Proceedings of Machine Learning Research vol 144:1–13, 2021

Adaptive Risk Sensitive Model Predictive Control with Stochastic Search

Ziyi Wang ZIYIWANG@GATECH.EDU
Oswin So OSWINSO@GATECH.EDU
Keuntaek Lee KEUNTAEK.LEE@GATECH.EDU
Evangelos A. Theodorou EVANGELOS.THEODOROU@GATECH.EDU
Georgia Institute of Technology

Abstract

We present a general framework for optimizing the Conditional Value-at-Risk for dynamical systems using stochastic search. The framework is capable of handling the uncertainty from the initial condition, stochastic dynamics, and uncertain parameters in the model. The algorithm is compared against a risk-sensitive distributional reinforcement learning framework and demonstrates improved performance on a simulated pendulum and cartpole with stochastic dynamics. We also showcase the applicability of the framework to robotics as an adaptive risk-sensitive controller by optimizing with respect to the fully nonlinear belief provided by a particle filter on a pendulum, cartpole, and quadcopter in simulation.

Keywords: Risk Sensitive Control, CVaR, Model Predictive Control, Stochastic Search

1. Introduction

The majority of Stochastic Optimal Control (SOC) and Reinforcement Learning (RL) literature handles uncertainty in dynamical optimization problems by simply optimizing the expected cost/reward. In many applications, however, it is desirable to consider the risk associated with the long tail events with high cost or low reward related to a policy instead of its performance on average. A simple but practical risk measure is the variance or mean-plus-variance (Gosavi, 2014; Di Castro et al., 2012). A major problem with variance is that it is a symmetric risk measure. The undesired high cost scenario is penalized the same way as the desired low cost outcome. Common asymmetric risk measures include exponential utility (Howard and Matheson, 1972; Fleming and McEneaney, 1995), which quantifies the exponential growth of risk as the cost increases, and Value-at-Risk (VaR) (Jorion, 2000), $\text{VaR}^\gamma(X) = \inf\{t : \mathbb{P}(X \leq t) \geq \gamma\}$, which quantifies statistically the γ-quantile of the uncertain cost distribution with $\gamma \in (0, 1)$ being the risk level. While exponential utility and VaR penalize one side of the cost distribution as desired, they are not *coherent* risk measures (see Wang et al. (2020) Appendix A for definition) (Artzner et al., 1999). Conditional Value-at-Risk (CVaR) (Rockafellar et al., 2000) is a natural extension to VaR defined as $\text{CVaR}^\gamma(X) = \frac{1}{1-\gamma}\int_\gamma^1 \text{VaR}^r(X)\mathrm{d}r$, which is equivalent to the conditional expectation beyond VaR, $\mathbb{E}[X|X \geq \text{VaR}^\gamma(X)]$, if X has continuous distribution. The main advantages of CVaR as a risk measure are that it is coherent, measures only the worst cases compared to exponential utility, but takes into account the entire tail instead of only the γ-quantile compared to VaR. Figure 1 illustrates the difference between risk measures on three distributions. Comparing the top and middle distributions, it is clear that symmetric risk measures

cannot capture the risk associated with a heavy tail. From the middle and bottom distributions, it can be observed that CVaR is more sensitive to the tail distribution than VaR.

CVaR has been used as a risk metric extensively in the field of finance (Agarwal and Naik, 2004; Krokhmal et al., 2002; Zhu and Fukushima, 2009), power utility (Conejo et al., 2010; Morales et al., 2010), supply chain management (Chen et al., 2007), etc. In recent years, it is also seeing a rise in popularity in robotics research. However, CVaR optimization for dynamical systems suffers from the problem of time inconsistency (Bjork and Murgoci, 2010), meaning that the optimal policy at a particular timestep might be suboptimal at a future time. We encourage the readers to refer to Shapiro et al. (2014) Section 6.8.5 for the mathematical definition of time consistency and Chow and Pavone (2015) Example 2 for an intuitive example on the time inconsistency of CVaR. The time inconsistency makes directly applying popular methods in SOC and RL that originated from dynamic programming to the CVaR optimization problem hard. Several methods have been proposed to overcome this problem by lifting the state space of the problem. In Pflug and Pichler (2016) and Chow et al. (2015), the CVaR decomposition theorem is introduced to obtain a dual representation of CVaR and optimizes the associated Bellman equation over a space of probability densities. Alternatively, the convex extremal formulation of CVaR (Rockafellar et al., 2000) can be used to alleviate the time-inconsistency problem (Bäuerle and Ott, 2011). Both approaches, however, require some form of state space augmentation and require solving an additional optimization problem. On the other hand, algorithms that directly optimize the policy (Tamar et al., 2014) are not affected by the time-inconsistency problem since they do not rely on the dynamic programming principle.

A promising sampling-based approach that directly optimizes the policy for solving general nonlinear optimization problems is stochastic search. Stochastic search is a general class of optimization methods that optimizes the objective function by randomly sampling and updating candidate solutions. Many well-known algorithms, such as the Cross Entropy Method (CEM) (Rubinstein and Kroese, 2013), the genetic algorithm (Zames et al., 1981), and simulated annealing (Kirkpatrick et al., 1983) fall into this category. Recently, a Gradient-based Adaptive Stochastic Search (GASS) algorithm was proposed by Zhou and Hu (2014). GASS updates the candidate solution by taking the gradient with respect to the sampling distribution parameters of solutions and approximating the gradient with Monte Carlo sampling. Boutselis et al. (2019) extended GASS to constrained dynamic optimization problems, and Wang et al. (2019) showed that the Information Theoretic Model Predictive Path Integral (MPPI) (Williams et al., 2017) emerges as a special case of dynamic GASS in the case of Gaussian and Poisson sampling distributions.

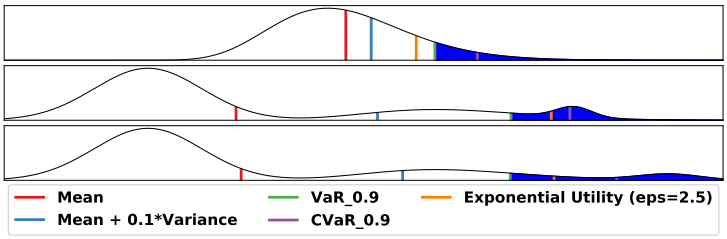

Figure 1: Comparison of different risk measures across three different distributions of costs.

In this paper, we extend the risk-sensitive formulation of GASS (Zhu et al., 2018) and present Risk-Sensitive Stochastic Search (RS3), a general framework for solving CVaR optimization for dynamical systems. The resulting algorithm bypasses the problem of time-inconsistency by directly performing stochastic gradient descent on the sampling distribution parameters. The framework is capable of handling uncertain initial states, parameters, and system stochasticity. We demonstrate the applicability of our framework to robotics by combining it with a particle filter to perform

risk-sensitive belief space control on a pendulum, cartpole and quadcopter in simulation. In addition, we show that our framework outperforms in terms of final average cost and under other risk measures compared to a risk-sensitive distributional RL algorithm, the Sample-based Distributional Policy Gradients (SDPG) (Singh et al., 2020b) on a pendulum and cartpole system in simulation.

The rest of this paper is organized as follows: in Section 2 we formulate the problem and derive the stochastic search framework. We present the algorithm and its application to belief space optimization in Section 3. The simulation results are included in Section 4. Finally, we conclude the paper in Section 5.

2. Stochastic Search

2.1. Notation, Mathematical Preliminaries, and Problem Formulation

Note that hereon we use $\mathbb{E}_{p(x)}[f(x)]$ to denote the integral $\int_{\Omega_x} f(x)p(x)\mathrm{d}x$ and $p(x)$ is dropped from the expectation for simplicity when it is clear which distribution the expectation is taken with respect to. We consider the problem of minimizing the CVaR of cost function $J : \mathbb{R}^{n_x \times (T+1)} \times \mathcal{U} \to \mathbb{R}^+$

$$U^* = \underset{U \in \mathcal{U}}{\operatorname{argmin}} \, \text{CVaR}^\gamma[J(X, U)], \tag{1}$$

subject to nonlinear stochastic dynamics

$$x_{t+1} \sim p(x_{t+1}|x_t, u_t; \phi). \tag{2}$$

Here we have $U = \{u_0, \cdots, u_{T-1}\} \in \mathcal{U}$ as the control path and $X = \{x_0, \cdots, x_T\} \in \mathbb{R}^{n_x \times (T+1)}$ as the state path where $T \in [0, \infty)$ is the optimization horizon. $\mathcal{U} \subset \mathbb{R}^{n_u \times T}$ is the set of admissible control sequences, and ϕ is the system parameters. This formulation is capable of handling uncertainties in state transition, $p(x_{t+1}|x_t, u_t)$, parameters, $p(\phi)$, and initial condition, $p(x_0)$. The CVaR is computed with respect to the uncertainty distributions. Assuming that $p(x_{t+1}|x_t, u_t; \phi)$, $p(x_0)$ and $p(\phi)$ are independent continuous density functions and J is continuous, the minimization problem (1) can be rewritten as

$$U^* = \underset{U \in \mathcal{U}}{\operatorname{argmin}} \, \mathbb{E}_{p(\chi)}[J(X, U)|J \geq \text{VaR}^\gamma(J)]. \tag{3}$$

where $p(\chi) = p(x_0)p(\phi)\prod_{t=0}^{T} p(x_{t+1}|x_t, u_t)$ is the joint pdf of all uncertainty distributions. We parameterize the control u with a policy π_η characterized by its parameters η. The policy can be of any functional form, i.e. open-loop ($u_t = v_t, \eta_t = v_t$), linear feedback ($u_t = k_t x_t + v_t, \eta_t = \{k_t, v_t\}$) or a deep neural network. We then define a sampling distribution for the policy parameters from the exponential family with a pdf of the form

$$p(\eta_t; \theta_t) = h(\eta_t) \exp\Big(\theta_t^{\mathrm{T}} T(\eta_t) - A(\theta_t)\Big), \tag{4}$$

where θ are the natural parameters of the distribution and $T(\eta)$ are the sufficient statistics of η. The minimization is now performed with respect to the natural parameters

$$\theta^* = \underset{\theta \in \Theta}{\operatorname{argmin}} \, \mathbb{E}_{p(\chi,\eta)}[J(X, U)|J \geq \text{VaR}^\gamma(J)]. \tag{5}$$

The expectation is now taken with respect to the joint distribution of uncertainty and sampling distribution. It is easy to show that the expected cost in (5) is an upper bound for optimal cost in (3).

2.2. Update Law Derivation

In this section we derive the gradient descent update rule for the optimization problem defined in (5). To be consistent with GASS, we turn the minimization problem into a maximization one by optimizing with respect to $-J$. We then introduce a shape function $S : \mathbb{R} \to \mathbb{R}^+$ which allows for different weighing schemes of the cost levels, leading to different optimization behaviors (Ollivier et al., 2017). The problem is then transformed into

$$\begin{aligned}\theta^* &= \underset{\theta\in\Theta}{\operatorname{argmax}}\, \mathbb{E}_{p(\eta)}[S(-\mathbb{E}_{p(\chi)}[J(X,U)|J \geq \text{VaR}^\gamma(J)])] \\ &= \underset{\theta\in\Theta}{\operatorname{argmax}}\, \mathbb{E}_{p(\eta)}[S(-\text{CVaR}^\gamma[J(X,U)])].\end{aligned} \tag{6}$$

The shape function needs to satisfy the following conditions:

i) $S(y)$ is nondecreasing in y and bounded from above and below for bounded y, with the lower bound being away from zero.

ii) The set of optimal solutions $\{\operatorname{argmax}_{y\in\mathcal{Y}} S(H(y))\}$ after the transform is a non-empty subset of the solutions $\{\operatorname{argmax}_{y\in\mathcal{Y}} H(y)\}$ of the original problem.

Common shape functions include: 1) the exponential function, $S(y;\kappa) = \exp(\kappa y)$, which leads to the Information Theoretic MPPI update law (Williams et al., 2017); 2) the sigmoid function, $S(y;\kappa,\varphi) = (y - y_{\text{lb}})\frac{1}{1+\exp(-\kappa(y-\varphi))}$, where y_{lb} is a lower bound for the cost and φ is the $(1-\rho)$-quantile, which results in an update law similar to CEM but with soft elite threshold ρ.

Finally, we apply another log transformation to obtain a gradient invariant to the scale of the objective function. The optimization problem becomes

$$\theta^* = \underset{\theta\in\Theta}{\operatorname{argmax}} \ln \mathbb{E}[S(-\text{CVaR}^\gamma[J(X,U)])] = \underset{\theta\in\Theta}{\operatorname{argmax}}\, l(\theta). \tag{7}$$

Since the natural logarithm function $\ln : \mathbb{R}^+ \to \mathbb{R}$ is a strictly increasing function, it does not change the maximization objective. We can now take its gradient with respect to the parameters. Writing the expectation as an integral with respect to the path probability $p(X,U;\theta)$ we get

$$\mathbb{E}\Big[S\Big(-\text{CVaR}^\gamma[J(X,U)]\Big)\Big] = \int_{\Omega_\eta} S\Big(-\int_{\Omega_\chi} J(X,U)p(X,U;\theta)\mathrm{d}\chi\Big)\mathrm{d}\eta, \tag{8}$$

where Ω_χ is defined such that $J \geq \text{VaR}^\gamma(J)$ if and only if $\chi \in \Omega_\chi$, and Ω_η is defined such that $\pi_{\eta_t}(x_t) \in \mathcal{U}_t, \forall t$. The path probability distribution can be decomposed as

$$p(X,U;\theta) = p(x_T|x_{T-1},\pi_\eta(x_{T-1});\phi)p(\eta_{T-1};\theta_{T-1})\cdots p(x_0)p(\phi) \tag{9}$$

$$= \underbrace{p(x_0)p(\phi)\prod_{t=0}^{T-1} p(x_{t+1}|x_t,\pi_\eta(x_t);\phi)}_{p(\chi)} \underbrace{\prod_{t=0}^{T-1} p(\eta_t;\theta_t)}_{p(\eta)}. \tag{10}$$

Note that since the uncertainty and sampling distribution are independent, their joint distribution can be broken into the product of the two. The gradient of the objective function (7) with respect to the parameters can be taken as

$$\nabla_\theta l(\theta) = \frac{\mathbb{E}[S(-\text{CVaR}^\gamma[J(X,U)])\nabla_\theta\big(\sum_{t=0}^{T-1}\ln p(\eta_t;\theta_t)\big)]}{\mathbb{E}[S(-\text{CVaR}^\gamma[J(X,U)])]}. \tag{11}$$

The detailed derivation can be found in Wang et al. (2020) Appendix B . The gradient of the log parameter distribution at each time step can be calculated as

$$\nabla_{\theta_t} \ln p(\eta_t; \theta_t) = \nabla_{\theta_t} \ln \Big(h(\eta_t) \exp \left(\theta_t^{\mathrm{T}} T(\eta_t) - A(\theta_t) \right) \Big) \tag{12}$$

$$= \nabla_{\theta_t} \left(\theta_t^{\mathrm{T}} T(\eta_t) - A(\theta_t) \right) \tag{13}$$

$$= T(\eta_t) - \nabla_{\theta_t} A(\theta_t). \tag{14}$$

Plugging it back into the gradient of the cost function, we get

$$\nabla_{\theta_t} l(\theta) = \frac{\mathbb{E}[S(-\text{CVaR}^{\gamma}[J(X,U)])(T(\eta_t) - \nabla_{\theta_t} A(\theta_t))]}{\mathbb{E}[S(-\text{CVaR}^{\gamma}[J(X,U)])]}. \tag{15}$$

With this, we have a gradient ascent update law for the parameters as

$$\theta_t^{k+1} = \Pi\{\theta_t^k + \alpha^k \nabla_{\theta_t} l(\theta^k)\}. \tag{16}$$

where Π is the projection operator ensuring the control constraints and the step size sequence α^k satisfies the typical assumptions in Stochastic Approximation (SA):

$$\alpha^k > 0 \quad \forall k, \quad \lim_{k \to \infty} \alpha^k = 0, \quad \sum_{k=0}^{\infty} \alpha_k = \infty \tag{17}$$

2.3. Practical Considerations

Numerical Approximation: The CVaR of the cost can be approximated Kolla et al. (2019) with

$$\hat{C}_n^{\gamma} = \hat{V}_n^{\gamma} + \frac{1}{M(1-\gamma)} \sum_{m=1}^{M} \left(J_{n,m} - \hat{V}_n^{\gamma} \right)^+ \tag{18}$$

$$\hat{V}_n^{\gamma} = \inf \left\{ x : \frac{1}{M} \sum_{m=1}^{M} \mathbb{1}_{\{J_{n,m} \leq x\}} \geq \gamma \right\}. \tag{19}$$

The outer expectation can be approximated as $\mathbb{E}[S(-\text{CVaR}^{\gamma}[J(X,U)])] = \frac{1}{N} \sum_{n=1}^{N} S(-\hat{C}_n^{\gamma})$. Note the expectation and CVaR in (15) are computed as averages over costs defined on entire trajectories.
Model Predictive Control Formulation: The parameter update in Equation (16) can be used for trajectory optimization as well as in a receding horizon or Model Predictive Control (MPC) fashion. MPC is a powerful algorithmic approach for nonlinear feedback control which is essential in tasks that involve risk measures or high order statistical characteristics of cost functions. In this paper we will leverage parallelization using GPUs to implement Equation (16) in MPC fashion.
Adaptive Stochastic Search: The MPC formulation allows online interaction with the stochastic system dynamics. Data from this online interaction can be used to feed adaptive or state estimation schemes that update the probability distribution $p(\chi)$ in an online fashion. In this work, we make use of a nonlinear state estimator, namely a particle filter, to propagate and update distribution $p(\chi)$ over time. The resulting control architecture is a sampling-based risk-sensitive adaptive MPC scheme that optimizes CVaR while adapting the probability distribution over parametric uncertainties. The details of this approach are further explained in the next section.

Algorithm 1 Stochastic Search for CVaR Optimization

Given:
F: Transition Model; K: Optimization Iterations; N: Number of Control Samples; M: Number of CVaR Optimization Samples; T: Number of timesteps; ϕ: System parameters vector; α_0: Initial step size; α_1: Step size decay rate; π_η: Policy; γ: Risk level;
Initialize:
θ^0: natural parameters for policy; S: Shape function;
while *task not completed* **do**
  $\{x_0^m\}_{m=1,\dots,M} \leftarrow$ *StateEstimator*()
  $\{\phi^m\}_{m=1,\dots,M} \leftarrow$ *SampleSystemParameters*()
  for $k \leftarrow 1$ to K **do**
    $\alpha^k \leftarrow \frac{\alpha_0}{k^{\alpha_1}}$
    $\{\eta^{n,k}\}_{n=1,\dots,N} \sim p(\eta;\theta^k)$
    for $n \leftarrow 1$ to N *in parallel* **do**
      for $m \leftarrow 1$ to M *in parallel* **do**
        for $t \leftarrow 0$ to $T-1$ **do**
          $u_{k,t}^{n,m} \sim \pi_{\eta^k}(x_t^{n,m})$
          $x_{t+1}^{n,m} \sim p(x_{t+1}^{n,m}|x_t^{n,m}, u_{k,t}^{n,m};\phi^m)$
        end for
        Compute cost: $J^{n,m}(X^{n,m}, U_k^n)$
      end for
      $\hat{C}^n \leftarrow$ *ComputeCVaR*$(S(-J^{n,m}), \gamma)$
    end for
    $\theta^{k+1} \leftarrow$ *UpdateParameters*$(\{\hat{C}^n\}, \alpha^k, \{\eta^{n,k}\})$
  end for
  Perform Polyak averaging: $\bar{\theta} \leftarrow \frac{1}{K}\sum_{i=1}^K \theta^i$
  Sample policy parameters $\bar{\eta} \sim p(\eta;\bar{\theta})$
  Apply policy: $\pi_{\bar{\eta}}$ for τ timesteps
  Recede horizon: $\theta^0 = \omega(\theta^K, \tau)$
end while

Algorithm 2 Parameter Update

Given:
N: Number of Control Samples; α^k: Stepsize; θ^k: Policy parameters; $\{\eta^k\}$: Policy samples; $\{\hat{C}\}$: CVaR values; Π: Projection operator for control constraints;
$\beta \leftarrow \min(\hat{C})$
$\eta \leftarrow \sum_{n=1}^N \exp\left(-(\hat{C}^n - \beta)\right)$
for $n \leftarrow 1$ to N *in parallel* **do**
  $\omega^n \leftarrow \frac{1}{\eta}\exp\left(-(\hat{C}^n - \beta)\right)$
  $\nabla_u^n = \eta^n - \frac{1}{N}\sum_{n=1}^N \eta^n$
end for
$\theta^{k+1} = \Pi\{\theta^k + \alpha^k \sum_{n=1}^N \frac{\omega^n \nabla_u^n}{\eta}\}$

3. Algorithm

In this section we present the RS3 algorithm implemented in MPC fashion, as shown in Algorithm 1. At initial time, the policy parameter distribution's natural parameters are initialized. Given an initial state and parameter distribution provided by an estimator, M i.i.d. samples are obtained. N policies are sampled from the policy parameter distribution and each copy of the policy is applied to all M samples of the initial states. In the case of stochastic dynamics, the states of each of the M samples are propagated with an independent realization of the stochastic dynamics. A cost is then calculated for each of the total $N \times M$ trajectories. For each policy sample, its associated CVaR cost is approximated with the M cost samples using (18) and (19). Using the CVaR values, the policy parameter distributions' natural parameters can be updated using (15) and (16). In our simulations, we use Gaussian distributions with fixed variance to sample policy parameters, for which the sufficient statistics are $T(\eta_t) = \eta_t$ and $\nabla_{\theta_t} A(\theta_t) = \mathbb{E}[T(\eta_t)]$. The parameter update step is detailed in algorithm 2. In this work, the projection step is done via clamping. As is common in SA algorithms, Polyak averaging is performed on the natural parameters to improve the convergence rate (Polyak and Juditsky, 1992). With the Polyak averaged natural parameters $\bar{\eta}$, an optimal policy can be sampled and applied to the system for τ timesteps. Finally, we apply a shift operator $\omega(\theta, \tau)$

that recedes the optimization horizon and outputs $\tilde{\theta}_t = \theta_{t+\tau}$. The last τ timesteps of the natural parameters are re-initialized.

Note that the RS3 algorithm can handle any or all uncertainties from initial state distribution, uncertain parameters and stochastic dynamics provided that i.i.d. samples can be generated from the uncertain distribution.

In our simulation examples of belief space control, we use a particle filter to provide the initial state distribution. To handle uncertain model parameters, we augment the states by the uncertain parameters and use a particle filter to learn its distribution (detailed in Wang et al. (2020) Appendix C) . In both cases, i.i.d. particles from non-Gaussian distributions can be directly used in the RS3 algorithm. However, we want to stress that any filter can be used together with the RS3 algorithm.

4. Simulation Results

In this section, we showcase the general applicability of RS3 in dealing various types of uncertainty. 1) *External noise:* Comparing its performance against the SDPG algorithm for CVaR optimization. 2) *Uncertain system parameters:* combining it with a particle filter to perform risk sensitive control in belief space. 3) *Uncertain initial condition*: This can be found in Wang et al. (2020) Appendix D.

All simulations were performed with a risk level of 0.9. The open loop policy $\pi_\eta(x) = \eta$ is used for all simulations, where η directly maps to the controls. The multivariate normal distribution is chosen as the sampling distribution, and we use the method proposed in Boutselis et al. (2019) to handle the box control constraints in the simulation tasks by sampling from a truncated multivariate normal distribution. The tuning parameters of all simulations are included in the Appendix of Wang et al. (2020).

4.1. Comparison Against Sample-based Distributional Policy Gradients

The Sample-based Distributional Policy Gradients algorithm Singh et al. (2020a,b) is one of the most recent work on optimizing CVaR for dynamical systems. SDPG Singh et al. (2020a) and the risk-sensitive version of SDPG Singh et al. (2020b) are actor-critic type policy gradient algorithms in the distributional RL Bellemare et al. (2017) setting.

The actor network parameterizes the policy and the critic network learns the return distribution by reparameterizing simple Gaussian noise samples. The risk-sensitive version of SDPG Singh et al. (2020b) is an extension of the naive SDPG Singh et al. (2020a) algorithm by using CVaR as a loss function to train the actor network to learn a risk-sensitive policy. In this paper, We compare RS3 and the risk-sensitive SDPG on two classic control systems in OpenAI Gym Brockman et al. (2016), a pendulum and a cartpole.

Typical RL algorithms always receive some state feedback either fully or partially from environments. Thus, to fairly compare against SDPG, we exploited an MPC scheme in RS3 to implicitly receive the state feedback and perform a receding-horizon optimization. In addition, as SDPG is unable to handle uncertainty in the initial states and controls, we consider deterministic initial states and system dynamics with additive noise h the control channels. To match the RS3 framework, the Gym environment's controls were modified to be continuous and use a quadratic cost function instead of the typical RL reward function -1, 0, or +1 implemented in Gym. The cost function used in the simulation can be found in Wang et al. (2020) Appendix D.1.1. All other training parameters for risk-sensitive SDPG were the same as the parameters used in the original work Singh et al. (2020b).

Table 1: Reward comparison results between RS3 (ours) vs. risk-sensitive SDPG Singh et al. (2020b)

System		Pendulum				Cartpole		
Noise Variance		0.3	1.0	2.0	3.0	0.3	1.0	2.0
RS3	Mean	169.2	172.0	177.0	183.1	291.0	299.3	311.9
	VaR	170.3	175.9	185.1	196.1	291.9	302.5	320.1
	CVaR	170.7	177.6	189.3	203.7	292.3	304.2	326.9
SDPG	Mean	171.1	173.4	178.4	185.2	302.1	430.4	591.3
	VaR	172.4	177.9	188.3	201.1	302.6	607.9	650.8
	CVaR	172.9	179.5	191.6	206.8	302.8	617.8	659.6

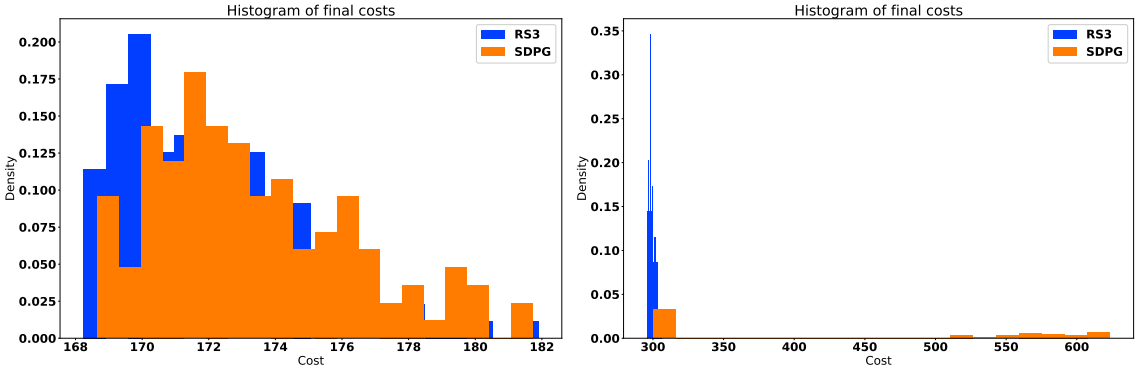

Figure 2: The histograms of the final cost in the case of injected control noise sampled from $\mathcal{N}(0, 1)$. *Left*: Pendulum, *Right*: Cartpole.

Under the aforementioned conditions, RS3 is shown to outperform SDPG overall by converging to a lower CVaR value, especially in the case of larger noise levels. The mean, VaR, and CVaR values of the final costs obtained from both algorithms for the pendulum and cartpole simulation are shown in Table 1 and a comparison of the histogram of the final costs are shown in Figure 2. The state, control, and cost histograms for all the simulation in Table 1 can be found in Wang et al. (2020) Appendix D.1.

It is clearly shown in Figure 2 that the distribution of the final cost has sharper tail on the high cost region in RS3's results compared to SDPG's. As a result, the mean, VaR, and CVaR of RS3's final costs are smaller than SDPG's.

The reason why our method outperforms the RL framework is that we perform online update of our policy whereas the RL policy is fixed after training. This disadvantage of RL algorithms comes from the nature of RL. Once a model is trained on a specific dataset or with a specific noise profile, the model fails to output correct predictions under a new environment or given unseen inputs or noise. Our online optimization scheme solves this issue and fits better in risk-sensitive control.

4.2. Belief Space Optimization

We next show results for the uncertain parameter case from the pendulum, cartpole and quadcopter systems. In each trajectory plot, the dotted lines represent estimates from the particle filter with the

error bars showing the $\pm 3\sigma$ uncertainties of the nonlinear belief. The solid line represents the ground truth states.

Pendulum:

We first apply RS3 to a pendulum for a swingup task with unknown pendulum mass. We assume deterministic initial condition and state transition model. The pendulum's true mass is set to 2 kg. The prior for pendulum mass is set to be $\mathcal{N}(5.0, 4.0)$. The initial states $\mathbf{x} = [\theta, \dot{\theta}]$ are drawn from a normal distribution with mean $[\pi, 0]$ and covariance matrix $\text{diag}([0.1, 0.1])$. We assume full-state observability with additive measurement noise $\xi \sim \mathcal{N}(0, 1)$. From Figure 3, we can observe that RS3 is able to correctly estimate the mass of the pendulum in the parameter estimation case. Without parameter estimation, RS3 overestimates the control effort required and overshoots the target angle.

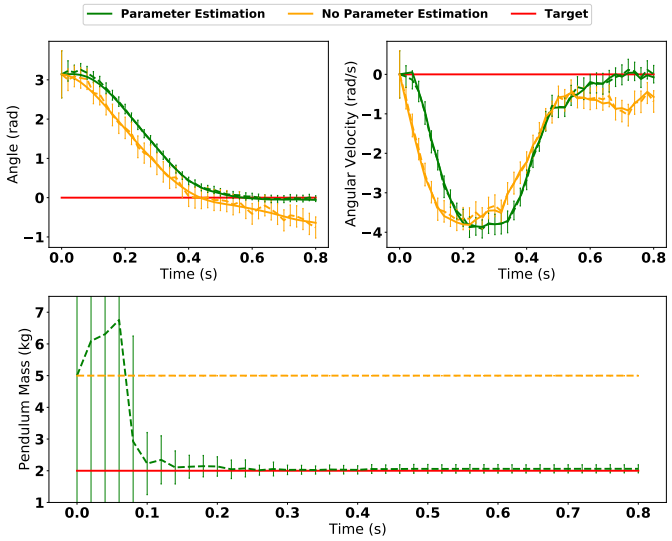

Figure 3: Nonlinear belief space optimization with uncertain pendulum mass in the Pendulum problem.

Cartpole:

We apply the proposed algorithm to the task of cartpole swingup with with unknown pole mass. The prior over the mass of the pole is a normal distribution $\mathcal{N}(5.0, 5.0)$ and the true value is 0.1 kg. Our algorithm is able to learn the true mass of the pole and successfully perform a swing up (Figure 4). We compare this with the case of not estimating the mass of the pole, where the algorithm does not sample from the correct dynamics and is unable to correctly optimize for a trajectory that successfully swings up.

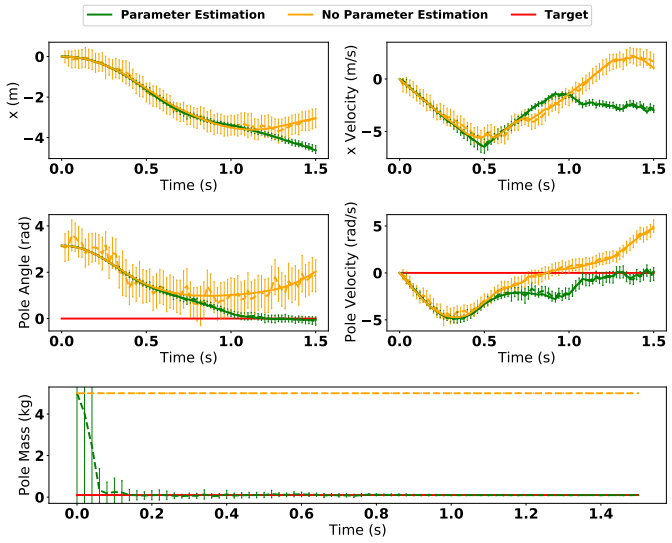

Figure 4: Nonlinear belief space optimization with uncertain pole mass in the Cartpole problem.

Quadcopter: Finally, we apply our algorithm to the quadcopter system (dynamics can be found in ElKholy (2014)), where the task is to fly a quadcopter with states $[x, y, z, \dot{x}, \dot{y}, \dot{z}, r, p, y, \dot{r}, \dot{p}, \dot{y}]$ from position $[0, 0, 0]$ to $[2, 2, 2]$. The drag coefficient of the system is unknown, the prior over the drag is a normal distribution $\mathcal{N}(0.5, 0.5)$ and the true value is 0.1. The algorithm is once again able to learn the correct drag coefficient and manages to pilot the quadcopter to the target position without significant overshoot despite the drag

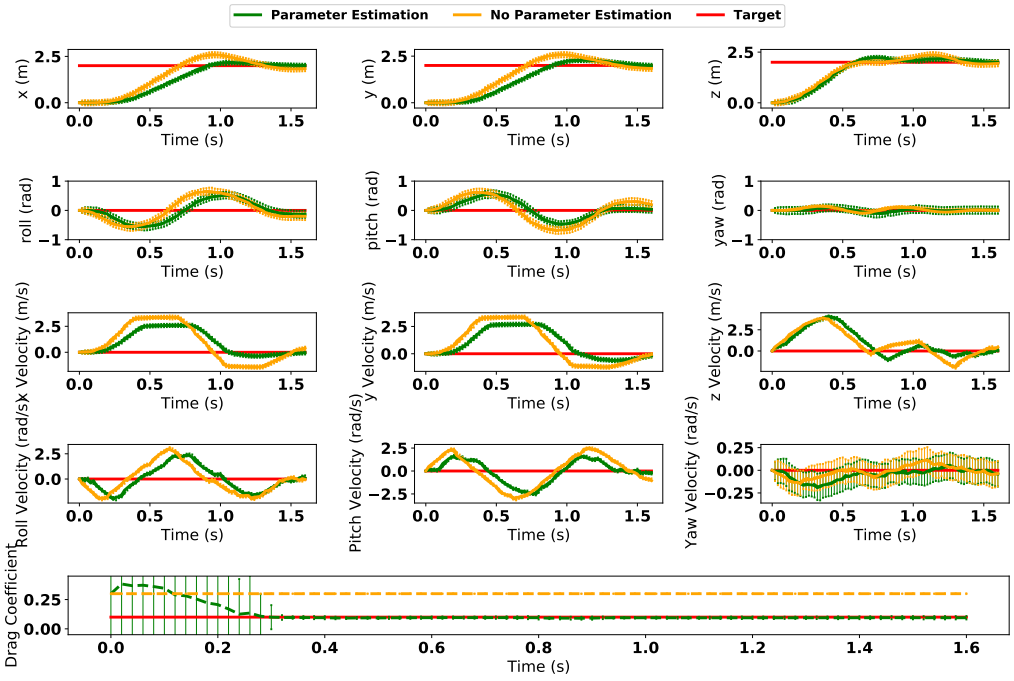

Figure 5: Nonlinear belief space optimization with uncertain drag coefficient in the Quadcopter problem. RS3 with parameter estimation is able to converge much closer in roll, pitch and pitch velocity compared to without parameter estimation.

coefficient being 500% larger than the mean of the prior (Figure 5). For the case where we do not perform parameter estimation, since the prior of the drag coefficient is greater than the actual value, the control policy found by the RS3 framework results in overshooting behavior before convergence, although it still manages to converge to the target state due to the robustness from optimizing for CVaR.

5. Conclusion

In this paper we introduced a general framework for CVaR optimization for dynamical systems. The resulting algorithm, RS3, is capable of handling uncertainties arising from uncertain initial conditions, model parameters and system stochasticity. The algorithm can be readily combined with a particle filter for belief space risk sensitive control. We compared RS3 against the distributional RL algorithm SDPG on the simulated systems of a pendulum and cartpole and demonstrated outperformance in terms of final CVaR cost. In addition, we combined RS3 with a particle filter for adaptive risk-sensitive control on non-Gaussian belief under different sources of uncertainty.

Acknowledgement

This work is supported by NASA Langley Research Center Grant #80NSSC19M0211, the NSF-CPS award #1932288, and the Amazon Web Services (AWS).

References

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

Philippe Artzner, Freddy Delbaen, Jean-Marc Eber, and David Heath. Coherent measures of risk. *Mathematical finance*, 9(3):203–228, 1999.

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

Marc G. Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 449–458, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

Tomas Bjork and Agatha Murgoci. A general theory of markovian time inconsistent stochastic control problems. *Available at SSRN 1694759*, 2010.

George I Boutselis, Ziyi Wang, and Evangelos A Theodorou. Constrained sampling-based trajectory optimization using stochastic approximation. *arXiv preprint arXiv:1911.04621*, 2019.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Xin Chen, Melvyn Sim, David Simchi-Levi, and Peng Sun. Risk aversion in inventory management. *Operations Research*, 55(5):828–842, 2007.

Yinlam Chow and Marco Pavone. A time consistent formulation of risk constrained stochastic optimal control. *arXiv preprint arXiv:1503.07461*, 2015.

Yinlam Chow, Aviv Tamar, Shie Mannor, and Marco Pavone. Risk-sensitive and robust decision-making: a cvar optimization approach. In *Advances in Neural Information Processing Systems*, pages 1522–1530, 2015.

Antonio J Conejo, Miguel Carrión, Juan M Morales, et al. *Decision making under uncertainty in electricity markets*, volume 1. Springer, 2010.

Dotan Di Castro, Aviv Tamar, and Shie Mannor. Policy gradients with variance related risk criteria. *arXiv preprint arXiv:1206.6404*, 2012.

Ht M ElKholy. Dynamic modeling and control of a quadrotor using linear and nonlinear approaches. *Master of Science in Robotics, The American University in Cairo*, 2014.

Wendell H Fleming and William M McEneaney. Risk-sensitive control on an infinite time horizon. *SIAM Journal on Control and Optimization*, 33(6):1881–1915, 1995.

Abhijit Gosavi. Variance-penalized markov decision processes: Dynamic programming and reinforcement learning techniques. *International Journal of General Systems*, 43(6):649–669, 2014.

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

Philippe Jorion. Value at risk. 2000.

Scott Kirkpatrick, C Daniel Gelatt, and Mario P Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

Ravi Kumar Kolla, LA Prashanth, Sanjay P Bhat, and Krishna Jagannathan. Concentration bounds for empirical conditional value-at-risk: The unbounded case. *Operations Research Letters*, 47(1): 16–20, 2019.

Pavlo Krokhmal, Jonas Palmquist, and Stanislav Uryasev. Portfolio optimization with conditional value-at-risk objective and constraints. *Journal of risk*, 4:43–68, 2002.

Juan M Morales, Antonio J Conejo, and Juan Pérez-Ruiz. Short-term trading for a wind power producer. *IEEE Transactions on Power Systems*, 25(1):554–564, 2010.

Yann Ollivier, Ludovic Arnold, Anne Auger, and Nikolaus Hansen. Information-geometric optimization algorithms: A unifying picture via invariance principles. *The Journal of Machine Learning Research*, 18(1):564–628, 2017.

Georg Ch Pflug and Alois Pichler. Time-inconsistent multistage stochastic programs: Martingale bounds. *European Journal of Operational Research*, 249(1):155–163, 2016.

Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992.

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

Reuven Y Rubinstein and Dirk P Kroese. *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation and machine learning*. Springer Science & Business Media, 2013.

Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczyński. *Lectures on stochastic programming: modeling and theory*. SIAM, 2014.

Rahul Singh, Keuntaek Lee, and Yongxin Chen. Sample-based distributional policy gradient. *arXiv preprint arXiv:2001.02652*, 2020a.

Rahul Singh, Qinsheng Zhang, and Yongxin Chen. Improving robustness via risk averse distributional reinforcement learning. *The 2nd Annual Conference on Learning for Dynamics and Control (L4DC)*, 2020b.

Aviv Tamar, Yonatan Glassner, and Shie Mannor. Policy gradients beyond expectations: Conditional value-at-risk. *arXiv preprint arXiv:1404.3862*, 2014.

Ziyi Wang, Grady Williams, and Evangelos A Theodorou. Information theoretic model predictive control on jump diffusion processes. In *2019 American Control Conference (ACC)*, pages 1663–1670. IEEE, 2019.

Ziyi Wang, Oswin So, Keuntaek Lee, Camilo A Duarte, and Evangelos A Theodorou. Adaptive cvar optimization for dynamical systems with path space stochastic search. *arXiv preprint arXiv:2009.01090*, 2020.

Grady Williams, Nolan Wagener, Brian Goldfain, Paul Drews, James M Rehg, Byron Boots, and Evangelos A Theodorou. Information theoretic mpc for model-based reinforcement learning. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1714–1721. IEEE, 2017.

G Zames, NM Ajlouni, NM Ajlouni, NM Ajlouni, JH Holland, WD Hills, and DE Goldberg. Genetic algorithms in search, optimization and machine learning. *Information Technology Journal*, 3(1): 301–302, 1981.

Enlu Zhou and Jiaqiao Hu. Gradient-based adaptive stochastic search for non-differentiable optimization. *IEEE Transactions on Automatic Control*, 59(7):1818–1832, 2014.

Helin Zhu, Joshua Hale, and Enlu Zhou. Simulation optimization of risk measures with adaptive risk levels. *Journal of Global Optimization*, 70(4):783–809, 2018.

Shushang Zhu and Masao Fukushima. Worst-case conditional value-at-risk with application to robust portfolio management. *Operations research*, 57(5):1155–1168, 2009.