Epistemic risk-sensitive reinforcement learning

Downloaded from: https://research.chalmers.se, 2026-09-05 02:08 UTC

Citation for the original published paper (version of record):
Eriksson, H., Dimitrakakis, C. (2020). Epistemic risk-sensitive reinforcement learning. ESANN 2020 - Proceedings, 28th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning: 339-344

N.B. When citing this work, cite the original published paper.

research.chalmers.se offers the possibility of retrieving research publications produced at Chalmers University of Technology. It covers all kind of research output: articles, dissertations, conference papers, reports etc. since 2004. research.chalmers.se is administrated and maintained by Chalmers Library

(article starts on next page)

arXiv:1906.06273v1 [cs.LG] 14 Jun 2019

Epistemic Risk-Sensitive Reinforcement Learning

Hannes Eriksson [1 2] **Christos Dimitrakakis** [1]

Abstract

We develop a framework for interacting with uncertain environments in reinforcement learning (RL) by leveraging preferences in the form of utility functions. We claim that there is value in considering different risk measures during learning. In this framework, the preference for risk can be tuned by variation of the parameter β and the resulting behavior can be risk-averse, risk-neutral or risk-taking depending on the parameter choice. We evaluate our framework for learning problems with model uncertainty. We measure and control for *epistemic* risk using dynamic programming (DP) and policy gradient-based algorithms. The risk-averse behavior is then compared with the behavior of the optimal risk-neutral policy in environments with epistemic risk.

1. Introduction

In this work, we consider the problem of reinforcement learning (RL) for risk-sensitive policies under epistemic uncertainty, i.e. the uncertainty due to the agent not knowing how the environment responds to the agent's actions. This is in contrast to typical approaches in risk-sensitive decision making, which have focused on the aleatory uncertainty due to inherent stochasticity in the environment. Modelling risk-sensitivity in this way makes more sense for applications such as autonomous driving, where systems are nearly deterministic, and most uncertainty is due to the lack of information about the environment. In this paper, we consider epistemic uncertainty and risk sensitivity both within a Bayesian utilitarian framework. We develop novel algorithms for policy optimisation in this setting, and compare their performance quantitatively with policies optimised for the *risk-neutral* setting.

Reinforcement learning (RL) is a sequential decision-making problem under uncertainty. Typically, this is formulated as the maximisation of an expected return, where the return R is defined as the sum of scalar rewards obtained over time $R \triangleq \sum_{t=1}^{T} r_t$ to some potentially infinite or random horizon T. As is common in RL, we assume that agents acting in a discrete Markov decision process (MDP). For any given finite MDP $\mu \in \mathcal{M}$, the optimal risk-neutral policy $\pi^*(\mu) \in \arg\max_\pi \mathbb{E}^\pi_\mu[R]$ be found efficiently via dynamic programming algorithms. Because the learning problem specifies that μ is unknown, the optimal policy under epistemic uncertainty must take into account expected information gain. In the Bayesian setting, we maintain a subjective belief in the form of a probability distribution ξ over MDPs $\mathcal{M}$ and the optimal solution is given by:

$$\max_\pi \mathbb{E}^\pi_\xi[R] = \max_\pi \int_{\mathcal{M}} \mathbb{E}^\pi_\mu[R]\, d\xi(\mu). \tag{1}$$

This problem is generally intractable, as the optimisation is performed over *adaptive* policies, which is exponentially large in the problem horizon. A risk-neutral agent would only wish to maximise expected return. However, we are interested in the case where the agent is risk-sensitive, and in particular with respect to uncertainty about μ.

Contribution. Typically, risk sensitivity in reinforcement learning has addressed risk due to the inherent environment stochasticity (aleatory) and that due to uncertainty (epistemic) with different mechanisms. We instead wish to consider both under a coherent utility maximising framework, where the convexity of the utility with respect to the return R determines whether our behaviour will be risk-seeking or risk-averse. By applying this utility function on the actual return or the expected return given the MDP we can easily trade off between aleatory and epistemic risk-sensitivity.

1.1. Related work

Defining risk sensitivity with respect to the return R can be done in a number of ways. In this paper, we focus on the expected utility formulation, whereby the utility is defined as a function of the return: concave functions lead to risk aversion and convex to risk seeking. In the context of reinforcement learning, this approach was first proposed by Mihatsch & Neuneier (2002), who derived efficient temporal-difference algorithms for exponential utility functions. However, the authors considered risk only with

[1]Department of Computer Science and Engineering, Chalmers University of Technology, Gothenburg, Sweden [2]Zenuity AB, Gothenburg, Sweden. Correspondence to: Hannes Eriksson <hannese@chalmers.se>.

respect to the MDP stochasticity.

Works on Robust MDPs have traditionally dealt with uncertainty due to epistemic risk. In Givan et al. (2000) the authors extend the exact MDP setting by allowing bounded transition probabilities and rewards. They introduce optimal pessimistic and optimistic policy updates for this setting. Mannor et al. (2012) further extend this by allowing the parameter uncertainties to be coupled which allows for less conservative solutions. Tamar et al. (2014) use RL and approximate dynamic programming (ADP) to scale the Robust MDP framework to problems with larger state spaces using function approximators.

Another way of dealing with risk is conditional value-at-risk (CVaR) (Rockafellar et al., 2000; Sivaramakrishnan & Stamicar, 2017) measures. Compared to more traditional risk measures such as Mean-Variance trade-off (Markowitz, 1952; Tamar et al., 2012) or expected utility framework (Friedman & Savage, 1948), CVaR allows us to control for tail risk. CVaR has been used for risk-sensitive MDPs in Chow & Ghavamzadeh (2014); Chow et al. (2015b;a); Stanko (2018).

A Bayesian setting is also considered by Depeweg et al. (2018), who focus on a risk decomposition in aleatory and epistemic components. The authors model the underlying dynamics with a neural network. Under the risk-sensitive framework, they aim to trade-off expected performance with variation in performance. The risk-sensitive criterion they aim to maximise is $\mathbb{E}\big[\sum_t r_t\big] + \beta\sigma\big(\sum_t r_t\big)$ where β controls for the level of risk-aversion. Thus, they are essentially considering risk only with respect to individual rewards. Our paper instead considers the risk with respect to the total return, which we believe is a more interesting setting for long-term planning problems under uncertainty.

2. Optimal policies for epistemic risk

Under the expected utility hypothesis, risk sensitivity can be modelled (Friedman & Savage, 1948) through a concave or convex utility function $U : \mathbb{R} \to \mathbb{R}$ of the return R. Then, for a given model μ, the optimal U-sensitive policy with respect to *aleatory* risk is the solution to $\max_\pi \mathbb{E}^\pi_\mu[U(R)]$. In the case where we are uncertain about what is the true MDP, we can express it through belief ξ over models μ. Then optimal policy is the solution to

$$\pi^A(U,\xi) \triangleq \arg\max_\pi \int_{\mathcal{M}} \mathbb{E}^\pi_\mu[U(R)]\, d\xi(\mu). \tag{2}$$

However, this is only risk-sensitive with respect to the stochasticity of the underlying MDPs. We believe that *epistemic* risk, i.e. the risk due to our uncertainty about the model is more pertinent for reinforcement learning. The optimal *epistemic risk sensitive* policy maximises:

$$\pi^E(U,\xi) \triangleq \arg\max_\pi \int_{\mathcal{M}} U\left(\mathbb{E}^\pi_\xi(R)\right)\, d\xi(\mu). \tag{3}$$

When U is the identity function, both solutions are risk-neutral. In this paper, we shall consider functions of the form $U(x) = \beta^{-1}e^{\beta x}$, so that $\beta > 0$ is risk-seeking and $\beta < 0$ risk-averse.

We consider two algorithms for this problem. The first, based on an approximate dynamic programming algorithm for Bayesian reinforcement learning introduced in (Dimitrakakis, 2011), is introduced in Section 2.2. The second, based on the policy gradient framework (Sutton et al., 2000), allows us to extend the previous algorithm to larger MDPs and allows for learning of stochastic policies. The priors used in the experiments are detailed in Section 3.

2.1. Utility functions.

Previous works (Mihatsch & Neuneier, 2002; Howard & Matheson, 1972) on utility functions for risk-sensitive reinforcement learning used an *exponential* utility function of the form $U(x) = \beta^{-1}e^{\beta x}$. The β parameter could then be used to control the shape of the utility function and determine how risk-sensitive we want to be with respect to x. This is the utility function we will use in this paper.[1] Mihatsch & Neuneier (2002) consider not only exponential utility functions, but a special form of them, that is:

$$\pi^E(U) = \arg\max_\pi \frac{1}{\beta}\log\mathbb{E}\Big(\exp(\beta R)\Big) \tag{4}$$

Mihatsch & Neuneier (2002); Coraluppi & Marcus (1999); Marcus et al. (1997) argue that this utility function has some interesting properties. In particular, maximising leads to maximising:

$$\mathbb{E}[R] + \frac{\beta}{2}Var[R] + \mathcal{O}(\beta^2) \tag{5}$$

From this, we get that the behavior of our policy $\pi^E(U)$ as $\beta \to 0$ is the same as for the risk-neutral case. For $\beta < 0$ we get risk-averse behavior and for $\beta > 0$ risk-taking behavior.

In our work, we are working in a Bayesian setting. Consequently, we introduce a belief ξ over models and define the expected utility of a policy π related to the model uncertainty as follows:

$$U^E_\beta(\xi,\pi) \triangleq \frac{1}{\beta}\log\int_{\mathcal{M}} \exp\left(\beta\mathbb{E}^\pi_\mu[R]\right)d\xi(\mu). \tag{6}$$

[1] Other choices are $U(x) = x^\beta$ or $U(x) = \log(x)$, however they both come with significant drawbacks, such as handling negative returns R.

Algorithm 1 Epistemic Risk Sensitive Backwards Induction

Input: $\mathcal{M}$ (set of MDPs), ξ (current posterior)
repeat
  for $\mu \in \mathcal{M}$ $s \in \mathcal{S}$, $a \in \mathcal{A}$ **do**
    $Q_\mu(s,a) = \mathcal{R}_\mu(s,a) + \gamma \sum_{s'} \mathcal{T}_\mu^{ss'} V_\mu(s')$
  end for
  for $s \in \mathcal{S}$ **do**
    for $a \in \mathcal{A}$ **do**
      $\mathcal{Q}_\xi(s,a) = \sum_\mu \xi(\mu) U[(Q_\mu(s,a)]$
    end for
    $\pi(s) = \arg\max_a \mathcal{Q}_\xi(s,a)$.
    for $\mu \in \mathcal{M}$ **do**
      $V_\mu(s) = Q_\mu(s, \pi(s))$.
    end for
  end for
until *convergence*
return π

2.2. Epistemic risk sensitive backward induction

Algorithm 1 is an Approximate Dynamic Programming (ADP) (Powell, 2007) algorithm for optimising policies in our setting. While the algorithm is given for a belief over a finite set of MDPs, it can be easily extended to arbitrary ξ through simple Monte-Carlo sampling, as in Dimitrakakis (2011).

The algorithm essentially maintains a separate Q_μ-value function for every MDP μ. At every step, it finds the best local policy π with respect to the utility function U. Then the value function V_μ of each MDP reflects the value of π within that MDP.

This algorithm will be used as a baseline for the experiments with risk aversion. We know that as the epistemic uncertainty vanishes; if there is only one underlying true model, then the optimal policy for the epistemic risk-sensitive algorithm will be the same as the optimal policy for the epistemic risk-neutral algorithm. It is important to point out that this is only true when our belief is concentrated around the true MDP. Behavior during learning of the MDP could be very different. For cases with multiple models, the epistemic uncertainty will always exist and there are no such guarantees.

2.3. Bayesian policy gradient

A common method for model-free reinforcement learning is policy gradient (Sutton et al., 2000). It is also very useful in model-based settings, and specifically for the Bayesian reinforcement learning problem, where sampling from the posterior allows us to construct efficient stochastic gradient algorithms. Bayesian policy gradient (BPG) methods have been explored for these contexts before, Ghavamzadeh & Engel (2006) uses BPG to plan how to maximise information gain given our current belief ξ_t for the risk-neutral setting. In our specific setting, we are interested in maximising (3), where we use utility function (6) to model risk sensitivity.

Our choice of policy parametrisation is a softmax policy with non-linear features. The probability of selecting action a in state s, given current parameters θ, is $\pi_\theta(a|s) = \frac{e^{\phi(s,a,\theta)}}{\sum_{a' \in \mathcal{A}} e^{\phi(s,a',\theta)}}$, where the features $\phi(s,a,\theta)$ are calculated by a feedforward neural network with one hidden layer. Full parameter details of the neural networks are given in Section 3.

We choose to introduce a policy parametrisation over (3). This gives us (7).

$$\pi^E(U,\xi,\theta) = \int_{\mathcal{M}} U\left(\mathbb{E}^\mu_{\pi_\theta}[R]\right) d\xi(\mu) \tag{7}$$

Combining our choice of utility function in (6) and (7) gives us our new objective function (8).

$$\pi^E(U,\xi,\theta) = \frac{1}{\beta} \log \int_{\mathcal{M}} \exp\left(\beta \mathbb{E}^\mu_{\pi_\theta}[R]\right) d\xi(\mu) \tag{8}$$

Our goal is now to find the set of parameters θ so as to maximise information gain. Taking the gradient of (8) gives (9) which leads to (10) after a straightforward derivation.

$$\nabla_\theta \pi^E(U,\xi,\theta) = \nabla_\theta \frac{1}{\beta} \log \int_{\mathcal{M}} \exp\left(\beta \mathbb{E}^\mu_{\pi_\theta}[R]\right) d\xi(\mu) \tag{9}$$

$$= \frac{\int_{\mathcal{M}} \exp\left(\beta \mathbb{E}^\mu_{\pi_\theta}[R]\right) \nabla_\theta \mathbb{E}^\mu_{\pi_\theta}[R] d\xi(\mu)}{\int_{\mathcal{M}} \exp\left(\beta \mathbb{E}^\mu_{\pi_\theta}[R]\right) d\xi(\mu)} \tag{10}$$

The RHS of the numerator is the classical policy gradient (Sutton et al., 2000). We can replace the integrals in (10) with a sum by sampling models from our belief ξ. Note that this has to be done independently for the numerator and the denominator to avoid bias. To get each of the separate $\mathbb{E}^\mu_{\pi_\theta}[R]$ we make use of rollouts. Each expected return term is estimated independently with their own set of rollouts.

We now have an estimate for the gradient and can move our policy parameters accordingly to act optimally given our belief ξ_t.

The procedure of calculating the gradient in (10) is given in Algorithm 2. The algorithm builds upon the classic episodical *REINFORCE* algorithm (Williams, 1992). Basing the algorithm on an episodical update makes sense in this case since it decreases the number of rollouts we have to do. One

Algorithm 2 Epistemic Risk Sensitive Policy Gradient

Input: Policy parametrisation θ_t, ξ_t (current posterior).
repeat
  `Simulate to get` θ_{t+1}
  for $i = 1$ **to** N **do**
    $\mu_{(1)}, \mu_{(2)} \sim (\mathcal{M}_t, \mathcal{R}_t)$
    for $j = 1$ **to** M **do**
      $\tau^{(1)}_{\mu_{(1)}}, \tau^{(2)}_{\mu_{(1)}} \sim \pi_\theta, \mu_{(1)}$
      $\tau^{(3)}_{\mu_{(2)}} \sim \pi_\theta, \mu_{(2)}$
    end for
  end for
  $\theta_{t+1} \leftarrow \theta_t - \dfrac{\sum_{i=0}^{N} \exp\left(\beta \tau_{\mu_i}{}^{(1)}\right) \tau_{\mu_i}{}^{(2)} \nabla_\theta \log \pi_\theta(a|s)}{\sum_{i=0}^{N} \exp\left(\beta \tau_{\mu_i}{}^{(3)}\right)}$
  `Deploy` $\pi_{\theta_{t+1}}$
  $\tau \sim \mu, \pi_{\theta_{t+1}}$
  $\xi_{t+1} \leftarrow \xi_t, \tau$
until *convergence*

drawback is that it restricts updates episode by episode and so the information gained in the current episode cannot be used until after the episode ends. Monte-Carlo methods such as *REINFORCE* also have a few other drawbacks such as high variance and low convergence, problems that could be addressed by extending the current framework to an actor-critic based one, as in Barto et al. (1983); Ghavamzadeh & Engel (2007); Ghavamzadeh et al. (2016).

The algorithm consists of two stages. One planning stage which is used to identify the best policy parameters θ_t given our current belief ξ_t. This is done through model-based simulation by sampling MDPs from our belief. After the rollouts have been collected the neural network is optimised and a new policy $\pi_{\theta_{t+1}}$ is attained. We then commit to this policy for one episode and move on to the next stage of the algorithm.

The second stage of the algorithm uses the new policy to act in the real environment. Trajectories $\tau = (s_0, a_0, r_1, s_1, ..., s_{T-1}, a_{T-1}, r_T, s_T)$ are collected from this environment and used to update our belief over transitions and reward functions.

2.4. Bayesian Epistemic CVaR

Another common approach of handling risk-aversion is to optimize with respect to a CVaR objective, (Chow & Ghavamzadeh, 2014; Chow et al., 2015b; Tamar et al., 2015; Stanko, 2018). In this paper, we investigate Bayesian epistemic CVaR, first studied by Chen et al. (2018) in the context of investment strategies. In their case, they use it to do posterior inference of a SV-ALD model to efficiently estimate risk. We define Bayesian epistemic CVaR as follows, defining the set of MDPs where we get at least z utility as the following;

$$\mathcal{M}^\pi_z \triangleq \Big\{ \mu \,|\, \mathbb{E}^\pi_\mu[R] \geq z \Big\}. \tag{11}$$

The $\nu^\pi_\xi(\alpha)$ is value-at-risk in the Bayesian setting for a given quantile α.

$$\nu^\pi_\xi(\alpha) = \inf \Big\{ z \,|\, \xi(\mathcal{M}^\pi_z) \geq \alpha \Big\}. \tag{12}$$

$$C^\pi_\xi(\alpha) = \mathbb{E}^\pi_\xi \Big[\mathbb{E}^\pi_\mu[R] \mid \mathbb{E}^\pi_\mu[R] \leq \nu^\pi_\xi(\alpha) \Big] \tag{13}$$

$$= \int_{\mathcal{M}^\pi_{\nu^\pi_\xi(\alpha)}} \mathbb{E}^\pi_\mu[R] \, d\xi(\mu). \tag{14}$$

Concisely stated; we want to maximize our performance for the α least likely MDPs according to our belief ξ. The intuition behind this is that we want to be risk-averse with respect to what we are the most uncertain about.

3. Experiments

In this paper, we conduct two kinds of experiments. Firstly, a classical learning problem on a *Gridworld*. This is discrete, so we can use Algorithm 1 to find a near-optimal deterministic policy. Similar problems have been studied before in the field of robust MDPs and there are already solutions for finite state and action-space problems. As this is a discrete state-action space, the belief maintained over the MDP transitions is in the form of a Dirichlet-product prior and Belief over rewards is in the form of a NormalGamma prior. Further experimental details about this problem can be found in Section 3.1 and Appendix A.1.

Secondly, we consider a continuous state-space problem in Section 3.2. We do not see a trivial extension of previous works in robust MDPs to the case with infinite state-space. Previous works such as Tamar et al. (2014) scale earlier works on robust MDP to large state-space problems but not to continuous. The problem has been studied to a great extent in the field of risk-aversion, see Tamar et al. (2012; 2015); Chow & Ghavamzadeh (2014) and Appendix A.2. For this problem, we maintain function priors in the form of Gaussian Processes on the reward functions and transition kernels.

3.1. Gridworld experiment

Shown in Figure 1 are the results of Algorithm 1 ran for different values of β. We aimed to test both risk-aversion and risk-taking behavior through the choice of this parameter. The regret is averaged over ≈ 100 independent experiment

Figure 1. Experiment detailing the results of the run of *ADP* Algorithm 1 for varying choices of β. (a) The top plot of the grid is the regret over time with respect to the optimal deterministic risk-neutral policy. (b) The bottom plot shows the distribution of falls throughout over all experiments.

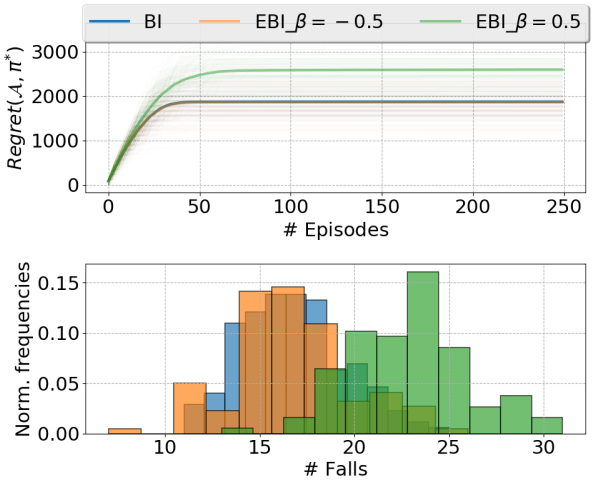
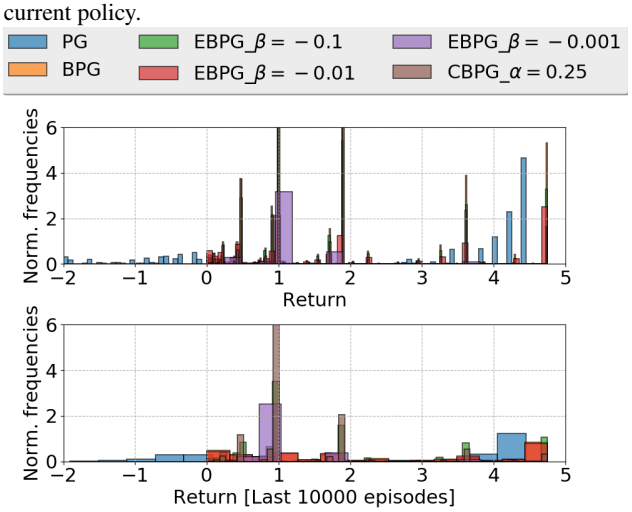


Figure 2. Experiment detailing the results of the run of Epistemic Bayesian Policy gradient *EBPG* Algorithm 2 for varying choices of β. For comparison we also include a risk-neutral *PG*, a risk-neutral *BPG* and *CVaR BPG*. (a) The upper plot is a histogram over returns for all episodes observed. (b) The bottom plot is a histogram of the return over the last 10000 episodes, so the most current policy.



runs. We can see that the regret increases for more risk-taking policies in the top plot. In this environment the risk-neutral and the risk-averse behavior is almost identical, with only a minor difference in $\#Falls$. The risk-taking policy is in one sense, over-exploring while the risk-averse policy in general, would be more inclined to do exploitation. Note that in this experiment the only epistemic uncertainty comes from that the agent does not know the true MDP μ. This uncertainty will go down over time as more transitions are observed.

3.2. Option pricing experiment

The result of runs in this environment is depicted in Figure 2. The blue colored line will serve as a reference and has been verified to have an almost optimal risk-neutral policy. It also has the most observed data (millions of episodes) compared to the other algorithms (hundreds of thousands of episodes). We do see some notable difference in behavior between the policies and can identify that the algorithm with CVaR objective and the algorithm with $\beta = -0.001$ are overly cautious. On the other hand for $\beta = \{-0.01, -0.1\}$ and BPG we see behavior similar to that of regular PG.

4. Conclusions

We have introduced a framework that allows us to control for how risk-sensitive we want to be with respect to model uncertainty in the Bayesian RL setting. In Section 3.1 we detail the performance of Algorithm 1, an ADP algorithm that has the ability to control for risk-sensitiveness in environments with discrete state and action-spaces. The results point towards risk-averse policies lead to less exploratory policies; that is, we would only expect to explore additionally if we are convinced there is much more to be gained by doing so, compared to a risk-neutral policy. Figure 2 shows an extension of this, using Policy gradient algorithms to more complicated environments with continuous state-space.

A few hurdles noted is the speed of Algorithm 2. In order to get a rich representation of the model uncertainty, a lot of MDPs should be sampled. However, sampling MDPs in this setup is quite expensive. Other belief models could be considered instead of GP if we were to scale this up to real problems.

Future work. We can see an adaptive agent that could change its risk-sensitive parameter β with time. For problems where we know uncertainties will resolve later into the future, this could be one approach.

There could be an avenue to explore using utility functions to induce more exploratory behavior in complicated environments. Current methods use concepts such as curiosity (Houthooft et al., 2016; Pathak et al., 2017) for reward shaping or entropy regularization (Williams & Peng, 1991; Mnih et al., 2016; O'Donoghue et al., 2016) to enforce additional exploration. However, under the expected utility framework we could perhaps get this for free by changing

the utility function.

Decreasing variance and speeding up Algorithm 2 is of utmost importance for this framework to be used for more complex problems. As we touched upon in Section 2.3 a straight-forward improvement would be to move towards the Bayesian actor-critic framework introduced in Ghavamzadeh & Engel (2007).

Acknowledgement

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.
The simulations were performed on resources provided by the Swedish National Infrastructure for Computing (SNIC) at Chalmers Centre for Computational Science and Engineering (C3SE).

References

Barto, A. G., Sutton, R. S., and Anderson, C. W. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, (5):834–846, 1983.

Chen, L., Zerilli, P., and Baum, C. F. Leverage effects and stochastic volatility in spot oil returns: A bayesian approach with var and cvar applications. *Energy Economics*, 2018.

Chow, Y. and Ghavamzadeh, M. Algorithms for cvar optimization in mdps. In *Advances in neural information processing systems*, pp. 3509–3517, 2014.

Chow, Y., Ghavamzadeh, M., Janson, L., and Pavone, M. Risk-constrained reinforcement learning with percentile risk criteria, 2015a.

Chow, Y., Tamar, A., Mannor, S., and Pavone, M. Risk-sensitive and robust decision-making: a cvar optimization approach. In *Advances in Neural Information Processing Systems*, pp. 1522–1530, 2015b.

Coraluppi, S. P. and Marcus, S. I. Risk-sensitive and minimax control of discrete-time, finite-state markov decision processes. *Automatica*, 35(2):301–309, 1999.

Depeweg, S., Hernandez-Lobato, J.-M., Doshi-Velez, F., and Udluft, S. Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning. In *International Conference on Machine Learning*, pp. 1192–1201, 2018.

Dimitrakakis, C. Robust bayesian reinforcement learning through tight lower bounds. In *European Workshop on Reinforcement Learning (EWRL 2011)*, pp. 177–188, 2011.

Friedman, M. and Savage, L. J. The Utility Analysis of Choices Involving Risk. *The Journal of Political Economy*, 56(4):279, 1948.

Ghavamzadeh, M. and Engel, Y. Bayesian policy gradient algorithms. In *NIPS 2006*, 2006.

Ghavamzadeh, M. and Engel, Y. Bayesian actor-critic algorithms. In *Proceedings of the 24th international conference on Machine learning*, pp. 297–304. ACM, 2007.

Ghavamzadeh, M., Engel, Y., and Valko, M. Bayesian policy gradient and actor-critic algorithms. *The Journal of Machine Learning Research*, 17(1):2319–2371, 2016.

Givan, R., Leach, S., and Dean, T. Bounded-parameter markov decision processes. *Artificial Intelligence*, 122 (1-2):71–109, 2000.

Houthooft, R., Chen, X., Duan, Y., Schulman, J., De Turck, F., and Abbeel, P. Vime: Variational information maximizing exploration. In *Advances in Neural Information Processing Systems*, pp. 1109–1117, 2016.

Howard, R. A. and Matheson, J. E. Risk-sensitive markov decision processes. *Management science*, 18(7):356–369, 1972.

Leike, J., Martic, M., Krakovna, V., Ortega, P. A., Everitt, T., Lefrancq, A., Orseau, L., and Legg, S. Ai safety gridworlds. *arXiv preprint arXiv:1711.09883*, 2017.

Mannor, S., Mebel, O., and Xu, H. Lightning does not strike twice: Robust mdps with coupled uncertainty. *arXiv preprint arXiv:1206.4643*, 2012.

Marcus, S. I., Fernández-Gaucherand, E., Hernández-Hernandez, D., Coraluppi, S., and Fard, P. Risk sensitive markov decision processes. In *Systems and control in the twenty-first century*, pp. 263–279. Springer, 1997.

Markowitz, H. Portfolio selection. *The journal of finance*, 7 (1):77–91, 1952.

Mihatsch, O. and Neuneier, R. Risk-sensitive reinforcement learning. *Machine learning*, 49(2-3):267–290, 2002.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937, 2016.

Murphy, K. Conjugate bayesian analysis of the gaussian distribution. Technical report, UBC, 2007.

O'Donoghue, B., Munos, R., Kavukcuoglu, K., and Mnih, V. Combining policy gradient and q-learning. *arXiv preprint arXiv:1611.01626*, 2016.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 16–17, 2017.

Powell, W. B. *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons, 2007.

Rockafellar, R. T., Uryasev, S., et al. Optimization of conditional value-at-risk. *Journal of risk*, 2:21–42, 2000.

Sivaramakrishnan, K. and Stamicar, R. A cvar scenario-based framework: Minimizing downside risk of multi-asset class portfolios. *The Journal of Portfolio Management*, 44:114–129, 12 2017. doi: 10.3905/jpm.2018.44.2.114.

Stanko, S. Risk-averse distributional reinforcement learning, 2018.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pp. 1057–1063, 2000.

Tamar, A., Di Castro, D., and Mannor, S. Policy gradients with variance related risk criteria. In *Proceedings of the twenty-ninth international conference on machine learning*, pp. 387–396, 2012.

Tamar, A., Mannor, S., and Xu, H. Scaling up robust mdps using function approximation. In *International Conference on Machine Learning*, pp. 181–189, 2014.

Tamar, A., Glassner, Y., and Mannor, S. Optimizing the cvar via sampling. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Williams, R. J. and Peng, J. Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 3(3):241–268, 1991.

A. Environment descriptions

A.1. Gridworld

A visual representation of the environment can be seen in Leike et al. (2017). The experiments are run with slightly different parameters, which are the following; Let $\gamma = 0.99$ be the discount factor, and $T = 100$ be the maximum allowed steps within an episode. The allowed actions correspond to the four cardinal directions $\{North, East, South, West\}$. The environment consists of a *GOAL* tile, which terminates the episode and gives a reward of $r_g = 0.0$. The environment also has *WATER* tiles which, upon entering, terminates the episode and gives a reward of $r_w = -10.0$. Every other action in any other state gives a reward of $r_0 = -0.1$. To ensure there is always epistemic uncertainty, we introduce another almost identical environment but with a goal at another location.

We measure performance in *Regret* with respect to the optimal deterministic risk-neutral policy. The regret of an algorithm $\mathcal{A}$ with respect to the optimal deterministic risk-neutral policy is given by the following;

$$Regret(\mathcal{A}, \pi^*) = \int_{\mathcal{M}} \left(V_\mu^{\pi^*} - \sum_{t=0}^{T} \gamma^t r_t \right) d\mu. \tag{15}$$

We let our prior belief over the MDP transitions be $\mathcal{T}_\mu^0(s'|s,a) = Dir(\alpha_0), \alpha_0 = 0.5$. We can then, given observed transitions in the environment, update $\mathcal{T}_\mu^T(s'|s,a) = Dir(\frac{\alpha_0 + n_{s,a}^{s'}}{\sum_{s''} n_{s,a}^{s''}})$, that is, we count the number of transitions $(s,a) \to s'$ and $(s,a) \to \cdot$. From these counts we can compute α_t which is our posterior on the transition $(s,a) \to s'$ at time t.

We also hold a belief over the reward matrices $\mathcal{R}_\mu(s,a)$. A common prior is to use is a $NormalGamma$-prior (Murphy, 2007) over each $r_\mu(s,a)$. The NormalGamma distribution is the conjugate prior to the $Normal$ distribution with unknown parameters μ, λ. The model is given as $p(\mu, \lambda|D) = NG(\mu, \lambda|\mu_n, \kappa_n, \alpha_n, \beta_n)$. So our belief over rewards $\mathcal{R}_\mu(s,a) = NG(\mu, \lambda|\mu_n, \kappa_n, \alpha_n, \beta_n)$.

A.2. Options

A common experiment (Chow & Ghavamzadeh, 2014; Tamar et al., 2012), used to test risk-averse algorithms in a continuous environment is the *Option Pricing* experiment. It can have many forms and classically we aim to minimise expected cost. In this work, we consider the reward perspective instead and the goal is to maximise the expected return. The environment is as follows; At any time t the agent represents an investor with the opportunity of buying an option which gives an immediate reward of r_t. The reward of this option varies with time and with probability p, $r_{t+1} \leftarrow \gamma f_u r_t$, and with probability $(1-p)$, $r_{t+1} \leftarrow \gamma f_d r_t$. The goal is then for the agent to chose a suitable time to stop. Upon choosing to buy the option, the episode ends and the agent returns to x_0. Every time step the agent decides to wait will also incur a small negative reward p_h which represents the opportunity cost for not using its resources. If the agent chooses to wait until the horizon T, it is then forced to take the option for its current value.

We use the following parameters in our experiment; $x_0 = [1; 0], p_h = -0.1, T = 20, \gamma = 0.95, f_u = 2.0, f_d = 0.5, p = [0.45; 0.65; 0.85], K = 5.0$.

Algorithm 2 requires us to be able to sample MDPs from the environment and to be able to do rollouts in them. There are no obvious candidates of priors over continuous state-space transitions and reward functions. In this work, we chose to use Gaussian Processes as priors over functions. We use multiple Gaussian Processes to maintain our belief over transition kernels and reward functions. We update our belief with transitions and reward from the true underlying models. We use a Dirichlet prior over models and update it when we receive information on which model we acted in.

Given data from rollouts sampled from our belief, we can use any standard Policy gradient-like algorithm to update our policy π_{θ_t}. In this paper, we chose to focus on an episodic Policy gradient algorithm similar to the *REINFORCE* framework. We parametrise our policy by a one hidden layer neural network with parameters θ.