart-Pole and $h = 64$ neurons for the Acrobot system. ReLU activation functions and Adam optimization updates are used. The objective functions to be optimized are as defined in Section III-C. The best performing learning rates within the set $\{0.0001, 0.0003, 0.0005, 0.0007, 0.001\}$ are used. The same hyper-parameters, e.g., discount factor of $\gamma = 0.99$, are used across all algorithms.

A. Inverted Pendulum (Cart-Pole)

The Cart-Pole problem is the classical inverted pendulum control problem, in which the agent is tasked to balance a pole mounted on a moving cart by an un-actuated joint. The state variable of the cart-pole system has four components $(x, \theta, \dot{x}, \dot{\theta})$, where x and $\dot{x}$ are the position and velocity of the cart on the track, and θ and $\dot{\theta}$ are the angle and angular velocity of the pole with the vertical. The action space consists of an impulsive "left" or "right" force $\{-10, +10\}$N of fixed magnitude to the cart at discrete time intervals. A reward of $r_t = +1$ is given for each time-step t that the pole kept balanced. An episode terminates successfully after $N_t = 200$ time-steps and the average number of time-steps $\hat{N}_t \leq N_t$ (as well as its variance) across different attempts, is used to quantify the performance of the learning algorithm. Failure occurs when $|\theta| > 12^\circ$ or when $|x| > 2.4$m. .

We train the agent for $n_e = 2000$ episodes and test the learned policy in an additional $n_e = 1000$ testing episodes. Average rewards, 90% confidence intervals, and Conditional-Value-at-Risk (CVaR) values at 10% across different runs are shown in Fig. 1. CVaR values with respect to the reward random variable R are defined by:

$$CVaR_p(R) = \mathbb{E}\left[R | R \leq VaR_p(R)\right]$$

where p (here $p = 0.1$) denotes the confidence interval and the Value-at-Risk $VaR_p(R) = \inf\{r \in \mathbb{R} : P(R \leq r) > p\}$ is the p-quantile of the reward.

The training behavior of the risk-neutral algorithm against the proposed risk-sensitive approach for $\beta = +0.001$ and $\beta = -0.001$ are compared in Fig. 1. Both risk-sensitive approaches are able to learn a policy with an average training reward close to $\hat{N}_t \simeq 200$ over independent runs, outperforming the risk-neutral approach. More importantly, the CVaR values (the higher the better) indicate that the risk-sensitive approach yields lower performance variability, which translates to robustness in the testing phase.

In Fig. 2, we train the risk-neutral agent over a total of 5000 steps, which increases its average reward performance. This implies that the risk-sensitive approach is more sample efficient as it converges faster (less observations) to a robust

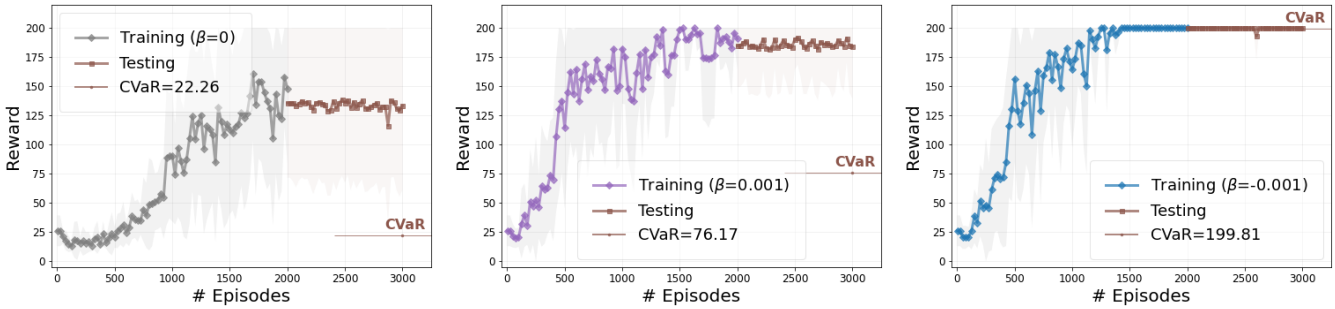

Fig. 1: Training and testing behavior of a risk-neutral agent, a risk-seeking agent with a risk parameter $\beta = 0.001$, and a risk-averse agent with a risk parameter $\beta = -0.001$ in the Cart-Pole problem. Average reward over 10 random runs with 90% confidence intervals are depicted. CVaR values (at 10%) of the policies during testing are also shown.

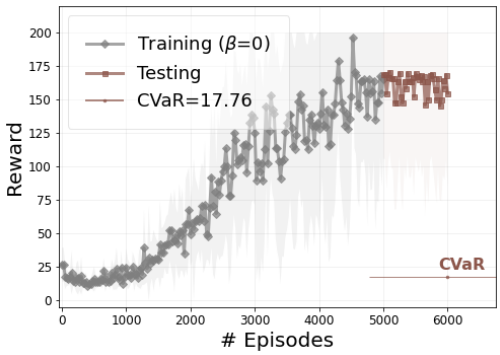

Fig. 2: Training and testing behavior of a risk-neutral agent trained over 5000 episodes. Average reward increases, but CVaR values (at 10%) do not.

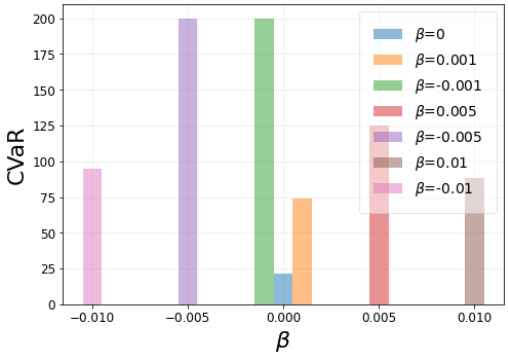

Fig. 3: CVaR (at 10%) values of RL policies for different risk-sensitive parameters β.

policy. Moreover, the CVaR value of the risk-neutral policy appears to stay the same, indicating that reliability of the risk-neutral policy is inherent in the optimization approach and does not necessarily improve with more training.

Finally, in Fig. 3, we compare the performance and robustness (CVaR values) of the proposed risk-sensitive approach for different values of the risk parameter β.

B. Acrobot

The Acrobot problem is a double pendulum, with the joint between the two pendulum links being actuated and the other joint being un-actuated. The state variable of the acrobot system has six components $(\cos\theta_1, \cos\theta_2, \sin\theta_1, \sin\theta_2, \dot{\theta}_1, \dot{\theta}_2,)$, where θ_1 is the angle of the first link with respect to the vertical axis (facing downwards) and θ_2 is the relative angle of the second link with respect to the first link. The action space consists of a torque of $\{-1, 0, +1\}$Nm of fixed magnitude applied to the actuated joint between the two links. A reward of $r_t = -1$ is given for each time-step that the double pendulum has not reached a given height. Note that the reward structure in the Acrobot environment is always negative. An episode is terminated after $N_t = 200$ time-steps when the pendulum has not reached the given height.

We train the agent for $n_e = 2000$ episodes and test the learned policy in an additional $n_e = 1000$ testing episodes. The average rewards and their CVaR values across different runs are shown in Fig. 4 for the risk-neutral algorithm and the proposed risk-sensitive approach for β=0.1 and β=-0.01. Similar to the cart-pole problem, Fig. 4 shows the average reward over 10 random seeds with 90% confidence intervals for the Acrobot environment. CVaR values (at 10%) of the policies during testing are also shown. The risk-averse approach yields better performance in terms of CVaR value of the test runs and converges faster compared to the risk-neutral method. The risk-seeking approach ($\beta > 0$) is still shown to converge faster than the risk-neutral method, but does not outperform it in terms of CVaR values at 10%. Note that, it is not expected that a risk-seeking approach will yield higher CVaR values at 10%. By definition, the risk-seeking behavior aims to maximize for the right tale of the distribution, e.g., the CVaR value at 90%.

These experimental results are consistent with the theory described in Section III and indicate that sufficiently close to zero values of β put a significant enough weight on the importance of maximization of the expected value and lead to learning of a policy with high expected return and reduced variability, which leads to acceleration in learning, sample efficiency, and improved robustness.

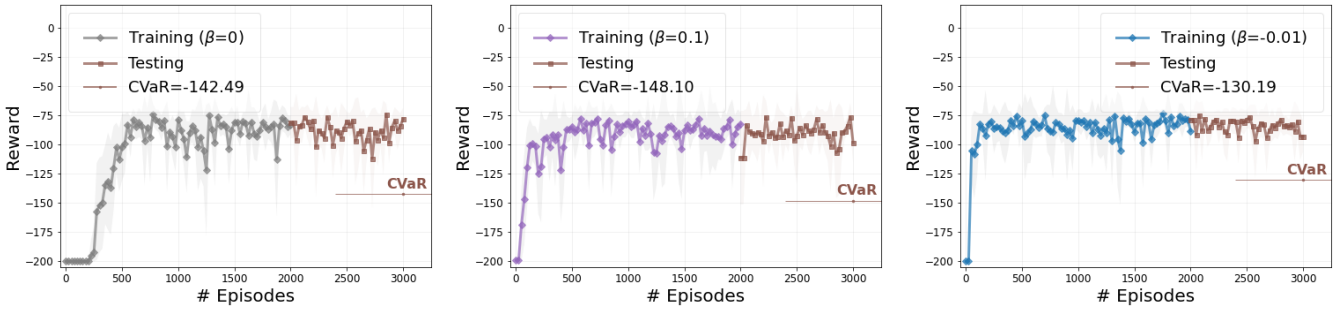

Fig. 4: Training and testing behavior of a risk-neutral agent, a risk-seeking agent with a risk parameter $\beta = 0.1$, and a risk-averse agent with a risk parameter $\beta = -0.01$ in the Acrobot problem. Average reward over 10 random runs with 90% confidence intervals are depicted. CVaR values (at 10%) of the policies during testing are also shown.

V. Conclusion

We develop an actor-critic risk-sensitive reinforcement learning algorithm using the exponential criteria commonly used in risk-sensitive control. The 'actor' implements a policy gradient algorithm based on a function approximation of the exponential of the reward-to-go, estimated by the 'critic' based on the multiplicative Bellman's equation associated with risk-sensitive dynamic programming. Our results suggest that risk-sensitivity can be utilized by actor-critic methods and can accelerate the learning process and reduce its variability. Ongoing work focuses on quantifying the robustness of the proposed approach with respect to disturbances and model perturbations that can lead the way towards robust reinforcement learning algorithms suitable for real-life robotics and cyber-physical systems applications.

References

[1] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT press, 2018.

[2] L. Prashanth, M. C. Fu *et al.*, "Risk-Sensitive Reinforcement Learning via Policy Gradient Search," *Foundations and Trends® in Machine Learning*, vol. 15, no. 5, pp. 537–693, 2022.

[3] L. Pinto, J. Davidson, R. Sukthankar, and A. Gupta, "Robust Adversarial Reinforcement Learning," in *International Conference on Machine Learning*. PMLR, 2017, pp. 2817–2826.

[4] R. J. Williams and J. Peng, "Function Optimization Using Connectionist Reinforcement Learning Algorithms," *Connection Science*, vol. 3, no. 3, pp. 241–268, 1991.

[5] E. Todorov, "Linearly-Solvable Markov Decision Problems," in *Advances in Neural Information Processing Systems*, 2007, pp. 1369–1376.

[6] D. Jacobson, "Optimal Stochastic Linear Systems with Exponential Performance Criteria and Their Relation to Deterministic Differential Games," *IEEE Transactions on Automatic Control*, vol. 18, no. 2, pp. 124–131, 1973.

[7] S. S. Varadhan, *Large deviations and applications*. SIAM, 1984.

[8] M. R. James and J. Baras, "Partially Observed Differential Games, Infinite-Dimensional Hamilton–Jacobi–Isaacs Equations, and Nonlinear H_∞ Control," *SIAM Journal on Control and Optimization*, vol. 34, no. 4, pp. 1342–1364, 1996.

[9] E. Noorani and J. S. Baras, "Embracing Risk in Reinforcement Learning: The Connection between Risk-Sensitive Exponential and Distributionally Robust Criteria," in *2022 American Control Conference (ACC)*, 2022, pp. 2703–2708.

[10] ——, "Risk-Sensitive Reinforcement Learning and Robust Learning for Control," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 2976–2981.

[11] V. S. Borkar and S. P. Meyn, "Risk-sensitive Optimal Control for Markov Decision Processes with Monotone Cost," *Mathematics of Operations Research*, vol. 27, no. 1, pp. 192–209, 2002.

[12] V. Konda and J. Tsitsiklis, "Actor-Critic Algorithms," *Advances in Neural Information Processing Systems*, vol. 12, 1999.

[13] V. S. Borkar, "Q-learning for Risk-Sensitive Control," *Mathematics of Operations Research*, vol. 27, no. 2, pp. 294–311, 2002.

[14] J. S. Baras and M. R. J. , "Robust and Risk-sensitive Output Feedback Control for Finite State Machines and Hidden Markov Models," *Journal of Mathematical Systems, Estimation, and Control*, vol. 7, no. 3, pp. 371–374, 1997.

[15] Y. Zhang, Z. Yang, and Z. Wang, "Provably Efficient Actor-Critic for Risk-Sensitive and Robust Adversarial RL: A Linear-Quadratic Case," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 2764–2772.

[16] E. Noorani and J. S. Baras, "Risk-sensitive REINFORCE: A Monte Carlo Policy Gradient Algorithm for Exponential Performance Criteria," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 1522–1527.

[17] P. La and M. Ghavamzadeh, "Actor-Critic Algorithms for Risk-Sensitive MDPs," *Advances in Neural Information Processing Systems*, vol. 26, 2013.

[18] J. N. Tsitsiklis and B. Van Roy, "An Analysis of Temporal-Difference Learning with Function Approximation," *IEEE Transactions on Automatic Control*, vol. 42, no. 5, pp. 674–690, 1997.

[19] F. L. Lewis, D. Vrabie, and K. G. Vamvoudakis, "Reinforcement Learning and Feedback Control: Using Natural Decision Methods to Design Optimal Adaptive Controllers," *IEEE Control Systems Magazine*, vol. 32, no. 6, pp. 76–105, 2012.

[20] S. Gu, T. Lillicrap, I. Sutskever, and S. Levine, "Continuous Deep Q-Learning with Model-Based Acceleration," in *International Conference on Machine Learning*. PMLR, 2016, pp. 2829–2838.

[21] R. J. Williams, "Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning," *Machine learning*, vol. 8, no. 3, pp. 229–256, 1992.

[22] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy Gradient Methods for Reinforcement Learning with Function Approximation," *Advances in Neural Information Processing Systems*, vol. 12, 1999.

[23] L. Weaver and N. Tao, "The Optimal Reward Baseline for Gradient-Based Reinforcement Learning," *arXiv preprint arXiv:1301.2315*, 2013.

[24] C. N. Mavridis, N. Suriyarachchi, and J. S. Baras, "Maximum-Entropy Progressive State Aggregation for Reinforcement Learning," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 5144–5149.

[25] C. N. Mavridis and J. S. Baras, "Vector Quantization for Adaptive State Aggregation in Reinforcement Learning," in *2021 American Control Conference (ACC)*. IEEE, 2021, pp. 2187–2192.

[26] C. Mavridis and J. Baras, "Annealing optimization for progressive learning with stochastic approximation," *IEEE Transactions on Automatic Control*, 2023.

[27] V. S. Borkar, *Stochastic Approximation: A Dynamical Systems Viewpoint*. Springer, 2009, vol. 48.