Risk-averse Distributional Reinforcement Learning

A CVaR optimization approach

Silvestr Stanko[1]

[1]Department of Computer Science
Czech Technical University

February 7, 2018

Outline

1. Introduction
   - Motivation
   - Conditional Value-at-Risk
2. CVaR Value Iteration
   - Previous results
   - Linear-time improvement
3. Other Results

Motivation

Figure: Robotics

Figure: Finance

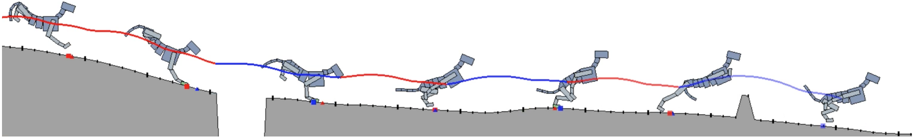


Figure: AI safety

Value-at-Risk

- Easy to understand
- Historically the most used risk-measure
- Undesirable computational properties
- Does not differentiate between large and catastrophic losses

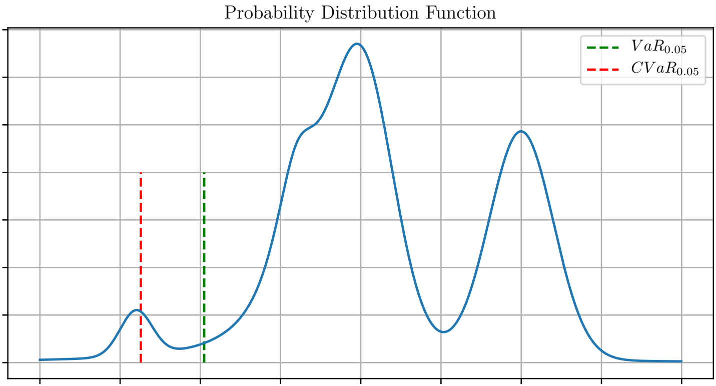
$$\mathrm{VaR}_\alpha(Z) = F^{-1}(\alpha) = \max\left\{z | F(z) \leq \alpha\right\}$$

Conditional Value-at-Risk

- Coherent risk measure
- Basel Committee on Banking Supervision $\rightarrow$ CVaR
- Equivalent to robustness

$$\mathrm{CVaR}_\alpha(Z) = \frac{1}{\alpha}\int_0^\alpha F_Z^{-1}(\beta)\mathrm{d}\beta = \frac{1}{\alpha}\int_0^\alpha \mathrm{VaR}_\beta(Z)\mathrm{d}\beta$$

VaR, CVaR



Reinforcement Learning

Definition

An MDP is a 5-tuple $\mathcal{M} = (\mathcal{X}, \mathcal{A}, R, P, \gamma)$, where $\mathcal{X}$ and $\mathcal{A}$ are the finite state and action spaces.
$R(x, a) \in [R_{\text{min}}, R_{\text{max}}]$ is a random variable representing the reward generated by being in state x and selecting action a; $P(\cdot|x, a)$ is the transition probability distribution; $\gamma \in [0, 1)$ is a discount factor.
We also assume we are given a starting state x_0.

Goal

Definition

$Z^\pi(x_t)$ Is a random variable representing the discounted reward along a trajectory generated by the MDP by following the policy π, starting at state x_t.

$$Z^\pi(x_t) = \sum_{t=0}^{\infty} \gamma^t R(x_t, \pi(x_t))$$

Reinforcement Learning with CVaR

For a given α, our goal is to find a globally optimal policy π^*

$$\pi^* = \arg\max_\pi CVaR_\alpha^\pi(Z^\pi(x_0))$$

Value Iteration

Theorem (CVaR decomposition)

For any $t \geq 0$, denote by $Z = (Z_{t+1}, Z_{t+2}, \dots)$ the reward sequence from time $t+1$ onwards. The conditional CVaR under policy π obeys the following decomposition:

$$CVaR_\alpha\left(Z^\pi(x,a)\right) = \min_{\xi \in \mathcal{U}_{CVaR}(\alpha, P(\cdot|x,a))} \sum_{x'} p(x'|x,a)\xi(x') CVaR_{\xi(x')\alpha}\left(Z^\pi(x')\right)$$

Theorem (CVaR Value Iteration)

The following Bellman operator is a contraction:

$$\mathbf{T}V(x,y) = \max_a \left[R(x,a) + \gamma \min_\xi \sum_{x'} p(x'|x,a)\xi(x') V\left(x', y\xi(x')\right) \right]$$

CVaR Value Iteration

Theorem (CVaR Value Iteration)

The following Bellman operator is a contraction:

$$\mathbf{T}V(x,y) = \max_a \left[R(x,a) + \gamma \min_\xi \sum_{x'} p(x'|x,a)\xi(x')V\left(x', y\xi(x')\right) \right]$$

The operator $\mathbf{T}$ describes the following relationship:

$$\mathbf{T}CVaR_y(Z(x)) = \max_a \left[R(x,a) + \gamma CVaR_y(Z(x,a)) \right]$$

Linear interpolation

Computing operator $\mathbf{T}$ s intractable, as the state-space is continuous. A solution would be to approximate the operator with linear interpolation.

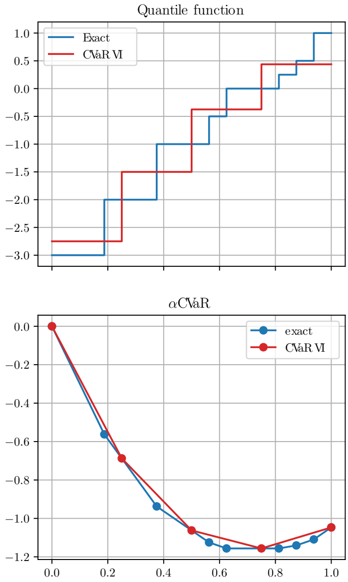
Theorem

The function $\alpha CVaR_\alpha$ is convex. The operator $\mathbf{T}_\mathcal{I}$ is a contraction.

$$\mathcal{I}_x[V](y) = y_i V(x, y_i) + \frac{y_{i+1} V(x, y_{i+1}) - y_i V(x, y_i)}{y_{i+1} - y_i}(y - y_i)$$

$$\mathbf{T}_\mathcal{I} V(x, y) = \max_a \left[R(x, a) + \gamma \min_\xi \sum_{x'} p(x'|x, a) \frac{\mathcal{I}_{x'}[V](y\xi(x'))}{y} \right]$$

This iteration can be formulated and solved as a linear program.


Quantile function
Exact
CVaR VI
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.5
−3.0
CVaR
exact
CVaR VI
−1.00
−1.25
−1.50
−1.75
−2.00
−2.25
−2.50
−2.75
−3.00
0.0
0.2
0.4
0.6
0.8
1.0
αCVaR
exact
CVaR VI
0.0
−0.2
−0.4
−0.6
−0.8
−1.0
−1.2
0.0
0.2
0.4
0.6
0.8
1.0

αCVaR$_\alpha$ duality

Lemma

Any discrete distribution has a piecewise linear $\alpha CVaR_\alpha$ function. Similarly, any a piecewise linear $\alpha CVaR_\alpha$ function can be seen as representing a certain discrete distribution.

αCVaR$_\alpha$ $\Leftarrow$ VaR

$$\frac{\partial}{\partial\alpha}\alpha\mathrm{CVaR}_\alpha(Z) = \frac{\partial}{\partial\alpha}\int_0^\alpha VaR_\beta(Z)d\beta = VaR_\alpha(Z)$$

αCVaR$_\alpha$ $\Rightarrow$ VaR

$$\alpha\mathrm{CVaR}_\alpha(Z) = \int_0^\alpha VaR_\beta(Z)\mathrm{d}\beta$$

Linear-time Computation

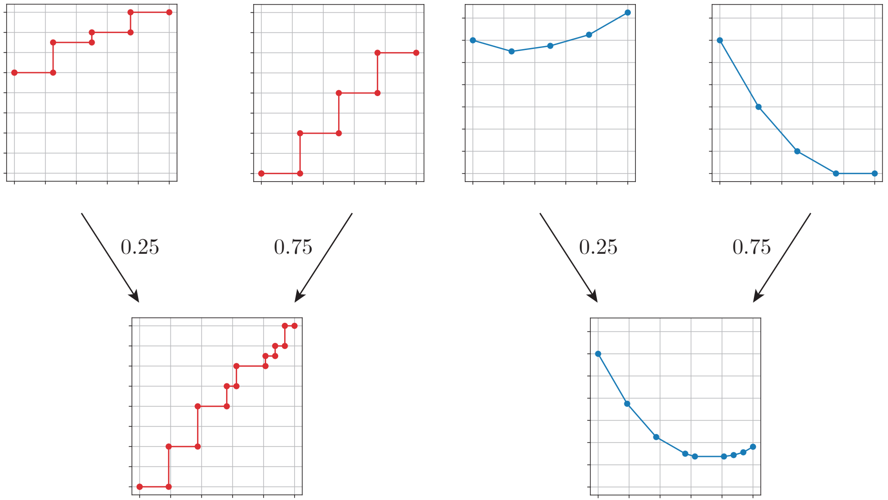
Theorem

Solution to minimization problem present in the CVaR Value Iteration can be computed by setting

$$\xi(x') = \frac{F_{x'}(F_x^{-1}(\alpha))}{\alpha}$$

The computational complexity is $O(n \cdot m)$ where n is the number of transition states and m is the number of atoms.

Next state CVaR computation



VaR-based Policy Improvement

Theorem

Let π be a fixed policy, $\alpha \in (0, 1]$. By following policy π' from the following algorithm, we will improve $CVaR_\alpha(Z)$ in expectation:

$$CVaR_\alpha(Z^\pi) \leq CVaR_\alpha(Z^{\pi'})$$

input α, x_0, γ
$a = \arg\max_a CVaR_\alpha(Z(x_0, a))$
$s = VaR_\alpha(Z(x_0, a))$
$x_t, r_t = \text{envTransition}(x_0, a)$
while x_t is not terminal **do**
  $s = \frac{s - r_t}{\gamma}$
  $a = \arg\max_a \mathbb{E}\left[(Z(x_t, a) - s)^-\right]$
  $x_t, r_t = \text{envTransition}(x_t, a)$
end while

Implementations

https://github.com/Silvicek/policy-improvement

https://github.com/Silvicek/distributional-dqn

TODO

- CVaR Q-learning
  - (?) Use Wasserstein distance with quantile improvement
  - (?) Extend the VaR-based algorithm
  - (?) Combine with quantile regression
- Experiments
  - Value Iteration + Q-learning
  - Deep Q-learning