Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence

Optimizing the CVaR via Sampling

Aviv Tamar, Yonatan Glassner, and **Shie Mannor**
Electrical Engineering Department
The Technion - Israel Institute of Technology
Haifa, Israel 32000
{avivt, yglasner}@tx.technion.ac.il, shie@ee.technion.ac.il

Abstract

Conditional Value at Risk (CVaR) is a prominent risk measure that is being used extensively in various domains. We develop a new formula for the gradient of the CVaR in the form of a conditional expectation. Based on this formula, we propose a novel sampling-based estimator for the gradient of the CVaR, in the spirit of the likelihood-ratio method. We analyze the bias of the estimator, and prove the convergence of a corresponding stochastic gradient descent algorithm to a local CVaR optimum. Our method allows to consider CVaR optimization in new domains. As an example, we consider a reinforcement learning application, and learn a risk-sensitive controller for the game of Tetris.

1 Introduction

Conditional Value at Risk (CVaR; Rockafellar and Uryasev, 2000) is an established risk measure that has found extensive use in finance among other fields. For a random payoff R, whose distribution is parameterized by a controllable parameter θ, the α-CVaR is defined as the expected payoff over the $\alpha\%$ worst outcomes of Z:

$$\Phi(\theta) = \mathbb{E}^{\theta}\left[R | \, R \leq \nu_{\alpha}(\theta)\right],$$

where $\nu_{\alpha}(\theta)$ is the α-quantile of R. CVaR optimization aims to find a parameter θ that maximizes $\Phi(\theta)$.

When the payoff is of the structure $R = f_{\theta}(X)$, where f_{θ} is a deterministic function, and X is random but does not depend on θ, CVaR optimization may be formulated as a stochastic program, and solved using various approaches (Rockafellar and Uryasev 2000; Hong and Liu 2009; Iyengar and Ma 2013). Such a payoff structure is appropriate for certain domains, such as portfolio optimization, in which the investment strategy generally does not affect the asset prices. However, in many important domains, for example queueing systems, resource allocation, and reinforcement learning, the tunable parameters also control the *distribution* of the random outcomes. Since existing CVaR optimization methods are not suitable for such cases, and due to increased interest in risk-sensitive optimization recently in these domains (Tamar, Di Castro, and Mannor 2012; Prashanth and Ghavamzadeh 2013), there is a strong incentive to develop more general CVaR optimization algorithms.

In this work, we propose a CVaR optimization approach that is applicable when θ also controls the distribution of X. The basis of our approach is a new formula that we derive for the CVaR gradient $\frac{\partial \Phi(\theta)}{\partial \theta}$ in the form of a conditional expectation. Based on this formula, we propose a sampling-based estimator for the CVaR gradient, and use it to optimize the CVaR by stochastic gradient descent.

In addition, we analyze the bias of our estimator, and use the result to prove convergence of the stochastic gradient descent algorithm to a local CVaR optimum. Our method allows us to consider CVaR optimization in new domains. As an example, we consider a reinforcement learning application, and learn a risk-sensitive controller for the game of Tetris. To our knowledge, CVaR optimization for such a domain is beyond the reach of existing approaches. Considering Tetris also allows us to easily interpret our results, and show that we indeed learn sensible policies.

We remark that in certain domains, CVaR is often not maximized directly, but used as a constraint in an optimization problem of the form $\max_{\theta} \mathbb{E}^{\theta}[R]$ s.t. $\Phi(\theta) \geq b$. Extending our approach to such problems is straightforward, using standard penalty method techniques (see, e.g., Tamar, Di Castro, and Mannor, 2012, and Prashanth and Ghavamzadeh, 2013, for a such an approach with a variance-constrained objective), since the key component for these methods is the CVaR gradient estimator we provide here. Another appealing property of our estimator is that it naturally incorporates importance sampling, which is important when α is small, and the CVaR captures *rare* events.

Related Work Our approach is similar in spirit to the *likelihood-ratio* method (LR; Glynn, 1990), that estimates the gradient of the *expected* payoff. The LR method has been successfully applied in diverse domains such as queueing systems, inventory management, and financial engineering (Fu 2006), and also in reinforcement learning (RL; Sutton and Barto, 1998), where it is commonly known as the *policy gradient* method (Baxter and Bartlett 2001; Peters and Schaal 2008). Our work extends the LR method to estimating the gradient of the CVaR of the payoff.

Closely related to our work are the studies of Hong and Liu (2009) and Scaillet (2004), who proposed perturbation

analysis style estimators for the gradient of the CVaR, for the setting mentioned above, in which θ *does not affect* the distribution of X. Indeed, their gradient formulae are different than ours, and do not apply in our setting.

LR gradient estimators for other risk measures have been proposed by Borkar (2001) for exponential utility functions, and by Tamar, Di Castro, and Mannor (2012) for mean–variance. These measures, however, consider a very different notion of risk than the CVaR. For example, the mean–variance measure is known to underestimate the risk of rare, but catastrophic events (Agarwal and Naik 2004).

Risk-sensitive optimization in RL is receiving increased interest recently. A mean-variance criterion was considered by Tamar, Di Castro, and Mannor (2012) and Prashanth and Ghavamzadeh (2013). Morimura et al. (2010) consider the expected return, with a CVaR based risk-sensitive policy for guiding the exploration while learning. Their method, however, does not scale to large problems. Borkar and Jain (2014) optimize a CVaR constrained objective using dynamic programming, by augmenting the state space with the accumulated reward. As such, that method is only suitable for a finite horizon and a small state-space, and *does not scale-up* to problems such as the Tetris domain we consider. A function approximation extension of (Borkar and Jain 2014) is mentioned, using a three time scales stochastic approximation algorithm. In that work, three different learning rates are decreased to 0, and convergence is determined by the slowest one, leading to an overall slow convergence. In contrast, our approach requires only a single learning rate. Recently, Prashanth (2014) used our gradient formula of Proposition 2 (from a preliminary version of this paper) in a two time-scale stochastic approximation scheme to show convergence of CVaR optimization. Besides providing the theoretical basis for that work, our current convergence result (Theorem 5) obviates the need for the extra time-scale, and results in a simpler and faster algorithm.

2 A CVaR Gradient Formula

In this section we present a new LR-style formula for the gradient of the CVaR. This gradient will be used in subsequent sections to optimize the CVaR with respect to some parametric family. We start with a formal definition of the CVaR, and then present a CVaR gradient formula for 1-dimensional random variables. We then extend our result to the multi-dimensional case.

Let Z denote a random variable with a cumulative distribution function (C.D.F.) $F_Z(z) = \Pr(Z \leq z)$. For convenience, we assume that Z is a continuous random variable, meaning that $F_Z(z)$ is everywhere continuous. We also assume that Z is bounded. Given a confidence level $\alpha \in (0,1)$, the α-Value-at-Risk, (VaR; or α-quantile) of Z is denoted $\nu_\alpha(Z)$, and given by

$$\nu_\alpha(Z) = F_Z^{-1}(\alpha) \doteq \inf\{z : F_Z(z) \geq \alpha\}. \quad (1)$$

The α-Conditional-Value-at-Risk of Z is denoted by $\Phi_\alpha(Z)$ and defined as the expectation of the α fraction of the worst outcomes of Z

$$\Phi_\alpha(Z) = \mathbb{E}[Z | Z \leq \nu_\alpha(Z)]. \quad (2)$$

We next present a formula for the sensitivity of $\Phi_\alpha(Z)$ to changes in $F_Z(z)$.

2.1 CVaR Gradient of a 1-Dimensional Variable

Consider again a random variable Z, but now let its probability density function (P.D.F.) $f_Z(z;\theta)$ be parameterized by a vector $\theta \in \mathbb{R}^k$. We let $\nu_\alpha(Z;\theta)$ and $\Phi_\alpha(Z;\theta)$ denote the VaR and CVaR of Z as defined in Eq. (1) and (2), when the parameter is θ, respectively.

We are interested in the sensitivity of the CVaR to the parameter vector, as expressed by the gradient $\frac{\partial}{\partial\theta_j}\Phi_\alpha(Z;\theta)$. In all but the most simple cases, calculating the gradient analytically is intractable. Therefore, we derive a formula in which $\frac{\partial}{\partial\theta_j}\Phi_\alpha(Z;\theta)$ is expressed as a conditional expectation, and use it to calculate the gradient by *sampling*. For technical convenience, we make the following assumption:

Assumption 1. *Z is a continuous random variable, and bounded in $[-b,b]$ for all θ.*

We also make the following smoothness assumption on $\nu_\alpha(Z;\theta)$ and $\Phi_\alpha(Z;\theta)$

Assumption 2. *For all θ and $1 \leq j \leq k$, the gradients $\frac{\partial\nu_\alpha(Z;\theta)}{\partial\theta_j}$ and $\frac{\partial\Phi_\alpha(Z;\theta)}{\partial\theta_j}$ exist and are bounded.*

Note that since Z is continuous, Assumption 2 is satisfied whenever $\frac{\partial}{\partial\theta_j}f_Z(z;\theta)$ is bounded. Relaxing Assumptions 1 and 2 is possible, but involves technical details that would complicate the presentation, and is left to future work. The next assumption is standard in LR gradient estimates

Assumption 3. *For all θ, z, and $1 \leq j \leq k$, we have that $\frac{\partial f_Z(z;\theta)}{\partial\theta_j} / f_Z(z;\theta)$ exists and is bounded.*

In the next proposition we present a LR-style sensitivity formula for $\Phi_\alpha(Z;\theta)$, in which the gradient is expressed as a conditional expectation. In Section 3 we shall use this formula to suggest a sampling algorithm for the gradient.

Proposition 1. *Let Assumptions 1, 2, and 3 hold. Then*

$$\frac{\partial\Phi_\alpha(Z;\theta)}{\partial\theta_j} = \mathbb{E}^\theta\left[\frac{\partial\log f_Z(Z;\theta)}{\partial\theta_j}(Z-\nu_\alpha(Z;\theta))\middle| Z \leq \nu_\alpha(Z;\theta)\right].$$

Proof. Define the level-set $D_\theta = \{z \in [-b,b] : z \leq \nu_\alpha(Z;\theta)\}$. By definition, $D_\theta \equiv [-b, \nu_\alpha(Z;\theta)]$, and $\int_{z\in D_\theta} f_Z(z;\theta)\,dz = \alpha$. Taking a derivative and using the Leibniz rule we obtain

$$0 = \frac{\partial}{\partial\theta_j}\int_{-b}^{\nu_\alpha(Z;\theta)} f_Z(z;\theta)\,dz \quad (3)$$
$$= \int_{-b}^{\nu_\alpha(Z;\theta)} \frac{\partial f_Z(z;\theta)}{\partial\theta_j}dz + \frac{\partial\nu_\alpha(Z;\theta)}{\partial\theta_j} f_Z(\nu_\alpha(Z;\theta);\theta).$$

By definition (2) we have $\Phi_\alpha(Z;\theta) = \int_{z\in D_\theta}\frac{f_Z(z;\theta)z}{\alpha}dz = \alpha^{-1}\int_{-b}^{\nu_\alpha(Z;\theta)} f_Z(z;\theta)\,z dz$. Now, taking a derivative and using the Leibniz rule we obtain

$$\frac{\partial}{\partial\theta_j}\Phi_\alpha(Z;\theta) = \alpha^{-1}\int_{-b}^{\nu_\alpha(Z;\theta)}\frac{\partial f_Z(z;\theta)}{\partial\theta_j}z dz \quad (4)$$
$$+\alpha^{-1}\frac{\partial\nu_\alpha(Z;\theta)}{\partial\theta_j} f_Z(\nu_\alpha(Z;\theta);\theta)\,\nu_\alpha(Z;\theta).$$

Rearranging, and plugging (3) in (4) we obtain $\frac{\partial}{\partial\theta_j}\Phi_\alpha(Z;\theta) = \alpha^{-1}\int_{-b}^{\nu_\alpha(Z;\theta)}\frac{\partial f_Z(z;\theta)}{\partial\theta_j}\left(z-\nu_\alpha(Z;\theta)\right)dz$. Finally, using the likelihood ratio trick – multiplying and dividing by $f_Z(z;\theta)$ inside the integral, which is justified due to Assumption 3, we obtain the required expectation. □

Let us contrast the CVaR LR formula of Proposition 1 with the standard LR formula for the expectation (Glynn 1990) $\frac{\partial}{\partial\theta_j}\mathbb{E}^\theta[Z] = \mathbb{E}^\theta\left[\frac{\partial\log f_Z(Z;\theta)}{\partial\theta_j}(Z-b)\right]$, where the baseline b could be any arbitrary constant. Note that in the CVaR case the baseline is *specific*, and, as seen in the proof, accounts for the sensitivity of the level-set D_θ. Quite surprisingly, this specific baseline turns out to be exactly the VaR, $\nu_\alpha(Z;\theta)$, which, as we shall see later, also leads to an elegant sampling based estimator.

In a typical application, Z would correspond to the performance of some system, such as the profit in portfolio optimization, or the total reward in RL. Note that in order to use Proposition 1 in a gradient estimation algorithm, one needs access to $\frac{\partial}{\partial\theta_j}\log f_Z(Z;\theta)$: the sensitivity of the performance distribution to the parameters. Typically, the system performance is a complicated function of a high-dimensional random variable. For example, in RL and queueing systems, the performance is a function of a trajectory from a stochastic dynamical system, and calculating its probability distribution is usually intractable. The sensitivity of the trajectory distribution to the parameters, however, is often easy to calculate, since the parameters typically control how the trajectory is generated. We shall now generalize Proposition 1 to such cases. The utility of this generalization is further exemplified in Section 5, for the RL domain.

2.2 CVaR Gradient Formula – General Case

Let $\mathbf{X} = (X_1, X_2, \ldots, X_n)$ denote an $n-$dimensional random variable with a finite support $[-b,b]^n$, and let Y denote a discrete random variable taking values in some countable set $\mathcal{Y}$. Let $f_Y(y;\theta)$ denote the probability mass function of Y, and let $f_{\mathbf{X}|Y}(\mathbf{x}|y;\theta)$ denote the probability density function of $\mathbf{X}$ given Y. Let the reward function r be a bounded mapping from $[-b,b]^n\times\mathcal{Y}$ to $\mathbb{R}$, and consider the random variable $R \doteq r(\mathbf{X},Y)$. We are interested in a formula for $\frac{\partial}{\partial\theta_j}\Phi_\alpha(R;\theta)$.

We make the following assumption, similar to Assumptions 1, 2, and 3.

Assumption 4. *The reward R is a continuous random variable for all θ. Furthermore, for all θ and $1\le j\le k$, the gradients $\frac{\partial}{\partial\theta_j}\nu_\alpha(R;\theta)$ and $\frac{\partial}{\partial\theta_j}\Phi_\alpha(R;\theta)$ are well defined and bounded. In addition $\frac{\partial\log f_{\mathbf{X}|Y}(\mathbf{x}|y;\theta)}{\partial\theta_j}$ and $\frac{\partial\log f_Y(y;\theta)}{\partial\theta_j}$ exist and are bounded for all $\mathbf{x}$, y, and θ.*

Define the level-set $\mathcal{D}_{y;\theta} = \{\mathbf{x}\in[-b,b]^n : r(\mathbf{x},y)\le\nu_\alpha(R;\theta)\}$. We require some smoothness of the function r, that is captured by the following assumption on $\mathcal{D}_{y;\theta}$.

Assumption 5. *For all y and θ, the set $\mathcal{D}_{y;\theta}$ may be written as a finite sum of $L_{y;\theta}$ disjoint, closed, and connected components $D^i_{y;\theta}$, each with positive measure: $\mathcal{D}_{y;\theta} = \sum_{i=1}^{L_{y;\theta}} D^i_{y;\theta}$.*

Assumption 5 may satisfied, for example, when $r(\mathbf{x},y)$ is Lipschitz in $\mathbf{x}$ for all $y\in\mathcal{Y}$. We now present a sensitivity formula for $\Phi_\alpha(R;\theta)$.

Proposition 2. *Let Assumption 4 and 5 hold. Then*

$$\frac{\partial}{\partial\theta_j}\Phi_\alpha(R;\theta) = \mathbb{E}^\theta\left[\left(\frac{\partial\log f_Y(Y;\theta)}{\partial\theta_j} + \frac{\partial\log f_{\mathbf{X}|Y}(\mathbf{X}|Y;\theta)}{\partial\theta_j}\right)\left(R-\nu_\alpha(R;\theta)\right)\middle| R\le\nu_\alpha(R;\theta)\right].$$

The proof of Proposition 2 is similar in spirit to the proof of Proposition 1, but involves some additional difficulties of applying the Leibnitz rule in a multidimensional setting. It is given in (Tamar, Glassner, and Mannor 2014). We reiterate that relaxing Assumptions 4 and 5 is possible, but is technically involved, and left for future work. In the next section we show that the formula in Proposition 2 leads to an effective algorithm for estimating $\frac{\partial}{\partial\theta_j}\Phi_\alpha(R;\theta)$ by sampling.

3 A CVaR Gradient Estimation Algorithm

The sensitivity formula in Proposition 2 suggests a natural Monte–Carlo (MC) estimation algorithm. The method, which we label `GCVaR` (Gradient estimator for CVaR), is described as follows. Let $\mathbf{x}_1,y_1\ldots,\mathbf{x}_N,y_N$ be N samples drawn i.i.d. from $f_{\mathbf{X},Y}(\mathbf{x},y;\theta)$, the joint distribution of $\mathbf{X}$ and Y. We first estimate $\nu_\alpha(R;\theta)$ using the empirical α-quantile[1] $\tilde{v}$

$$\tilde{v} = \inf_z \hat{F}(z)\ge\alpha, \tag{5}$$

where $\hat{F}(z)$ is the empirical C.D.F. of R: $\hat{F}(z) \doteq \frac{1}{N}\sum_{i=1}^N \mathbf{1}_{r(\mathbf{x}_i,y_i)\le z}$. The MC estimate of the gradient $\Delta_{j;N}\approx\frac{\partial}{\partial\theta_j}\Phi_\alpha(R;\theta)$ is given by

$$\Delta_{j;N} = \frac{1}{\alpha N}\sum_{i=1}^N\left(\frac{\partial\log f_Y(y_i;\theta)}{\partial\theta_j}+\frac{\partial\log f_{\mathbf{X}|Y}(\mathbf{x}_i|y_i;\theta)}{\partial\theta_j}\right)\times \times\left(r(\mathbf{x}_i,y_i)-\tilde{v}\right)\mathbf{1}_{r(\mathbf{x}_i,y_i)\le\tilde{v}}. \tag{6}$$

It is known that the empirical α-quantile is a biased estimator of $\nu_\alpha(R;\theta)$. Therefore, $\Delta_{j;N}$ is also a biased estimator of $\frac{\partial}{\partial\theta_j}\Phi_\alpha(R;\theta)$. In the following we analyze and bound this bias. We first show that $\Delta_{j;N}$ is a consistent estimator. The proof is similar to the proof of Theorem 4.1 in (Hong and Liu 2009), and given in (Tamar, Glassner, and Mannor 2014).

Theorem 3. *Let Assumption 4 and 5 hold. Then $\Delta_{j;N}\to\frac{\partial}{\partial\theta_j}\Phi_\alpha(R;\theta)$ w.p. 1 as $N\to\infty$.*

With an additional smoothness assumption we can explicitly bound the bias. Let $f_R(\cdot;\theta)$ denote the P.D.F. of R, and define the function $g(\beta;\theta) \doteq \mathbb{E}^\theta\left[\left(\frac{\partial\log f_Y(Y;\theta)}{\partial\theta_j}+\frac{\partial\log f_{\mathbf{X}|Y}(\mathbf{X}|Y;\theta)}{\partial\theta_j}\right)(R-\nu_\alpha(R;\theta))\middle| R=\beta\right]$.

[1] Algorithmically, this is equivalent to first sorting the $r(\mathbf{x}_i,y_i)$'s in ascending order, and then selecting $\tilde{v}$ as the $\lceil\alpha N\rceil$ term in the sorted list.

Algorithm 1 GCVaR

1: **Given:**

- CVaR level α
- A reward function $r(\mathbf{x}, y) : \mathbb{R}^n \times \mathcal{Y} \to \mathbb{R}$
- Derivatives $\frac{\partial}{\partial \theta_j}$ of the probability mass function $f_Y(y;\theta)$ and probability density function $f_{\mathbf{X}|Y}(\mathbf{x}|y;\theta)$
- An i.i.d. sequence $\mathbf{x}_1, y_1, \ldots, \mathbf{x}_N, y_N \sim f_{\mathbf{X},Y}(\mathbf{x}, y; \theta)$.

2: Set $r_1^s, \ldots, r_N^s = \text{Sort}\left(r(\mathbf{x}_1, y_1), \ldots, r(\mathbf{x}_N, y_N)\right)$
3: Set $\tilde{v} = r^s_{\lceil \alpha N \rceil}$
4: For $j = 1, \ldots, k$ do

$$\Delta_{j;N} = \frac{1}{\alpha N} \sum_{i=1}^{N} \left(\frac{\partial \log f_Y(y_i;\theta)}{\partial \theta_j} + \frac{\partial \log f_{\mathbf{X}|Y}(\mathbf{x}_i|y_i;\theta)}{\partial \theta_j} \right) \times \\ \times \left(r(\mathbf{x}_i, y_i) - \tilde{v}\right) \mathbf{1}_{r(\mathbf{x}_i, y_i) \le \tilde{v}}$$

5: **Return:** $\Delta_{1;N}, \ldots, \Delta_{k;N}$

Assumption 6. *For all θ, $f_R(\cdot;\theta)$ and $g(\cdot;\theta)$ are continuous at $\nu_\alpha(R;\theta)$, and $f_R(\nu_\alpha(R;\theta);\theta) > 0$.*

Assumption 6 is similar to Assumption 4 of (Hong and Liu 2009), and may be satisfied, for example, when $\frac{\partial \log f_{\mathbf{X}|Y}(\mathbf{x}|y;\theta)}{\partial \theta_j}$ is continuous and $r(\mathbf{x}, y)$ is Lipschitz in $\mathbf{x}$. The next theorem shows that the bias is $\mathcal{O}(N^{-1/2})$. The proof, given in (Tamar, Glassner, and Mannor 2014), is based on separating the bias to a term that is bounded using a result of Hong and Liu (2009), and an additional term that is bounded using well-known results for the bias of empirical quantiles.

Theorem 4. *Let Assumptions 4, 5, and 6 hold. Then $\mathbb{E}[\Delta_{j;N}] - \frac{\partial}{\partial \theta_j} \Phi_\alpha(R;\theta)$ is $O(N^{-1/2})$.*

At this point, let us again contrast GCVaR with the standard LR method. One may naively presume that applying a standard LR gradient estimator to the $\alpha\%$ worst samples would work as a CVaR gradient estimator. This corresponds to applying the GCVaR algorithm without subtracting the $\tilde{v}$ baseline from the reward in (6). Theorems 3 and 4 show that such an estimator *would not be consistent*. In fact, in (Tamar, Glassner, and Mannor 2014) we give an example where the gradient error of such an approach may be arbitrarily large.

In the sequel, we use GCVaR as part of a stochastic gradient descent algorithm for CVaR optimization. An asymptotically decreasing gradient bias, as may be established from Theorem 3, is necessary to guarantee convergence of such a procedure. Furthermore, the bound of Theorem 4 will allow us to quantify how many samples are needed at each iteration for such convergence to hold.

Variance Reduction by Importance Sampling

For very low quantiles, i.e., α close to 0, the GCVaR estimator would suffer from a high variance, since the averaging is effectively only over αN samples. This is a well-known issue in sampling based approaches to VaR and CVaR estimation, and is often mitigated using variance reduction techniques such as Importance Sampling (IS; Rubinstein and Kroese, 2011; Bardou, Frikha, and Pagès, 2009). In IS, the variance of a MC estimator is reduced by using samples from a *different* sampling distribution, and suitably modifying the estimator to keep it unbiased. It is straightforward to incorporate IS into LR gradient estimators in general, and to our GCVaR estimator in particular. Due to space constraints, and since this is fairly standard textbook material (e.g., Rubinstein and Kroese, 2011), we provide the full technical details in (Tamar, Glassner, and Mannor 2014). In our experiments we show that IS indeed improves performance significantly.

4 CVaR Optimization

In this section, we consider the setting of Section 2.2, and aim to solve the CVaR optimization problem:

$$\max_{\theta \in \mathbb{R}^k} \Phi_\alpha(R;\theta). \tag{7}$$

For this goal we propose CVaRSGD: a stochastic gradient descent algorithm, based on the GCVaR gradient estimator. We now describe the CVaRSGD algorithm in detail, and show that it converges to a local optimum of (7).

In CVaRSGD, we start with an arbitrary initial parameter $\theta^0 \in \mathbb{R}^k$. The algorithm proceeds iteratively as follows. At each iteration i of the algorithm, we first sample n_i i.i.d. realizations $x_1, y_1, \ldots, x_{n_i}, y_{n_i}$ of the random variables $\mathbf{X}$ and Y, from the distribution $f_{\mathbf{X},Y}(\mathbf{x}, y; \theta^i)$. We then apply the GCVaR algorithm to obtain an estimate $\Delta_{j;n_i}$ of $\frac{\partial}{\partial \theta_j} \Phi_\alpha(R;\theta^i)$, using the samples $x_1, y_1, \ldots, x_{n_i}, y_{n_i}$. Finally, we update the parameter according to

$$\theta_j^{i+1} = \Gamma\left(\theta_j^i + \epsilon_i \Delta_{j;n_i}\right), \tag{8}$$

where ϵ_i is a positive step size, and $\Gamma : \mathbb{R}^k \to \mathbb{R}^k$ is a projection to some compact set Θ with a smooth boundary. The purpose of the projection is to facilitate convergence of the algorithm, by guaranteeing that the iterates remain bounded (this is a common stochastic approximation technique; Kushner and Yin, 2003). In practice, if Θ is chosen large enough so that it contains the local optima of $\Phi_\alpha(R;\theta)$, the projection would rarely occur, and would have a negligible effect on the algorithm. Let $\hat{\Gamma}_\theta(\nu) \doteq \lim_{\delta \to 0} \frac{\Gamma(\theta + \delta\nu) - \theta}{\delta}$ denote an operator that, given a direction of change ν to the parameter θ, returns a modified direction that keeps θ within Θ. Consider the following ordinary differential equation:

$$\dot{\theta} = \hat{\Gamma}_\theta\left(\nabla \Phi_\alpha(R;\theta)\right), \quad \theta(0) \in \Theta. \tag{9}$$

Let $\mathcal{K}$ denote the set of all asymptotically stable equilibria of (9). The next theorem shows that under suitable technical conditions, the CVaRSGD algorithm converges to $\mathcal{K}$ almost surely. The theorem is a direct application of Theorem 5.2.1 of Kushner and Yin (2003), and given here without proof.

Theorem 5. *Consider the CVaRSGD algorithm (8). Let Assumptions 4, 5, and 6 hold, and assume that $\Phi_\alpha(R;\theta)$ is continuously differentiable in θ. Also, assume that $\sum_{i=1}^{\infty} \epsilon_i = \infty$, $\sum_{i=1}^{\infty} \epsilon_i^2 < \infty$, and that $\sum_{i=1}^{\infty} \epsilon_i \left| \mathbb{E}[\Delta_{j;n_i}] - \frac{\partial}{\partial \theta_j} \Phi_\alpha(R;\theta^i) \right| < \infty$ w.p. 1 for all j. Then $\theta^i \to \mathcal{K}$ almost surely.*

Note that from the discussion in Section 3, the requirement $\sum_{i=1}^{\infty} \epsilon_i \left| \mathbb{E}\left[\Delta_{j;n_i}\right] - \frac{\partial}{\partial \theta_j} \Phi_\alpha(R;\theta^i) \right| < \infty$ implies that we must have $\lim_{i\to\infty} n_i = \infty$. However, the rate of n_i could be very slow, for example, using the bound of Theorem 4 the requirement may be satisfied by choosing $\epsilon_i = 1/i$ and $n_i = (\log i)^4$.

5 Application to Reinforcement Learning

In this section we show that the CVaRSGD algorithm may be used in an RL policy-gradient type scheme, for optimizing performance criteria that involve the CVaR of the total return. We first describe some preliminaries and our RL setting, and then describe our algorithm.

We consider an episodic[2] Markov Decision Problem (MDP) in discrete time with a finite state space $\mathcal{S}$ and a finite action space $\mathcal{A}$. At time $t \in \{0, 1, 2, \dots\}$ the state is s_t, and an action a_t is chosen according to a parameterized policy π_θ, which assigns a distribution over actions $f_{a|h}(a|h;\theta)$ according to the observed history of states $h_t = s_0, \dots, s_t$. Then, an immediate random reward $\rho_t \sim f_{\rho|s,a}(\rho|s,a)$ is received, and the state transitions to s_{t+1} according to the MDP transition probability $f_{s'|s,a}(s'|s,a)$. We denote by ζ_0 the initial state distribution and by s^* a terminal state, and we assume that for all θ, s^* is reached w.p. 1.

For some policy π_θ, let $s_0, a_0, \rho_0, s_1, a_1, \rho_1, \dots, s_\tau$ denote a state-action-reward trajectory from the MDP under that policy, that terminates at time τ, i.e., $s_\tau = s^*$. The trajectory is a random variable, and we decompose[3] it into a discrete part $Y \doteq s_0, a_0, s_1, a_1, \dots, s^*$ and a continuous part $X \doteq \rho_0, \rho_1, \dots, \rho_{\tau-1}$. Our quantity of interest is the total reward along the trajectory $R \doteq \sum_{t=0}^{\tau} \rho_t$. In standard RL, the objective is to find the parameter θ that maximizes the expected return $V(\theta) = \mathbb{E}^\theta[R]$. Policy gradient methods (Baxter and Bartlett 2001; Marbach and Tsitsiklis 1998; Peters and Schaal 2008) use simulation to estimate $\partial V(\theta)/\partial\theta_j$, and then perform stochastic gradient ascent on the parameters θ. In this work we are risk-sensitive, and our goal is to *maximize the CVaR of the total return* $J(\theta) \doteq \Phi_\alpha(R;\theta)$. In the spirit of policy gradient methods, we estimate $\partial J(\theta)/\partial\theta_j$ from simulation, using GCVaR, and optimize θ using CVaRSGD. We now detail our approach.

First, it is well known (Marbach and Tsitsiklis 1998) that by the Markov property of the state transitions:

$$\partial \log f_Y(Y;\theta)/\partial\theta = \sum_{t=0}^{\tau-1} \partial \log f_{a|h}(a_t|h_t;\theta)/\partial\theta. \quad (10)$$

Also, note that in our formulation we have

$$\partial \log f_{\mathbf{X}|Y}(x_i|y_i;\theta)/\partial\theta = 0, \quad (11)$$

since the reward does not depend on θ directly.

To apply CVaRSGD in the RL setting, at each iteration i of the algorithm we simulate n_i trajectories $x_1, y_1, \dots, x_{n_i}, y_{n_i}$ of the MDP using policy π_{θ^i} (each x_k and y_k here together correspond to a single trajectory, as realizations of the random variables X and Y defined above). We then apply the GCVaR algorithm to obtain an estimate $\Delta_{j;n_i}$ of $\partial J(\theta)/\partial\theta_j$, using the simulated trajectories $x_1, y_1, \dots, x_{n_i}, y_{n_i}$, Eq. (10), and Eq. (11). Finally, we update the policy parameter according to Eq. (8). Note that due to Eq. (10), the transition probabilities of the MDP, which are generally not known to the decision maker, are not required for estimating the gradient using GCVaR. Only policy-dependent terms are required.

We should remark that for the standard RL criterion $V(\theta)$, a Markov policy that depends only on the current state suffices to achieve optimality (Bertsekas 2012). For the CVaR criterion this is not necessarily the case. Bäuerle and Ott (2011) show that under certain conditions, an augmentation of the current state with a function of the accumulated reward suffices for optimality. In our simulations, we used a Markov policy, and still obtained useful and sensible results.

Assumptions 4, 5, and 6, that are required for convergence of the algorithm, are reasonable for the RL setting, and may be satisfied, for example, when $f_{\rho|s,a}(\rho|s,a)$ is smooth, and $\partial \log f_{a|h}(a|h;\theta)/\partial\theta_j$ is well defined and bounded. This last condition is standard in policy gradient literature, and a popular policy representation that satisfies it is softmax action selection (Sutton et al. 2000; Marbach and Tsitsiklis 1998), given by $f_{a|h}(a|h;\theta) = \frac{\exp(\phi(h,a)^\top \theta)}{\sum_{a'} \exp(\phi(h,a')^\top \theta)}$, where $\phi(h,a) \in \mathbb{R}^k$ are a set of k features that depend on the history and action.

In some RL domains, the reward takes only discrete values. While this case is not specifically covered by the theory in this paper, one may add an arbitrarily small smooth noise to the total reward for our results to hold. Since such a modification has negligible impact on performance, this issue is of little importance in practice. In our experiments the reward was discrete, and we did not observe any problem.

5.1 Experimental Results

We examine Tetris as a test case for our algorithms. Tetris is a popular RL benchmark that has been studied extensively. The main challenge in Tetris is its large state space, which necessitates some form of approximation in the solution technique. Many approaches to learning controllers for Tetris are described in the literature, among them are approximate value iteration (Tsitsiklis and Van Roy 1996), policy gradients (Kakade 2001; Furmston and Barber 2012), and modified policy iteration (Gabillon, Ghavamzadeh, and Scherrer 2013). The standard performance measure in Tetris is the expected number of cleared lines in the game. Here, we are interested in a risk-averse performance measure, captured by the CVaR of the total game score. Our goal in this section is to compare the performance of a policy optimized for the CVaR criterion versus a policy obtained using the standard policy gradient method. As we will show, optimizing the CVaR indeed produces a different policy, characterized by a risk-averse behavior. We note that at present, the best results in the literature (for the standard performance measure) were obtained using a modified policy iteration

[2] Also known as a stochastic shortest path (Bertsekas 2012).

[3] This decomposition is not restrictive, and used only to illustrate the definitions of Section 2. One may alternatively consider a continuous state space, or discrete rewards, so long as Assumptions 4, 5, and 6 hold.

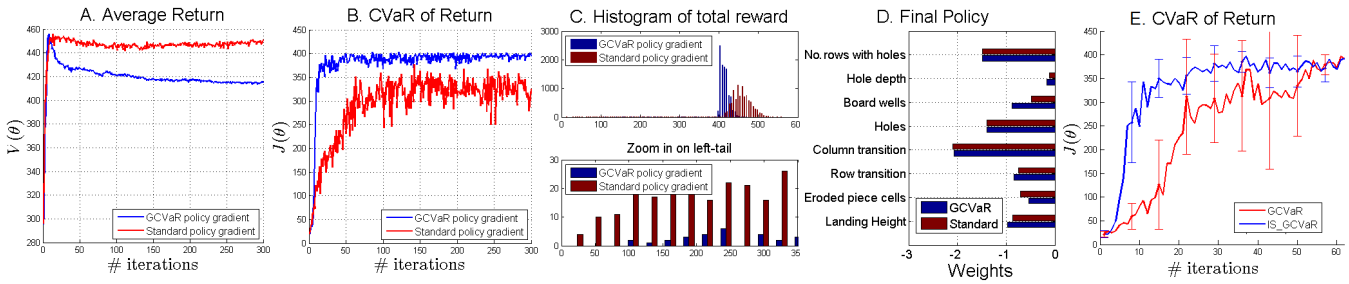

Figure 1: **GCVaR vs. policy gradient.** (A,B) Average return (A) and CVaR ($\alpha = 0.05$) of the return (B) for CVaRSGD and standard policy-gradient vs. iteration. (C) Histogram (counts from 10,000 independent runs) of the total return of the final policies. The lower plot is a zoom-in on the left-tail, and clearly shows the risk-averse behavior of the CVaRSGD policy. (D) Final policy parameters. Note the difference in the Board Well feature, which encourages risk taking. (E) CVaR ($\alpha = 0.01$) of the return for CVaRSGD vs. iteration, with and without importance sampling.

approach (Gabillon, Ghavamzadeh, and Scherrer 2013), and not using policy gradients. We emphasize that our goal here is not to compete with those results, but rather to illustrate the application of CVaRSGD. We do point out, however, that whether the approach of Gabillon, Ghavamzadeh, and Scherrer (2013) could be extended to handle a CVaR objective is currently not known.

We used the regular 10×20 Tetris board with the 7 standard shapes (a.k.a. *tetrominos*). In order to induce risk-sensitive behavior, we modified the reward function of the game as follows. The score for clearing 1,2,3 and 4 lines is 1,4,8 and 16 respectively. In addition, we limited the maximum number of steps in the game to 1000. These modifications strengthened the difference between the risk-sensitive and nominal policies, as they induce a tradeoff between clearing many 'single' lines with a low profit, or waiting for the more profitable, but less frequent, 'batches'.

We used the softmax policy, with the feature set of Thiery and Scherrer (2009). Starting from a fixed policy parameter θ_0, which was obtained by running several iterations of standard policy gradient (giving both methods a 'warm start'), we ran both CVaRSGD and standard policy gradient[4] for enough iterations such that both algorithms (approximately) converged. We set $\alpha = 0.05$ and $N = 1000$.

In Fig. 1A and Fig. 1B we present the average return $V(\theta)$ and CVaR of the return $J(\theta)$ for the policies of both algorithms at each iteration (evaluated by MC on independent trajectories). Observe that for CVaRSGD, the average return has been compromised for a higher CVaR value.

This compromise is further explained in Fig. 1C, where we display the reward distribution of the final policies. It may be observed that the left-tail distribution of the CVaR policy is significantly lower than the standard policy. For the risk-sensitive decision maker, such results are very important, especially if the left-tail contains catastrophic outcomes, as is common in many real-world domains, such as finance. To better understand the differences between the policies, we compare the final policy parameters θ in Fig. 1D. The most significant difference is in the parameter that corresponds to the Board Well feature. A *well* is a succession of unoccupied cells in a column, such that their left and right cells are both occupied. The controller trained by CVaRSGD has a smaller negative weight for this feature, compared to the standard controller, indicating that actions which create deep-wells are repressed. Such wells may lead to a high reward when they get filled, but are risky as they heighten the board.

To demonstrate the importance of IS in optimizing the CVaR when α is small, we chose $\alpha = 0.01$, and $N = 200$, and compared CVaRSGD against its IS version, IS_CVaRSGD, described in (Tamar, Glassner, and Mannor 2014). As Fig. 1E shows, IS_GCVaRSGD converged significantly faster, improving the convergence rate by more than a factor of 2. The full details are provided in (Tamar, Glassner, and Mannor 2014).

6 Conclusion and Future Work

We presented a novel LR-style formula for the gradient of the CVaR performance criterion. Based on this formula, we proposed a sampling-based gradient estimator, and a stochastic gradient descent procedure for CVaR optimization that is guaranteed to converge to a local optimum. To our knowledge, this is the first extension of the LR method to the CVaR performance criterion, and our results extend CVaR optimization to new domains.

We evaluated our approach empirically in an RL domain: learning a risk-sensitive policy for Tetris. To our knowledge, such a domain is beyond the reach of existing CVaR optimization approaches. Moreover, our empirical results show that optimizing the CVaR indeed results in useful risk-sensitive policies, and motivates the use of simulation-based optimization for risk-sensitive decision making.

Acknowledgments

The authors thank Odalric-Ambrym Maillard for many helpful discussions. The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Program (FP/2007-2013) / ERC Grant Agreement n. 306638.

[4] Standard policy gradient is similar to CVaRSGD when $\alpha = 1$. However, it is common to subtract a baseline from the reward in order to reduce the variance of the gradient estimate. In our experiments, we used the average return $< r >$ as a baseline, and our gradient estimate was $\frac{1}{N}\sum_{i=1}^{N} \frac{\partial \log f_Y(y_i;\theta)}{\partial \theta_j}(r(x_i,y_i) - < r >)$.

References

Agarwal, V., and Naik, N. Y. 2004. Risks and portfolio decisions involving hedge funds. *Review of Financial Studies* 17(1):63–98.

Bardou, O.; Frikha, N.; and Pagès, G. 2009. Computing VaR and CVaR using stochastic approximation and adaptive unconstrained importance sampling. *Monte Carlo Methods and Applications* 15(3):173–210.

Bäuerle, N., and Ott, J. 2011. Markov decision processes with average-value-at-risk criteria. *Mathematical Methods of Operations Research* 74(3):361–379.

Baxter, J., and Bartlett, P. L. 2001. Infinite-horizon policy-gradient estimation. *JAIR* 15:319–350.

Bertsekas, D. P. 2012. *Dynamic Programming and Optimal Control, Vol II*. Athena Scientific, 4th edition.

Borkar, V., and Jain, R. 2014. Risk-constrained Markov decision processes. *IEEE TAC* PP(99):1–1.

Borkar, V. S. 2001. A sensitivity formula for risk-sensitive cost and the actor–critic algorithm. *Systems & Control Letters* 44(5):339–346.

Fu, M. C. 2006. Gradient estimation. In Henderson, S. G., and Nelson, B. L., eds., *Simulation*, volume 13 of *Handbooks in Operations Research and Management Science*. Elsevier. 575 – 616.

Furmston, T., and Barber, D. 2012. A unifying perspective of parametric policy search methods for Markov decision processes. In *Advances in Neural Information Processing Systems 25*.

Gabillon, V.; Ghavamzadeh, M.; and Scherrer, B. 2013. Approximate dynamic programming finally performs well in the game of tetris. In *Advances in Neural Information Processing Systems 26*.

Glynn, P. W. 1990. Likelihood ratio gradient estimation for stochastic systems. *Communications of the ACM* 33(10):75–84.

Hong, L. J., and Liu, G. 2009. Simulating sensitivities of conditional value at risk. *Management Science*.

Iyengar, G., and Ma, A. 2013. Fast gradient descent method for mean-CVaR optimization. *Annals of Operations Research* 205(1):203–212.

Kakade, S. 2001. A natural policy gradient. In *Advances in Neural Information Processing Systems 14*.

Kushner, H., and Yin, G. 2003. *Stochastic approximation and recursive algorithms and applications*. Springer Verlag.

Marbach, P., and Tsitsiklis, J. N. 1998. Simulation-based optimization of Markov reward processes. *IEEE Transactions on Automatic Control* 46(2):191–209.

Morimura, T.; Sugiyama, M.; Kashima, H.; Hachiya, H.; and Tanaka, T. 2010. Nonparametric return distribution approximation for reinforcement learning. In *International Conference on Machine Learning*, 799–806.

Peters, J., and Schaal, S. 2008. Reinforcement learning of motor skills with policy gradients. *Neural Networks* 21(4):682–697.

Prashanth, L., and Ghavamzadeh, M. 2013. Actor-critic algorithms for risk-sensitive mdps. In *Advances in Neural Information Processing Systems 26*.

Prashanth, L. 2014. Policy gradients for CVaR-constrained MDPs. In *International Conference on Algorithmic Learning Theory*.

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

Rubinstein, R. Y., and Kroese, D. P. 2011. *Simulation and the Monte Carlo method*. John Wiley & Sons.

Scaillet, O. 2004. Nonparametric estimation and sensitivity analysis of expected shortfall. *Mathematical Finance*.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement learning: An introduction*. Cambridge Univ Press.

Sutton, R. S.; McAllester, D.; Singh, S.; and Mansour, Y. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 13*.

Tamar, A.; Di Castro, D.; and Mannor, S. 2012. Policy gradients with variance related risk criteria. In *International Conference on Machine Learning*.

Tamar, A.; Glassner, Y.; and Mannor, S. 2014. Optimizing the CVaR via sampling. *arXiv:1404.3862*.

Thiery, C., and Scherrer, B. 2009. Improvements on learning tetris with cross entropy. *International Computer Games Association Journal* 32.

Tsitsiklis, J. N., and Van Roy, B. 1996. Feature-based methods for large scale dynamic programming. *Machine Learning* 22(1-3):59–94.