Robotics: Science and Systems 2021
Held Virtually, July 12–16, 2021

Variational Inference MPC using Tsallis Divergence

Ziyi Wang*†, Oswin So*, Jason Gibson, Bogdan Vlahov, Manan S. Gandhi
Guan-Horng Liu and Evangelos A. Theodorou
Autonomous Control and Decision Systems Lab
Georgia Institute of Technology, Atlanta, Georgia
*These authors contributed equally
†Correspondence to: ZiyiWang@gatech.edu

***Abstract*—In this paper, we provide a generalized framework for Variational Inference-Stochastic Optimal Control by using the non-extensive Tsallis divergence. By incorporating the deformed exponential function into the optimality likelihood function, a novel Tsallis Variational Inference-Model Predictive Control algorithm is derived, which includes prior works such as Variational Inference-Model Predictive Control, Model Predictive Path Integral Control, Cross Entropy Method, and Stein Variational Inference Model Predictive Control as special cases. The proposed algorithm allows for effective control of the cost/reward transform and is characterized by superior performance in terms of mean and variance reduction of the associated cost. The aforementioned features are supported by a theoretical and numerical analysis on the level of risk sensitivity of the proposed algorithm as well as simulation experiments on 5 different robotic systems with 3 different policy parameterizations.**

I. Introduction

Variational Inference (VI) is a powerful tool for approximating the posterior distribution of the unobserved random variables [1]. VI recasts the approximation problem as an optimization problem. Instead of directly approximating the target distribution $p(z|x)$ of the latent variable z, VI minimizes the Kullback-Leibler (KL) divergence between a tractable variational distribution $q(z)$ and the target distribution. Due to its faster convergence and comparable performance to Markov Chain Monte Carlo sampling methods, VI has received increasing attention in machine learning and robotics [2, 3, 4].

VI has been applied to Stochastic Optimal Control (SOC) problems recently. In Okada and Taniguchi [5], the authors formulated the SOC problem as a VI problem by setting the desired policy distribution as the target distribution. The VI-SOC framework works directly in the space of policy distributions instead of specific policy parameterizations in most SOC and Reinforcement Learning (RL) frameworks. This gives rise to a unified derivation for a variety of parametric policy distributions, such as unimodal Gaussian and Gaussian mixture in Okada and Taniguchi [5]. Lambert et al. [6] derived the VI-SOC algorithm for non-parametric policy distribution using Stein variational gradient descent. Apart from the policy distribution, the VI-SOC framework is characterized by two components, the optimality likelihood function and distributional distance metric. The optimality likelihood function measures the likelihood of trajectory samples to be optimal and defines the cost/reward transform to allow for different algorithmic developments. The distributional distance metric is the measure of distance between the variational distribution and target distribution.

In existing VI-SOC works, the KL divergence is used as the distributional distance metric due to its simplicity. On the other hand, recent advances in VI research involve extending the framework to other statistical divergences, such as the α-divergence [7] and χ-divergence [8]. In Wang et al. [9], Regli and Silva [10], the authors proposed variants of the α-divergence to improve the performance and robustness of the inference algorithm. Wan et al. [11] further extended the VI framework to f-divergence, which is a broad statistical divergence family that recovers the KL, α and χ-divergence as special cases.

Tsallis divergence is another generalized divergence rooted in non-extensive statistical mechanics [12]. Tsallis divergence and Tsallis entropy are generalizations of the KL divergence and Shannon entropy respectively. The Tsallis entropy is non-additive (hence non-extensive) in the sense that for a system composed of (probabilistically) independent subsystems, the total entropy differs from the sum of the entropies of the subsystems [13]. Over the last two decades, an increasing number of complex natural, artificial and social systems have verified the predictions and consequences derived from the Tsallis entropy and divergence [14, 15, 16, 17]. Lee et al. [17] demonstrated that Tsallis entropy regularization leads to improved performance and faster convergence in RL applications.

In this paper, we provide a generalized formulation of the VI-SOC framework using the Tsallis divergence and introduce a novel Model Predictive Control (MPC) algorithm. The main contribution of our work is threefold:

- We propose the Tsallis VI-MPC algorithm, which allows for additional control of the shape of the cost transform compared to previous VI-MPC algorithms using KL divergence.
- We provide a holistic view of connections between Tsallis VI-SOC and the state-of-the-art Model Predictive Path Integral (MPPI) control, Cross Entropy Method (CEM) and Stochastic Search (SS) methods.
- We show, both analytically and numerically, that the proposed Tsallis VI-SOC framework achieves lower cost variance than MPPI and CEM. We further demonstrate the superior performance of Tsallis VI-SOC in mean cost and variance minimization on simulated systems from control theory and robotics for 3 different choices of

policy distributions.

The rest of this paper is organized as follows: in Section II, we review KL VI-SOC and derive the Tsallis VI-SOC framework. In Section III, we discuss the connections between Tsallis VI-SOC and related works. A reparameterization and analysis of the framework is included in Section IV, and we propose the novel Tsallis VI-MPC algorithm in Section V. We showcase the performance of the proposed algorithm against related works in Section VI and conclude the paper in Section VII.

II. Tsallis Variational Inference-Stochastic Optimal Control

In this section, we review the VI-SOC formulation using KL divergence [5] and derive the novel Tsallis divergence VI-SOC framework.

A. SOC Problem Formulation

In discrete time SOC, we work with trajectories $\tau \coloneqq \{X, U\}$ of state, $X \coloneqq \{x_0, x_1, \dots, x_T\} \in \mathbb{R}^{n_x \times T+1}$, and control, $U \coloneqq \{u_0, u_1, \dots, u_{T-1}\} \in \mathbb{R}^{n_u \times T}$ over a finite time horizon $T > 0$. The goal in SOC problems is to minimize the expected cost defined by an arbitrary cost function $J : \mathbb{R}^{n_x \times T} \times \mathbb{R}^{n_u \times T-1} \to \mathbb{R}^+$ with initial state distribution $p(x_0)$ and state transition probability $p(x_{t+1}|x_t, u_t)$ corresponding to stochastic dynamics $x_{t+1} = F(x_t, u_t, \epsilon_t)$ where $\epsilon_t \in \mathbb{R}^{n_\epsilon}$ is the system stochasticity.

B. KL VI-SOC

We can formulate the SOC problem as an inference problem and apply methods from VI. To apply VI to SOC, we introduce a dummy optimality variable $o \in \{0, 1\}$ with $o = 1$ indicating that the trajectory $\tau = \{X, U\}$ is optimal. Table I compares the differences in notation between conventional VI methods and SOC formulated as a VI problem.

The objective of VI-SOC is to sample from the posterior distribution

$$\begin{aligned} p(\tau|o=1) &= \frac{p(o=1|\tau)p(\tau)}{p(o=1)} \\ &= \frac{p(o=1|\tau)}{p(o=1)} p(x_0) \prod_{t=0}^{T-1} p(x_{t+1}|x_t, u_t) p(u_t). \end{aligned} \tag{1}$$

Here $p(x_{t+1}|x_t, u_t)$ is the state transition probability, $p(x_0)$ is the initial state distribution and $p(u_t)$ is some prior control distribution (e.g. zero mean Gaussian or uniform distribution). For simplicity, we use o to indicate $o = 1$ and o' for $o = 0$ from here on. We can now formulate the VI-SOC objective as minimizing the distance between a controlled distribution $q(\tau)$ and the target distribution $p(\tau|o)$

$$\begin{aligned} q^*(\tau) &= \arg\min_{q(\tau)} \mathcal{D}_{KL}\left(q(\tau) \parallel p(\tau|o)\right) \\ &= \arg\min_{q(\tau)} \mathbb{E}_{q(\tau)}\left[\log \frac{p(X|U)q(U)}{\frac{p(o|\tau)}{p(o)} p(X|U)p(U)}\right] \\ &= \arg\min_{q(\tau)} \mathbb{E}_{q(\tau)}\left[-\log p(o|\tau) + \sum_{t=0}^{T-1} \log \frac{q(u_t)}{p(u_t)}\right], \end{aligned} \tag{2}$$

TABLE I: Notation comparison between Variational Inference and Variational Inference-Stochastic Optimal Control.

VI		VI-SOC	
Notation	Meaning	Notation	Meaning
x	data	o	optimality
z	latent variable	τ	trajectory
$p(z)$	prior distribution	$p(\tau)$	prior distribution
$q(z)$	variational distribution	$q(\tau)$	controlled distribution
$p(x\|z)$	generative model	$p(o\|\tau)$	optimality likelihood
$p(z\|x)$	posterior distribution	$p(\tau\|o)$	optimal distribution

where the constant $p(o)$ is dropped. The first term in the objective measures the likelihood of a trajectory being optimal while the second term serves as regularization. Splitting the expectation in the first term, we get

$$\begin{aligned} q^*(U) = \arg\min_{q(U)} \Big\{ &- \mathbb{E}_{q(U)}[\mathbb{E}_{p(X|U)}[\log p(o|\tau)]] \\ &+ \mathcal{D}_{KL}\left(q(U) \parallel p(U)\right) \Big\}, \end{aligned} \tag{3}$$

where $p(U)$ and $q(U)$ represent $\prod_{t=0}^{T-1} p(u_t)$ and $\prod_{t=0}^{T-1} q(u_t)$ respectively due to independence and $p(X|U) = p(x_0) \prod_{t=0}^{T-1} p(x_{t+1}|x_t, u_t)$.

Optimality Likelihood: The optimality likelihood in Eq. (3) can be parameterized by a non-increasing function of the trajectory cost $J(X, U)$ as $p(o|\tau) \coloneqq f(J(X, U))$. The monotonicity requirement ensures that trajectories incurring higher costs are always less likely to be optimal. Common choices of f include $f(x) = \exp(-x)$ and $f(x) = 1_{\{x \leq \gamma\}}$. In this paper, we choose $f(x) = \exp(-x)$ such that $\log p(o|\tau) = -J(X, U)$. To avoid excessive notation, we abuse the notation to define $J \coloneqq \mathbb{E}_{p(X|U)}[J(X, U)]$ and the N-sample empirical approximation of the expectation $J \coloneqq \sum_{n=1}^{N}[J^{(n)}(X, U)]$. Hence, Eq. (3) takes the form

$$q^*(U) = \arg\min_{q(U)} \mathbb{E}_{q(U)}[J] + \mathcal{D}_{KL}\left(q(U) \parallel p(U)\right), \tag{4}$$

which can be interpreted as an application of the famous maximum entropy principle.

C. Tsallis VI-SOC

In this subsection, we use the Tsallis divergence as the regularization function and derive the Tsallis-VI-SOC algorithm. First, we define the deformed logarithm and exponential as

$$\log_r(x) = \frac{x^{r-1} - 1}{r - 1}, \tag{5}$$

$$\exp_r(x) = (1 + (r-1)x)_+^{\frac{1}{r-1}}, \tag{6}$$

where $(\cdot)_+ \coloneqq \max(0, \cdot)$ and $r > 0$. Using $\log_r$ and $\exp_r$ we can define the Tsallis entropy and the corresponding Tsallis divergence [12] as

$$\begin{aligned} \mathcal{S}_r(q(z)) &\coloneqq -\mathbb{E}_q[\log_r q(z)] \\ &= -\frac{1}{r-1}\left(\int q(z)^r \mathrm{d}z - 1\right), \end{aligned} \tag{7}$$

and

$$\mathcal{D}_r\left(q(z) \parallel p(z)\right) := \mathbb{E}_q\left[\log_r \frac{q(z)}{p(z)}\right] = \frac{1}{r-1}\left(\int q(z)\left(\frac{q(z)}{p(z)}\right)^{r-1} \mathrm{d}z - 1\right). \tag{8}$$

Note that as $r \to 1$, $\log_r \to \log$, $\exp_r \to \exp$, $\mathcal{D}_r \to \mathcal{D}_{KL}$ and $\mathcal{S}_r \to \mathcal{S}$, where $\mathcal{S}$ is the Shannon entropy, recovering the KL VI-SOC framework.

Versions of Tsallis Statistics: In our definition of the deformed logarithm and exponential in Eqs. (5) and (6), we use the variable r instead of the q used in most literature to avoid assigning multiple meanings to q. Also, our definition of $\log_r$ and $\exp_r$ differ from its original definitions [18], but the original can be recovered with $r' = 2 - r$, where r' corresponding to the value used in [18]. Additionally, there are multiple formulations of the Tsallis entropy differing in their definitions of the internal energy and how expectations are taken [12, 19, 20]. We have chosen to use the formulation from [12] due to its simplicity in computing the expectation. However, as shown in [21], they are equivalent and can be recovered from each other via a change of variables.

We can now define a new objective by replacing the KL divergence regularizer in Eq. (4) with the Tsallis divergence and introducing a parameter λ that multiplies the optimality likelihood to make the regularization strength tunable:

$$q^*(U) = \arg\min_{q(U)} \lambda^{-1}\mathbb{E}_{q(U)}[J] + \mathcal{D}_r\left(q(U) \parallel p(U)\right). \tag{9}$$

The controlled distribution has to satisfy the additional constraint of $\int q(U)\,\mathrm{d}U = 1$. The optimal policy distribution q^* can be explicitly solved for with the form

$$q^*(U) = \frac{\exp_r\left(-\tilde{\lambda}^{-1}J\right)p(U)}{\int \exp_r\left(-\tilde{\lambda}^{-1}J\right)p(U)\,\mathrm{d}U}, \tag{10}$$

where $\tilde{\lambda} = \alpha(r-1)\lambda$ with α being the Lagrange multiplier for the constraint that q^* integrates to 1. A detailed derivation can be found in Section SM-1 of the supplementary materials. We can now use Eq. (10) to obtain the optimal control distribution by transforming the prior distribution.

D. Update Laws

In general, it is computationally inefficient to sample from $q^*(U)$ via Eq. (10) directly. Instead, we can approximate $q^*(U)$ by some policy $\pi(U)$ lying in a class Π of tractable distributions and solve for an iterative update law by minimizing the KL divergence between $\pi(U)$ and $q^*(U)$:

$$\pi^{k+1}(U) = \arg\min_{\pi(U)\in\Pi} \mathcal{D}_{KL}\left(q^*(U) \parallel \pi(U)\right). \tag{11}$$

However, because one can only evaluate $q^*(U)$ at a finite number of points $\{U^{(n)}\}_{n=1}^N$, we instead approximate $q^*(U)$ by the empirical distribution $\tilde{q}^*(U)$ with weights $w^{(n)}$:

$$\tilde{q}^*(U) = \sum_{n=1}^N w^{(n)} \mathbf{1}_{\{U=U^{(n)}\}}, \tag{12}$$

$$w^{(n)} = \frac{q^*(U^{(n)})}{\sum_{n'=1}^N q^*(U^{(n')})}. \tag{13}$$

We now solve Eq. (11) for 3 different policy classes: unimodal Gaussian, Gaussian mixture and a nonparametric policy corresponding to Stein Variational Gradient Descent (SVGD) [6, 22]. The full derivations for each can be found in the supplementary material in Section SM-2.

Unimodal Gaussian: For a unimodal Gaussian policy distribution with parameters $\Theta := \{(\mu_t, \Sigma_t)\}_{t=0}^{T-1}$, the update laws for the $k+1$th iteration take the form of

$$\mu_t^{k+1} = \sum_{n=1}^N w^{(n)} u_t^{(n)}, \tag{14}$$

$$\Sigma_t^{k+1} = \sum_{n=1}^N w^{(n)}(u_t^{(n)} - \mu_t^{k+1})(u_t^{(n)} - \mu_t^{k+1})^{\mathrm{T}}. \tag{15}$$

Gaussian Mixture: Alternatively, the policy distribution can be an L-mode mixture of Gaussian distribution with parameters $\Theta := \{\theta_l\}_{l=1}^L$ for $\theta_l := (\phi_l, \{\mu_{l,t}, \Sigma_{l,t}\}_{t=0}^{T-1})$, where ϕ_l is the mixture weight for the lth component. Although it is not possible to directly solve Eq. (11) in this case, we draw from Expectation Maximization (EM) to derive an iterative update scheme for the $k+1$th iteration with the form

$$\phi_l^{k+1} = \frac{N_l}{\sum_{l'=1}^L N_{l'}}, \tag{16}$$

$$\mu_{l,t}^{k+1} = \frac{1}{N_l}\sum_{n=1}^N \eta_l(u_t^n) w^{(n)} u_t^n, \tag{17}$$

$$\Sigma_{l,t}^{k+1} = \frac{1}{N_l}\sum_{n=1}^N \eta_l(u_t^n) w^{(n)}(u_t^n - \mu_{l,t}^{k+1})(u_t^n - \mu_{l,t}^{k+1})^{\mathrm{T}}, \tag{18}$$

where

$$\eta_l(u_t^{(n)}) = \frac{\phi_l^k \mathcal{N}(u_t^{(n)} | \mu_{l,t}^k, \Sigma_{l,t}^k)}{\sum_{l'=1}^L \phi_{l'}^k \mathcal{N}(u_t^{(n)} | \mu_{l',t}^k, \Sigma_{l',t}^k)}, \tag{19}$$

$$N_l = \sum_{n=1}^N \sum_{t=0}^{T-1} \eta_l(u_t^{(n)}) w^{(n)}. \tag{20}$$

Stein Variational Policy: The policy can also be a non-parametric distribution approximated by a set of particles $\Theta := \{\theta_l\}_{l=1}^L$ for some parametrized policy $\hat{\pi}(U;\theta)$. In [6], $\hat{\pi}$ is taken to be a unimodal Gaussian with fixed variance, where $\theta \in \mathbb{R}^{n_x \times (T-1)}$ corresponds to the mean. The update law of

each Stein particle for the $k+1$th iteration has the form

$$\theta_l^{k+1} = \theta_l^k + \epsilon\hat{\phi}^*(\theta_l^k), \tag{21}$$

$$\hat{\phi}^*(\theta) = \sum_{l=1}^{L} \hat{k}(\theta_l, \theta)G(\theta_m) + \nabla_{\theta_l}\hat{k}(\theta_l, \theta), \tag{22}$$

$$G(\theta_l) = \frac{\sum_{s=1}^{S} w^{(l,s)} \nabla_\theta \log \hat{\pi}(U^{(n)}, \theta_l)}{\sum_{s=1}^{S} w^{(l,s)}}, \tag{23}$$

where the N is chosen such that $N = LS$, $\hat{k}$ is a kernel function, and $w^{(l,s)} = w^{(m+L(l-1))}$, where S denotes the number of rollouts for each of the L particle. As noted in [23], SVGD becomes less effective as the dimensionality of the particles increases due to the inverse relationship between the repulsion force in the update law and the dimensionality. Hence, we follow [6] in choosing a sum of local kernel functions as our choice of $\hat{k}$.

III. Connections to Related Works

In this section, we compare VI-SOC with three sampling-based methods from optimization, thermodynamics, and information theory that have been applied to stochastic control problems. A comparison of problem formulations and update laws for different approaches with a unimodal Gaussian policy is in Table II.

A. Stochastic Search

SS [24] is a stochastic optimization scheme and has also been applied to the SOC setting [25, 26]. SS-SOC parameterizes the control policy with a distribution from the exponential family during problem formulation and optimizes with respect to policy parameters whereas VI-SOC performs optimization at the distribution level. The SS-SOC framework formulates the problem as

$$\theta^* = \arg\max_\theta \mathbb{E}_{\pi(U;\theta)} \left[S(\mathbb{E}_{p(X|U)}[J(X,U)]) \right], \tag{24}$$

where $S(\cdot)$ is a monotonically decreasing shape function. Note that in Eq. (24), the expectation taken with respect to $p(X|U)$ is inside the shape function S as opposed to outside as in Eq. (3). For convex shape/optimality likelihood functions, the objective in VI-SOC corresponds to an upper bound of that in SS-SOC, i.e.,

$$S(\mathbb{E}_{p(X|U)}[J(X,U)]) \leq \mathbb{E}_{p(X|U)}[S(J(X,U))]. \tag{25}$$

A detailed comparison of the two formulations can be found in [5].

The parameter update of SS-SOC has the form

$$\theta_t^{k+1} = \theta_t^k + \beta \sum_{n=1}^{N} \frac{S(J^{(n)})(T(u_t^{(n)}) - \frac{1}{N}\sum_{\tilde{n}=1}^{N} T(u_t^{(\tilde{n})}))}{\sum_{n'=1}^{N} S(J^{(n')})}, \tag{26}$$

where $T(\cdot)$ denotes the sufficient statistic for the corresponding parameter and β is the step size. In the case of a unimodal Gaussian policy, the update in Eq. (26) is equivalent to Eqs. (14) and (15) with $S(J) = \exp_r(-\tilde{\lambda}^{-1}J)$.

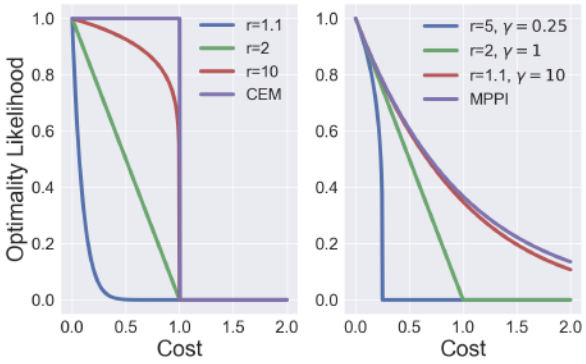

Fig. 1: Reparameterized $\exp_r$ (*Eq.* (28)). *Left*: r is varied with fixed $\gamma = 1$. It is clear that $\exp_r \to$ CEM as $r \to \infty$. *Right*: γ is adjusted accordingly such that $\exp_r \to$ MPPI as $r \to 1$.

B. Cross Entropy Method

CEM [27] is a widely used algorithm in reinforcement learning and optimal control problems [28, 29]. The objective function of CEM minimizes the expected cost $\mathbb{E}[J]$. The policy update law for CEM corresponds to that of SS-SOC in Eq. (26) with shape function $S(J) = \mathbf{1}_{\{J \leq \gamma\}}$ where γ is the elite threshold. As will be shown in Section IV-A, CEM is also a special case of the reparameterized Tsallis VI-SOC with $r \to \infty$.

C. Model Predictive Path Integral Control

MPPI is another approach closely related to VI-SOC and SS-SOC [30, 31]. The framework solves for the controls by minimizing the KL divergence between a controlled distribution and the optimal control distribution

$$q^*(U) = \frac{\exp\left(-\lambda^{-1}J\right)p(U)}{\int \exp\left(-\lambda^{-1}J\right)p(U)\,\mathrm{d}U}. \tag{27}$$

The optimal distribution achieves the free energy lower bound, $\mathcal{F} = -\lambda \log \mathbb{E}_{p(U)}[\exp(-\frac{1}{\lambda}J)]$, which has been shown to be the solution of the Hamilton-Jacobi-Bellman equation [32]. The optimal distribution here is equivalent to Eq. (10) when $r \to 1$. The corresponding update law can also be obtained from the SS-SOC framework with $S(J) = \exp(-\lambda^{-1}J)$.

IV. Analysis

A. Effect of $\exp_r$

To facilitate the analysis, we focus on the $\exp_r$ term in Eq. (10). Reparametrizing $\tilde{\lambda} = (r-1)\gamma$, we get

$$\begin{aligned}\exp_r(-\tilde{\lambda}^{-1}J) &= \left(1 - \frac{J}{\gamma}\right)_+^{\frac{1}{r-1}} \\ &= \begin{cases} \exp\left(\frac{1}{r-1}\log\left(1-\frac{J}{\gamma}\right)\right), & J < \gamma \\ 0, & J \geq \gamma \end{cases},\end{aligned} \tag{28}$$

where γ is now the threshold beyond which the optimality weight is set to 0. The reparameterization adjusts the original

TABLE II: Comparison of the objective and the update law for a unimodal Gaussian policy with fixed variance between different SOC approaches.

Formulation	Objective (Minimize)	Update Law for Unimodal Gaussian
Tsallis VI-SOC	$-\lambda\mathbb{E}[\log p(o\|\tau)] + \mathcal{D}_r\left(q \parallel p\right)$	$\theta_t^{k+1} = \sum_{n=1}^{N} \frac{\exp_r\left(-\tilde{\lambda}^{-1} J^n\right) s(u_t^n) u_t^n}{\sum_{n'=1}^{N} \exp_r\left(-\tilde{\lambda}^{-1} J^{n'}\right) s(u_t^{n'})}$
KL VI-SOC	$-\lambda\mathbb{E}[\log p(o\|\tau)] + \mathcal{D}_{KL}\left(q \parallel p\right)$	$\theta_t^{k+1} = \sum_{n=1}^{N} \frac{\exp\left(-\lambda^{-1} J^n\right) s(u_t^n) u_t^n}{\sum_{n'=1}^{N} \exp\left(-\lambda^{-1} J^{n'}\right) s(u_t^{n'})}$
SS-SOC	$\mathbb{E}[S(J)]$	$\theta_t^{k+1} = \theta_t^k + \beta \sum_{n=1}^{N} \frac{S(J^n)(u_t^n - \frac{1}{N}\sum_{\bar{n}=1}^{N} u_t^{\bar{n}})}{\sum_{n'=1}^{N} S(J^{n'}) u_t^{n'}}$
MPPI	$\mathbb{E}[J] + \mathcal{D}_{KL}\left(q \parallel p\right)$	$\theta_t^{k+1} = \theta_t^k + \sum_{n=1}^{N} \frac{\exp(-\lambda^{-1} J^n) s(u_t^n) u_t^n}{\sum_{n'=1}^{N} \exp(-\lambda^{-1} J^{n'}) s(u_t^{n'})}$
CEM	$\mathbb{E}[J]$	$\theta_t^{k+1} = \alpha\theta_t^k + (1-\alpha) \sum_{n=1}^{N} \frac{\mathbf{1}_{\{J^n \leq \gamma\}} u_t^n}{\sum_{n'=1}^{N} \mathbf{1}_{\{J^{n'} \leq \gamma\}}}$

parameter $\tilde{\lambda}$ at every iteration to maintain the same threshold γ, which is a more intuitive parameter. In addition, we have observed that the reparameterized framework is easier to tune and achieves better performance than the original formulation. Therefore, we focus our analysis and simulations on only the reparameterized version hereon after. Note that in practice, it is easier to define an *elite fraction*, which adjusts γ based on the scale of the costs, instead of using γ for easier tuning.

Figure 1 illustrates the shapes of the function corresponding to different r and γ values. For $J < \gamma$, as $r \to \infty$, $\exp_r(-\tilde{\lambda}^{-1}J) \to 1$. Hence, $\exp_r(-\tilde{\lambda}^{-1}J)$ converges pointwise to the step function $1_{\{J \leq \gamma\}}$ with $r \to \infty$. On the other hand, for any $0 < \frac{J}{\gamma} < 1$, we have that

$$\frac{\exp_r(-\tilde{\lambda}^{-1}0)}{\exp_r(-\tilde{\lambda}^{-1}J)} = \exp\left(-\frac{1}{r-1}\log\left(1-\frac{J}{\gamma}\right)\right), \tag{29}$$

which tends to ∞ as $r \to 1$. Hence, $\exp\left(\frac{1}{r-1}\log\left(1-\frac{J}{\gamma}\right)\right)$ converges to $1_{\{J=0\}}$ as $r \to 1$.

B. Variance Reduction

With the connection to SS, the effect of Tsallis divergence can be analyzed through the equivalent problem formulation in optimization. In Section III-A, it is shown that Tsallis VI-SOC corresponds to an upper bound of SS-SOC with objective

$$\theta^* = \arg\max_{\theta} \mathbb{E}_{\pi(U;\theta)}\left[\exp_r(-\tilde{\lambda}^{-1}J)\right]. \tag{30}$$

The variance reduction effect can be analyzed through the coefficient of Absolute Risk Aversion (ARA) [33]

$$A(J) = -\frac{S''(J)}{S'(J)}, \tag{31}$$

for the optimization problem

$$\min_{\theta} \mathbb{E}_{p(X|U),\pi(U;\theta)}[S(J(X,U))], \tag{32}$$

where Tsallis VI-SOC corresponds to $S(J) = \exp_r(-\tilde{\lambda}^{-1}J)$. The ARA coefficient measures a scaled ratio of terms corresponding to the mean and variance terms in the Taylor series expansion of the objective. Negative value of the coefficient corresponds to a risk-averse objective, and positive value of the coefficient corresponds to risk-seeking behavior.[1] The ARA coefficient of Tsallis VI-SOC is

$$A_{\text{Tsallis}}(J) = -\frac{r-2}{(r-1)(\gamma - J)}. \tag{33}$$

The ARA coefficients for MPPI and CEM are

$$A_{\text{MPPI}}(J) = \frac{1}{\lambda}, \tag{34}$$

$$A_{\text{CEM}}(J) = \lim_{k\to\infty} -k\tanh\left(\frac{1}{2}k(\gamma - J)\right). \tag{35}$$

Since A_{MPPI} is a positive constant, it corresponds to a risk-seeking algorithm. For a cost below the elite threshold, $J < \gamma$, $A_{\text{CEM}} = -\infty$ and $A_{\text{Tsallis}} \lesseqgtr 0$ for $r \lesseqgtr 2$. This leads to the Tsallis VI-SOC framework achieving lower variance than MPPI. For the same elite threshold γ and a properly selected r, we hypothesize that Tsallis VI-SOC results in lower mean cost than CEM since CEM penalizes variance infinitely harder than the mean and assigns equal weights to all elite samples. A more detailed analysis and the derivation of ARA coefficient are included in Section SM-3 of the supplementary material.

V. Model Predictive Control Algorithm

With the update laws in Section II-D and a choice of the optimality likelihood function f, we propose the novel Tsallis VI-MPC algorithm, summarized in Algorithm 1.

Given the initial state distribution $p(x_0)$ and a prior policy distribution, $N \times M$ initial states are sampled. Given the initial states, N control trajectories are sampled from the policy distribution. For each control trajectory sample, M state trajectories are propagated for a total of $N \times M$ rollouts. The state transitions at each timestep are sampled from the stochastic dynamics $F(x, u, \epsilon)$ where ϵ corresponds to system stochasticity (zero mean Gaussian $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2 \mathbf{I})$ used in this paper). The cost of each trajectory is normalized to $[0, 1]$ for numerical stability and easier tuning. Depending on the choice of policy distribution class, the policy parameters are updated based on the update laws in Section II-D.

[1] ARA is originally used for utility maximization, where a larger ARA corresponds to greater risk aversion. This is the opposite in our case when we are performing cost minimization.

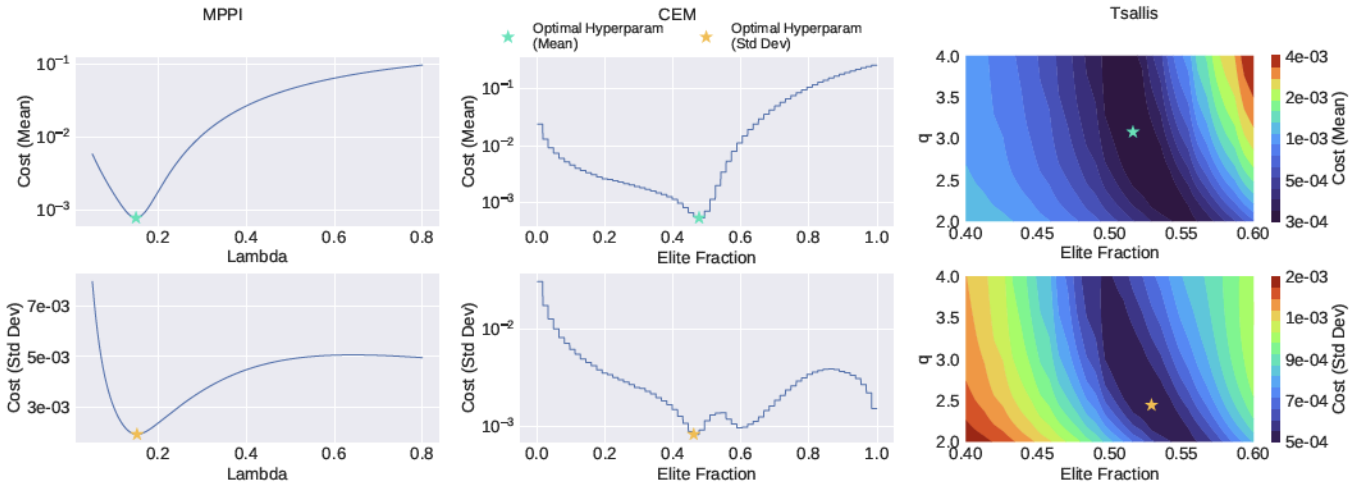

Fig. 2: Comparison of mean cost and cost variance for the numerical analysis system Eq. (36). The cost achieved using the updated u from a single update step, averaged over 4096 different seeds, is shown over the entire range of the hyperparameter set for CEM, MPPI and Tsallis VI-SOC. The hyperparameters which minimize the mean and standard deviation of the cost are shown as a cyan and orange star respectively.

Algorithm 1 Tsallis Variational Inference MPC

1: **Given:** $p(x_0)$: initial state distribution; N: number of policy samples; M: number of state samples; T: MPC horizon; T': number of MPC steps; K: optimization iterations per MPC step
2: Initialize Θ_0^0
3: $\{x_0^{n,m}\}_{n,m=1}^{N,M} \sim p(x_0)$
4: **for** $t' = 0$ to $T' - 1$ **do**
5: **for** $k = 0$ to K **do**
6: **for** $n = 1$ to N *in parallel* **do**
7: **for** $m = 1$ to M *in parallel* **do**
8: $X^{n,m}, U^n = f_{\text{rollout}}(x_0^{n,m}, \Theta_{t'}^k)$
9: $J^{n,m} = f_{\text{cost}}(X^{n,m}, U^n)$
10: **end for**
11: $J^n = \frac{1}{M} \sum_{m=1}^{M} f(\frac{J^{n,m} - \min J^{n,m}}{\max J^{n,m} - \min J^{n,m}})$
12: **end for**
13: $\Theta_{t'}^{k+1} = f_{\text{update}}(\{J^n, U^n\}_{n=1}^N)$
14: **end for**
15: Execute $q^K(u_{t',0})$
16: $\Theta_{t'+1}^0 = f_{\text{recede}}(\Theta_{t'}^K)$
17: **end for**

TABLE III: Comparison on a simple single stage stochastic optimization problem Eq. (36). The best values are boldfaced.

Algorithm	Cost (Mean)	Cost (Std Dev)	Mean Control Error
Tsallis	$\mathbf{2.65 \times 10^{-4}}$	$\mathbf{4.80 \times 10^{-4}}$	$\mathbf{2.20 \times 10^{-2}}$
CEM	5.36×10^{-4}	9.26×10^{-4}	5.04×10^{-2}
MPPI	7.95×10^{-4}	1.16×10^{-3}	4.33×10^{-2}

For the first iteration, we perform additional warm-up iterations by running the optimization loop (line 5 to 14) for a larger K_{warmup} iterations before executing the first control and performing K optimization iterations in the ensuing MPC steps.

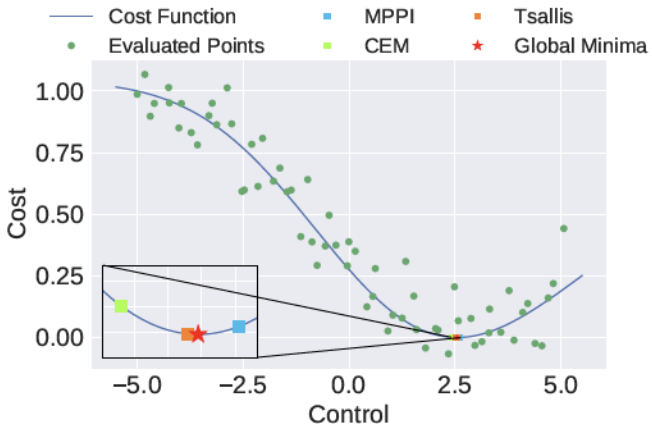

Fig. 3: A comparison of the updated means for CEM, MPPI and Tsallis VI-SOC using the best set of hyperparameters on a single realization of 64 noisy samples of u (green) for the numerical analysis system. Each sample is a noisy realization.

Control Selection: With the optimized policy distribution from the VI-SOC framework, the control to be executed on the real system is selected differently based on the choice of policy class. For the unimodal Gaussian policy, the mean of the distribution is used. For the Gaussian mixture policy, the mean of the model with the highest mixture weight is executed. In terms of the Stein policy, the Stein particle with the highest weight is used.

Receding Horizon: After the control execution, the policy distribution is shifted to warm start the optimization at the next MPC timestep. Let θ' be the next iteration's starting sequence and θ be the current iteration's sequence and set $\theta'_t = \theta_{t+1}$ for $t = 0, ..., T-2$. Finally, set the last item in the new sequence as $\theta'_{T-1} = \theta_{T-1}$. This is known as the receding horizon technique

in MPC. Note that in Algorithm 1, $\Theta = \{\theta_0, \cdots, \theta_{T-1}\}$.

VI. Simulations

In this section, we compare the proposed Tsallis VI-MPC algorithm against MPPI and CEM, which represent state-of-the-art sampling-based SOC algorithms. Since in practice, the shape functions used in SS-SOC corresponds to the ones which result in MPPI and CEM, we have chosen to only compare to MPPI and CEM. We first validate our analysis on a simple numerical experiment. We then showcase the scalability and performance of the Tsallis VI-MPC algorithm on 2D point mass, quadcopter, ant, manipulator and humanoid systems in simulation under the 3 aforementioned policy distributions.

A. Numerical Analysis

We verify the analysis in Section IV-B that Tsallis VI-SOC results in greater variance reduction by comparing the $\exp_r$ cost transform with the CEM and MPPI cost transforms on the following simple single-stage stochastic optimization problem:

$$\min_u \mathbb{E}_\xi \left[\frac{-\frac{\lambda}{2} e^{\frac{\lambda}{2}(\lambda\sigma^2 - 2u)} \operatorname{erf}\left(\frac{\lambda\sigma^2 - u}{\sqrt{2}\sigma}\right) - c}{d} + 0.1\xi \right], \quad (36)$$

where $\xi \sim \mathcal{N}(0,1)$, erf is the corresponding error function, the constants λ and σ are chosen to be $\lambda = 0.2, \sigma = 2.5$, and c and d are chosen such that the noiseless objective function is normalized between 0 and 1 for $u \in [-5, 5]$. Figure 3 plots the objective function near its minimum and the noisy objective values.

To ensure that only the weight computation of the three methods is tested, we sample 64 instances of u uniformly from $[-5, 5]$ and use the same set of samples for all three methods. We use the update law corresponding to the unimodal Gaussian with fixed variance for each algorithm and compare the resulting mean and standard deviation of the cost obtained from each method after a single optimization iteration. A grid search is performed over 4096 different hyperparameters, and the optimization metric is computed as the cost averaged over 4096 random seeds. The results for the best performing hyperparameters are shown in Table III and agree with our intuition that the objective function of Tsallis VI-SOC should result in a lower variance compared to MPPI and a lower mean compared to CEM. In addition, in Fig. 2, we observe that mean cost increases slower as the elite fraction decreases from the optimal value than when increases, while the opposite is true for cost standard deviation.

B. Controls and Robotics Systems

The dynamics for the planar navigation and quadcopter tasks are solved via an Euler discretization. Details of the dynamics can be found in Section SM-4. For the manipulator, ant, and humanoid tasks, we use the GPU-accelerated Isaac-Gym [34] to sample trajectory rollouts in parallel. Additional system stochasticity is injected to each system through the controls channel such that $F(x_t, u_t, \epsilon_t) = F(x_t, u_t + \epsilon_t)$.

The hyperparameters and system configurations for all simulations are included in Section SM-4. To ensure fair comparisons, all hyperparameters for each method are tuned using a combination of the TPE algorithm [35] from the Neural Network Intelligence (NNI) AutoML framework and hand tuning.

1) Planar Navigation: We first test the different algorithms on a point-mass planar navigation problem. The task is for the point-mass with double-integrator stochastic dynamics to navigate through an obstacle field to reach the target location. The dynamics and obstacle field are set up the same way as [6]. If a crash occurs, a crash cost is incurred and no further movement is allowed.

2) Quadcopter: We also set up a quadcopter 3-D navigation task. The task is for the quadcopter to reach a target location while avoiding obstacles. The quadcopter dynamics are taken from [36]. The quadcopter task is similiar to the planar navigation task, where the system is expected to fly through a randomly generated forest to reach the target location. If a crash occurs, a crash cost is incurred and no further movement is allowed.

3) Franka Manipulator: We next test on the Franka manipulator modified to have 7-DOF by removing the last joint and fixing the fingers. The objective of this task is to move the end effector around the obstacles to the goal. An illustration of the task is in Fig. 4. It is worth noting that no crash cost is in place for the Franka manipulator. Instead, contact is included as a part of the simulation dynamics.

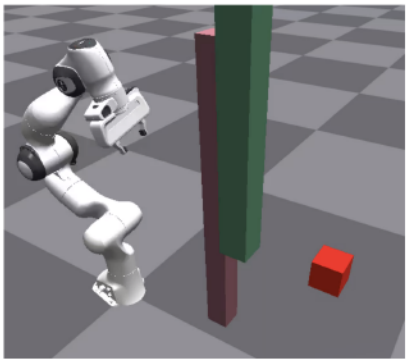

Fig. 4: Task setup for Franka manipulator. The goal is to reach the red block while avoiding the pink and green obstacles.

4) Ant: We also consider the task of locomotion. We test on the ant system, which has 29 state dimensions, 8 control dimensions, and is a widely used testbed for RL algorithms.

5) Humanoid: Finally, we test our approach on the complex humanoid locomotion task. The humanoid system has 56 state dimensions, 21 control dimensions, and has very unstable dynamics.

C. Discussion

In Table IV, we can see the results of each experiment summarized between each robotic system, policy parameterization, and choice of SOC framework. Across all systems we can

TABLE IV: Comparisons of mean and standard deviation of cost against MPPI and CEM on different systems and policy classes. The policy classes are defined as Unimodal Gaussian (UG), Gaussian Mixture (GM), and Stein (S). Note that the negative of the reward is used for the locomotion tasks (ant and humanoid). The best mean cost and cost variance for each system-policy distribution pair is boldfaced. The mean and standard deviation reduction percentages are included to the right where positive values correspond to a reduction.

		MPPI		CEM		Tsallis		Tsallis vs MPPI		Tsallis vs CEM	
System	Policy	Mean	Std	Mean	Std	Mean	Std	ΔMean%	ΔStd%	ΔMean%	ΔStd%
Planar Navigation	UG	30 023.7	3644.1	34 617.5	2523.3	**28 714.7**	**570.2**	4.4	84.4	17.1	77.4
	GM	39 313.8	8924.8	53 385.3	**5494.0**	**31 369.2**	5683.2	20.2	36.3	41.2	−3.4
	S	38 225.0	5169.9	38 847.6	6641.8	**33 324.5**	**4006.4**	12.8	22.5	14.2	39.7
Quadcopter	UG	15 266.2	**1374.0**	15 756.8	2707.1	**14 673.4**	1458.4	3.9	−6.1	6.9	46.1
	GM	22 145.5	4007.3	17 654.1	1593.9	**16 430.6**	**1498.2**	25.8	62.6	6.9	6.0
	S	29 238.3	5084.3	21 064.7	2218.0	**15 976.0**	**1218.3**	45.4	76.0	24.2	45.1
Franka	UG	64.2	8.4	66.4	18.4	**57.4**	**5.5**	10.6	34.5	13.6	70.1
	GM	67.2	8.3	68.7	13.0	**59.2**	**5.4**	12.0	34.9	13.9	58.5
	S	72.4	4.2	80.4	14.0	**71.8**	**3.3**	0.8	21.4	10.7	76.4
Ant	UG	−528.7	54.1	−652.6	**26.7**	**−692.9**	37.2	31.1	31.3	6.2	−39.0
	GM	−621.0	55.5	−659.1	53.6	**−663.8**	**36.3**	6.9	34.7	0.7	32.4
	S	−673.7	36.1	−670.0	32.2	**−677.4**	**13.4**	0.6	62.8	1.1	58.3
Humanoid	UG	−423.8	233.8	−660.8	208.5	**−899.3**	**80.1**	112.2	65.7	36.1	61.6
	GM	−592.9	170.2	−725.9	164.9	**−738.1**	**103.8**	24.5	39.0	1.7	37.1
	S	−794.7	174.0	−840.8	168.1	**−919.7**	**105.9**	15.7	39.2	9.4	37.0

see Tsallis VI-MPC results in lower means and variances in most policy parameterizations. The trend continues even as the complexity of the dynamics increases showing that the flexibility provided by the VI-MPC framework is useful even in high dimensional systems.

First in the planar navigation case, we see that Tsallis VI-MPC outperforms the other algorithms in almost all policy parameterizations. During testing, there was high variability in the trajectories computed by each algorithm. The best performing algorithms would have a large increase in velocity towards the goal state, and would have enough variation in sampled trajectories to "see" obstacles via large costs in order to avoid them. When comparing the trajectories computed through the GM policy parameterization we tend to see a reduced ability to stabilize at the goal resulting in larger costs overall when compared to other parameterizations. For these systems, the Stein policy results in longer trajectories to the goal. Next in the Quadcopter experiments, we observe that the Tsallis VI-MPC outperforms both MPPI and CEM in mean cost for the unimodal Gaussian and Stein policies. When looking at the high dimensional simulation environments, we see the same trend of Tsallis VI-MPC reducing the mean and variance in comparison to CEM and MPPI holds even for complex systems with contact dynamics.

The performance of the Tsallis VI-MPC can be attributed to the ability of the algorithm to sample policies that are low cost (via the elite fraction), but then improve beyond the CEM by performing the cost weighted averaging similar to MPPI. These characteristics of Tsallis VI-MPC are additionally heavily related to the choice of hyperparameters. From hand tuning, we observe that reducing the elite fraction generally results in lower mean cost but higher standard deviation and vice versa. As the cost transform approaches that of CEM or MPPI, the mean and standard deviation approaches their corresponding value. This verifies that the Tsallis VI-SOC framework can be thought of as an interpolation between CEM and MPPI to a certain degree. The best performing configuration is usually somewhere in between these two extremes.

Finally note that even though the Tsallis VI-SOC framework is typically used over multiple iterations in optimization schemes where the parameters can evolve as the optimization progresses, we have shown the benefits even in MPC mode, where a single iteration of optimization is performed, reiterating the empirical result that a single step has higher reduction in mean cost and variance compared to traditional MPC methods.

VII. Conclusion

We present a generalized Variational Inference-Stochastic Optimal Control framework using Tsallis divergence, which allows for additional control of the cost/reward transform and results in lower cost/reward variance. We provide a unifying study of the connections between Tsallis VI-SOC, MPPI, CEM, and SS methods. The performance and variance reduction benefits of the proposed Tsallis VI-SOC framework is verified analytically and numerically. We further showcase advantages of the Tsallis VI-MPC algorithm against MPPI and CEM on

5 different systems with 3 different policy distributions. We leave 2 extensions of this work as future research directions: 1. Implementation of the proposed algorithm on real systems; 2. Comparison against VI-MPC algorithms using other generalized divergences.

ACKNOWLEDGMENTS

This work is supported by ARO W911NF2010151, the NSF-CPS award #1932288 and NASA Langley. In addition, this work is supported by Sandia National Laboratories, a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.

REFERENCES

[1] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. Journal of the American statistical Association, 112(518):859–877, 2017.

[2] Cheng Zhang, Judith Bütepage, Hedvig Kjellström, and Stephan Mandt. Advances in variational inference. IEEE transactions on pattern analysis and machine intelligence, 41(8):2008–2026, 2018.

[3] Tanmay Shankar and Abhinav Gupta. Learning robot skills with temporal variational inference. In International Conference on Machine Learning, pages 8624–8633. PMLR, 2020.

[4] Emmanuel Pignat, Teguh Lembono, and Sylvain Calinon. Variational Inference with Mixture Model Approximation for Applications in Robotics. In 2020 IEEE International Conference on Robotics and Automation (ICRA), pages 3395–3401. IEEE, 2020.

[5] Masashi Okada and Tadahiro Taniguchi. Variational inference mpc for bayesian model-based reinforcement learning. In Conference on Robot Learning, pages 258–272. PMLR, 2020.

[6] Alexander Lambert, Adam Fishman, Dieter Fox, Byron Boots, and Fabio Ramos. Stein Variational Model Predictive Control. arXiv preprint arXiv:2011.07641, 2020.

[7] Yingzhen Li and Richard E Turner. Rényi Divergence Variational Inference. arXiv preprint arXiv:1602.02311, 2016.

[8] Adji B Dieng, Dustin Tran, Rajesh Ranganath, John Paisley, and David M Blei. Variational Inference via χ Upper Bound Minimization. arXiv preprint arXiv:1611.00328, 2016.

[9] Dilin Wang, Hao Liu, and Qiang Liu. Variational inference with tail-adaptive f-divergence. arXiv preprint arXiv:1810.11943, 2018.

[10] Jean-Baptiste Regli and Ricardo Silva. Alpha-beta divergence for variational inference. arXiv preprint arXiv:1805.01045, 2018.

[11] Neng Wan, Dapeng Li, and Naira Hovakimyan. f-Divergence Variational Inference. arXiv preprint arXiv:2009.13093, 2020.

[12] Constantino Tsallis. Possible generalization of Boltzmann-Gibbs statistics. Journal of statistical physics, 52(1):479–487, 1988.

[13] Constantino Tsallis, Murray Gell-Mann, and Yuzuru Sato. Asymptotically scale-invariant occupancy of phase space makes the entropy Sq extensive. Proceedings of the National Academy of Sciences, 102(43):15377–15382, 2005.

[14] Eric Lutz. Anomalous diffusion and Tsallis statistics in an optical lattice. Physical Review A, 67(5):051402, 2003.

[15] Bin Liu and J Goree. Superdiffusion and non-Gaussian statistics in a driven-dissipative 2D dusty plasma. Physical review letters, 100(5):055003, 2008.

[16] Ralph G DeVoe. Power-law distributions for a trapped ion interacting with a classical buffer gas. Physical review letters, 102(6):063001, 2009.

[17] Kyungjae Lee, Sungyub Kim, Sungbin Lim, Sungjoon Choi, and Songhwai Oh. Tsallis reinforcement learning: A unified framework for maximum entropy reinforcement learning. arXiv preprint arXiv:1902.00137, 2019.

[18] Constantino Tsallis. What are the numbers that experiments provide. Quimica Nova, 17(6):468–471, 1994.

[19] Evaldo MF Curado and Constantino Tsallis. Generalized statistical mechanics: connection with thermodynamics. Journal of Physics a: mathematical and general, 24(2): L69, 1991.

[20] Constantino Tsallis, RenioS Mendes, and Anel R Plastino. The role of constraints within generalized nonextensive statistics. Physica A: Statistical Mechanics and its Applications, 261(3-4):534–554, 1998.

[21] GL Ferri, S Martinez, and A Plastino. Equivalence of the four versions of tsallis's statistics. Journal of Statistical Mechanics: Theory and Experiment, 2005(04):P04009, 2005.

[22] Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. arXiv preprint arXiv:1608.04471, 2016.

[23] Jingwei Zhuo, Chang Liu, Jiaxin Shi, Jun Zhu, Ning Chen, and Bo Zhang. Message passing stein variational gradient descent. In International Conference on Machine Learning, pages 6018–6027. PMLR, 2018.

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

[25] George I Boutselis, Ziyi Wang, and Evangelos A Theodorou. Constrained sampling-based trajectory optimization using stochastic approximation. In 2020 IEEE International Conference on Robotics and Automation (ICRA), pages 2522–2528. IEEE, 2020.

[26] 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.

[27] Pieter-Tjerk De Boer, Dirk P Kroese, Shie Mannor, and Reuven Y Rubinstein. A tutorial on the cross-entropy method. Annals of operations research, 134(1):19–67, 2005.

[28] Shie Mannor, Reuven Y Rubinstein, and Yohai Gat. The cross entropy method for fast policy search. In Proceedings of the 20th International Conference on Machine Learning (ICML-03), pages 512–519, 2003.

[29] Marin Kobilarov. Cross-entropy randomized motion planning. In Robotics: Science and Systems, volume 7, pages 153–160, 2012.

[30] Grady Williams, Paul Drews, Brian Goldfain, James M Rehg, and Evangelos A Theodorou. Information-theoretic model predictive control: Theory and applications to autonomous driving. IEEE Transactions on Robotics, 34 (6):1603–1622, 2018.

[31] 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.

[32] Evangelos A Theodorou and Emanuel Todorov. Relative entropy and free energy dualities: Connections to path integral and kl control. In 2012 ieee 51st ieee conference on decision and control (cdc), pages 1466–1473. IEEE, 2012.

[33] John W Pratt. Risk aversion in the small and in the large. In Uncertainty in economics, pages 59–79. Elsevier, 1978.

[34] Miles Macklin, Kenny Erleben, Matthias Müller, Nuttapong Chentanez, Stefan Jeschke, and Viktor Makoviychuk. Non-smooth newton methods for deformable multi-body dynamics. ACM Transactions on Graphics (TOG), 38(5):1–20, 2019.

[35] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In 25th annual conference on neural information processing systems (NIPS 2011), volume 24. Neural Information Processing Systems Foundation, 2011.

[36] Winter Guerra, Ezra Tal, Varun Murali, Gilhyun Ryou, and Sertac Karaman. FlightGoggles: Photorealistic sensor simulation for perception-driven robotics using photogrammetry and virtual reality. In 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, November 2019. doi: 10.1109/iros40897.2019.8968116. URL https://doi.org/10.1109/iros40897.2019.8968116.

SM-1. Derivation of Optimal Distribution

To solve for the optimal distribution, we first formulate the Lagrangian as

$$\mathcal{L}(U, \lambda, \alpha) = \lambda^{-1}\mathbb{E}_{q(U)}[J] + \mathcal{D}_r\left(q(U) \parallel p(U)\right) \tag{37}$$

$$+ \alpha\left(1 - \int q(U)\mathrm{d}U\right), \tag{38}$$

which can be solved by setting $\frac{\partial \mathcal{L}}{\partial q} = 0$ as

$$\frac{\partial \mathcal{L}}{\partial q} = \lambda^{-1}J + \frac{r}{r-1}\left(\frac{q(U)}{p(U)}\right)^{r-1} - \alpha = 0 \tag{39}$$

$$\Rightarrow q^*(U) = \left(\frac{r-1}{r}\right)^{\frac{1}{r-1}}\left(\alpha - \lambda^{-1}J\right)^{\frac{1}{r-1}} p(U). \tag{40}$$

Setting $\lambda^{-1} = \alpha(r-1)\tilde{\lambda}^{-1}$, we get

$$q^*(U) = \left(\frac{\alpha(r-1)}{r}\right)^{\frac{1}{r-1}}\left(1 - (r-1)\tilde{\lambda}^{-1}J\right)^{\frac{1}{r-1}} p(U) \tag{41}$$

$$= \left(\frac{\alpha(r-1)}{r}\right)^{\frac{1}{r-1}} \exp_r\left(-\tilde{\lambda}^{-1}J\right) p(U). \tag{42}$$

We now use the constraint that $\int q(U)\,\mathrm{d}U = 1$ by integrating both sides to get

$$1 = \left(\frac{\alpha(r-1)}{r}\right)^{\frac{1}{r-1}} \int \exp_r\left(-\tilde{\lambda}^{-1}J\right) p(U)\,\mathrm{d}U \tag{43}$$

$$\Rightarrow \left(\frac{\alpha(r-1)}{r}\right)^{\frac{1}{r-1}} = \frac{1}{\int \exp_r\left(-\tilde{\lambda}^{-1}J\right) p(U)\,\mathrm{d}U}. \tag{44}$$

Plugging the normalizing constants back in we can get the optimal policy q^* to be

$$q^*(U) = \frac{\exp_r\left(-\tilde{\lambda}^{-1}J\right) p(U)}{\int \exp_r\left(-\tilde{\lambda}^{-1}J\right) p(U)\,\mathrm{d}U}. \tag{45}$$

SM-2. Derivation of Update Laws

For each of the following update laws, we make use of the following equality:

$$\pi^{k+1}(U) = \arg\min_{\pi(U)\in\Pi} \mathcal{D}_{KL}\left(\tilde{q}^*(U) \parallel \pi(U)\right) \tag{46}$$

$$= \arg\min_{\pi(U)\in\Pi} \int \tilde{q}^*(U)\log\tilde{q}^*(U)\mathrm{d}U - \int \tilde{q}^*(U)\log\pi(U)\mathrm{d}U. \tag{47}$$

We can drop the first term as it doesn't relate to the $\arg\min_{\pi(U)\in\Pi}$:

$$= \arg\min_{\pi(U)\in\Pi} -\int \tilde{q}^*(U)\log\pi(U)\mathrm{d}U \tag{48}$$

$$= \arg\max_{\pi(U)\in\Pi} \int \tilde{q}^*(U)\log\pi(U)\mathrm{d}U \tag{49}$$

$$= \arg\max_{\pi(U)\in\Pi} \sum_{n=1}^{N} w^{(n)}\log\pi(U^{(n)}). \tag{50}$$

A. Unimodal Gaussian

For the case of a policy class of unimodal Gaussian distributions where the controls $u_t \in \mathbb{R}^{n_u}$ of each timestep are independent, the p.d.f. for each timestep's control has the form

$$\mathcal{N}(u_t; \mu_t, \Sigma_t) = \frac{1}{\sqrt{(2\pi)^{n_u}|\Sigma_t|}} \exp\left(-\frac{1}{2}(u_t - \mu_t)^{\mathrm{T}} \Sigma_t^{-1}(u_t - \mu_t)\right). \tag{51}$$

where we have that

$$\nabla_{\mu_t} \log \mathcal{N}(u_t; \mu_t, \Sigma_t) = (u_t - \mu_t)^{\mathrm{T}} \Sigma_t^{-1} \tag{52}$$

$$\nabla_{\Sigma_t^{-1}} \log \mathcal{N}(u_t; \mu_t, \Sigma_t) = \frac{1}{2}\Sigma_t - \frac{1}{2}(u_t - \mu_t)(u_t - \mu_t)^{\mathrm{T}}. \tag{53}$$

Minimizing the KL divergence between $\pi(U)$ and $\tilde{q}^*(U)$ and using Eq. (46), we have

$$\pi^{k+1}(U) = \arg\min_{\pi(U)\in\Pi} \mathcal{D}_{KL}\left(\tilde{q}^*(U) \parallel \pi(U)\right) \tag{54}$$

$$= \arg\max_{\mu,\Sigma} \sum_{n=1}^{N} \sum_{t=0}^{T-1} w^{(n)} \log \mathcal{N}(U^{(n)}; \mu_t, \Sigma_t). \tag{55}$$

Hence, we have that

$$0 = \sum_{n=1}^{N} w^{(n)} \nabla_{\mu_t} \log \mathcal{N}(U^{(n)}; \mu_t, \Sigma_t) \tag{56}$$

$$0 = \sum_{n=1}^{N} w^{(n)} \nabla_{\Sigma_t} \log \mathcal{N}(u_t^{(n)}; \mu_t, \Sigma_t). \tag{57}$$

Using Eqs. (52) and (53) then results in the update laws

$$\mu_t = \sum_{n=1}^{N} w^{(n)} u_t^{(n)} \tag{58}$$

$$\Sigma_t = \sum_{n=1}^{N} w^{(n)} (u_t^{(n)} - \mu_t)(u_t^{(n)} - \mu_t)^{\mathrm{T}}. \tag{59}$$

B. Mixture of Gaussian

For a Gaussian mixture model $\pi(U; \theta)$ with L components with parameters $\theta := \{\theta_l\}_{l=1}^{L}$ and $\theta_l := \{(\phi_l, \{\mu_{l,t}, \Sigma_{l,t}\}_{t=0}^{T-1})$ such that

$$\pi(U; \theta) = \prod_{t=0}^{T-1} \pi_t(u_t; \theta) \tag{60}$$

$$\pi_t(u_t; \theta) := \sum_{l=1}^{L} \pi_m \mathcal{N}(u_t; \mu_l, \Sigma_l). \tag{61}$$

Directly trying to minimize the KL divergence between $\pi(U; \theta)$ and $\tilde{q}^*(U)$ and using Eq. (46) gives us

$$\theta^* = \arg\max_{\theta} \sum_{n=1}^{N} \sum_{t=0}^{T-1} \left(w^{(n)} \log \sum_{l=1}^{L} \phi_l \mathcal{N}(u_t^{(n)}; \mu_{l,t}, \Sigma_{l,t}) \right). \tag{62}$$

However, it is not straightforward to solve Eq. (62) directly.

Instead, we can use the expectation maximization algorithm and introduce the latent variables $\{Z_n\}_{n=1}^{N}$, $Z_n \in [L]$ which form a categorical distribution $q(Z)$ over each of the L mixtures and denotes which mixture each sample $U^{(n)}$ was sampled

from. Then, we can write $\log \pi(U;\theta)$ as

$$\log \pi(U;\theta) = \int q(z) \log \pi(U;\theta) \mathrm{d}z \tag{63}$$
$$= \int q(z) \log \frac{\pi(U;\theta) p(z|U;\theta)}{p(z|U;\theta)} \mathrm{d}z \tag{64}$$
$$= \int q(z) \log \frac{p(U,z;\theta)}{p(z|U;\theta)} \mathrm{d}z \tag{65}$$
$$= \int q(z) \log \frac{p(U,z;\theta)}{q(z)} \mathrm{d}z - \int q(z) \log \frac{p(z|U;\theta)}{q(z)} \mathrm{d}z \tag{66}$$
$$= F(q,\theta) + \mathcal{D}_{KL}\left(q \parallel p\right). \tag{67}$$

where $F(q,\theta)$ is the ELBO, and provides a lower bound for the log likelihood $\log p(U;\theta)$. Now, to optimize $F(q,\theta)$, we estimate $q(z)$ via $p(z|U;\theta^k)$, where θ^k is the previous iteration's parameters. Then, the ELBO becomes

$$F(q,\theta) = \int q(z) \log \frac{p(U,z;\theta)}{q(z)} \mathrm{d}z \tag{68}$$
$$= \int q(z) \log p(U,z;\theta) \mathrm{d}z - \int q(z) \log q(z) \mathrm{d}z \tag{69}$$
$$= \int p(z|U;\theta^k) \log p(U,z;\theta) \mathrm{d}z - \int p(z|U;\theta^k) \log p(z|U;\theta^k) \mathrm{d}z \tag{70}$$
$$= Q(\theta,\theta^k) + H(z|U). \tag{71}$$

Since the second term is a function of θ^k and thus is not dependent on θ, it suffices to optimize $Q(\theta,\theta^k)$. To do so, the EM algorithm separates this into two steps:

1) E-step: Compute $p(z|U;\theta^k)$.
2) M-step: Compute $\arg\max_\theta Q(\theta,\theta^k)$.

In our case, we have that

$$Q(\theta,\theta^k) = \int p(z|U;\theta^k) \log p(U,z;\theta) \mathrm{d}z \tag{72}$$
$$= \mathbb{E}_{z\sim p(z|U;\theta^k)} \left[\sum_{n=1}^{N} \sum_{l=1}^{L} \sum_{t=0}^{T-1} \mathbb{1}\{Z_n = l\} w^{(n)} \left(\log \phi_l + \log \mathcal{N}(u_t^{(n)} | \mu_{l,t}, \Sigma_{l,t}) \right) \right] \tag{73}$$
$$= \sum_{n=1}^{N} \sum_{l=1}^{L} \sum_{t=0}^{T-1} p(Z_n = l | u_t^{(n)}; \theta^k) w^{(n)} \left(\log \phi_l + \log \mathcal{N}(u_t^{(n)} | \mu_{l,t}, \Sigma_{l,t}) \right) \tag{74}$$
$$= \sum_{n=1}^{N} \sum_{l=1}^{L} \sum_{t=0}^{T-1} \eta_l(u_t^{(n)}) w^{(n)} \left(\log \phi_l + \log \mathcal{N}(u_t^{(n)} | \mu_{l,t}, \Sigma_{l,t}) \right). \tag{75}$$

where we have defined $\eta_l(u_t^{(n)}) \coloneqq p(Z_n = l | u_t^{(n)}; \theta^k)$ for simplicity. To compute this, note that

$$p(Z_k = l | u_t^{(n)}; \theta^k) = \frac{p(u | Z_n = l; \theta^k) p(Z_n = l; \theta^k)}{\sum_{l'=1}^{L} p(u | Z_n = l'; \theta^k) p(Z_n = l'; \theta^k)} \tag{76}$$
$$= \frac{\phi_l \mathcal{N}(u_t^{(n)}; \mu_l^k, \Sigma_l^k)}{\sum_{l'=1}^{L} \phi_l \mathcal{N}(u^{(n)}; \mu_{l'}^k, \Sigma_{l'}^k)}. \tag{77}$$

To solve for the optimal θ^* in the M-step, we apply the first order conditions to $\mu_{l,t}$ and $\Sigma_{l,t}$ to obtain

$$0 = \sum_{n=1}^{N} \eta_l(u_t^{(n)}) w^{(n)} (u_t^{(n)} - \mu_{l,t}^{k+1}) \Sigma_{l,t}^{-1} \tag{78}$$
$$\implies \mu_{l,t}^{k+1} = \frac{1}{N_{l,t}} \sum_{n=1}^{N} \eta_l(u_t^{(n)}) w^{(n)} u_t^{(n)}. \tag{79}$$

and

$$0 = \sum_{n=1}^{N} \eta_l(u_t^{(n)}) w^{(n)} \left(\frac{1}{2}\Sigma_{l,t}^{k+1} - \frac{1}{2}(u_{l,t} - \mu_{l,t}^{k+1})(u_{l,t} - \mu_{l,t}^{k+1})^{\mathrm{T}} \right) \tag{80}$$

$$\implies \Sigma_{l,t}^{k+1} = \frac{1}{N_{l,t}} \sum_{n=1}^{N} \eta_l(u_t^{(n)}) w^{(n)} (u_t^{(n)} - \mu_{l,t}^{k+1})(u_t^{(n)} - \mu_{l,t}^{k+1})^{\mathrm{T}}. \tag{81}$$

where we have defined $N_{l,t} \coloneqq \sum_{n=0}^{N} \eta_l(u_t^{(n)}) w^{(n)}$ for convenience. Since ϕ_l^* has the constraint $\sum_{l=1}^{L} \phi_l^* = 1$, we solve for ϕ_l^* by applying the Lagrangian multiplier λ, yielding the following Lagrangian:

$$L_t = \sum_{n=1}^{N} \sum_{l=1}^{L} \sum_{t=0}^{T-1} \eta_l(u_t^{(n)}) w^{(n)} \big(\log \phi_l + \log \mathcal{N}(u_t^{(n)} | \mu_{l,t}, \Sigma_{l,t}) + \lambda \left(1 - \sum_{l=1}^{L} \phi_l \right). \tag{82}$$

Applying first order conditions for ϕ_l gives

$$0 = \sum_{n=1}^{N} \sum_{t=0}^{T-1} \eta_l(u_t^{(n)}) w^{(n)} \frac{1}{\phi_l^{k+1}} - \lambda \tag{83}$$

$$\implies \phi_l^{k+1} = \frac{1}{\lambda} \sum_{n=1}^{N} \sum_{t=0}^{T-1} \eta_l(u_t^{(n)}) w^{(n)} \tag{84}$$

$$= \frac{1}{\lambda} \sum_{t=0}^{T-1} N_{l,t}. \tag{85}$$

To solve for the Lagrange multiplier λ, we plug in the constraint to get

$$\lambda = \sum_{l=1}^{L} \sum_{n=1}^{N} \sum_{t=0}^{T-1} \eta_l(u_t^{(n)}) w^{(n)} = \sum_{l=1}^{L} \sum_{t=1}^{T-1} N_{l,t}. \tag{86}$$

Hence, we finally get that

$$\phi_l = \frac{N_l}{\sum_{l'=1}^{L} N_{l'}}. \tag{87}$$

where we have defined $N_l \coloneqq \sum_{t=0}^{T-1} N_{l,t}$.

C. Non-parametric Policy via SVGD

We now derive the update law for the choice of a fully non-parametric policy via SVGD. SVGD solves the variational inference problem

$$\pi^* = \arg\min_{\pi \in \Pi} \{ \mathcal{D}_{KL} \left(\pi(U) \parallel q^*(U) \right) \}. \tag{88}$$

over the set $\Pi \coloneqq \{ z | z = T(x) \}$ consisting of all distributions obtained by smooth transforms T of random variables x, where x is drawn from some tractable reference distribution.

One can show that the direction of steepest descent ϕ^* which maximizes the negative gradient $-\nabla_\epsilon \mathcal{D}_{KL} \left(q_{[T]} \parallel p \right) |_{\epsilon=0}$ in zero-centered balls in the RKHS $\mathcal{H}^d$ has the form

$$\phi^*(\cdot) = \mathbb{E}_{U \sim \pi} \left[k(U, \cdot) \nabla_U \log q^*(U) + \nabla_U k(U, \cdot) \right]. \tag{89}$$

Hence, one can then compute the optimal distribution π^* by iteratively applying ϕ^* to some initial distribution p_0. However, given that we only have an empirical approximation $\tilde{q}^*$ of the optimal distribution q^*, the gradient of $\log q^*$ cannot be easily computed. To solve this, following [6], instead of optimizing directly over the distribution of *controls* U, one can instead optimize over some distribution g of *parameters* θ of a parametrized policy $\hat{\pi}(U; \theta)$, such that

$$\pi(U) = \mathbb{E}_{\theta \sim g} \left[\hat{\pi}_\theta(U) \right] \quad \text{and} \quad q^*(\theta) = \mathbb{E}_{U \sim \hat{\pi}(\cdot;\theta)} \left[q^*(U) \right]. \tag{90}$$

By doing so, we obtain the following new VI problem and steepest descent direction respectively:

$$g^* = \arg\min_{\theta \in \Theta} \{ \mathcal{D}_{KL} \left(g(\theta) \parallel q^*(\theta) \right) \} \tag{91}$$

$$\hat{\phi}^*(\cdot) = \mathbb{E}_{\theta \sim g} \Big[\hat{k}(\theta, \cdot) \nabla_\theta \log \mathbb{E}_{U \sim \hat{\pi}(\cdot;\theta)} [q^*(U)] + \nabla_\theta \hat{k}(\theta, \cdot) \Big], \tag{92}$$

As a result, we can now take the gradient of $\log \mathbb{E}_{U\sim\hat{\pi}(\cdot;\theta)}[q^*(U)]$ as

$$\nabla_\theta \log \mathbb{E}_{U\sim\hat{\pi}(\cdot;\theta)}[q^*(U)] = \frac{\nabla_\theta \mathbb{E}_{U\sim\hat{\pi}(\cdot;\theta)}[q^*(U)]}{\mathbb{E}_{U\sim\hat{\pi}(\cdot;\theta)}[q^*(U)]} \tag{93}$$

$$= \frac{\int q^*(U)\nabla_\theta \hat{\pi}(U;\theta)\,\mathrm{d}U}{\mathbb{E}_{U\sim\hat{\pi}(\cdot;\theta)}[q^*(U)]} \tag{94}$$

$$= \frac{\mathbb{E}_{U\sim\hat{\pi}(\cdot;\theta)}[q^*(U)\nabla_\theta \log \hat{\pi}(U;\theta)]}{\mathbb{E}_{U\sim\hat{\pi}(\cdot;\theta)}[q^*(U)]}. \tag{95}$$

Discretizing the above since we only have a discrete approximation $\tilde{q}^*$ of q^*, we get that

$$\nabla_\theta \log \mathbb{E}_{U\sim\hat{\pi}(\cdot;\theta)}[q^*(U)] = \frac{\sum_{n=1}^N \tilde{q}^*(U^{(n)})\nabla_\theta \log \hat{\pi}(U^{(n)};\theta)}{\sum_{n=1}^N \tilde{q}^*(U^{(n)})}. \tag{96}$$

Hence, by letting g be an empirical distribution $g(\theta) = \sum_{l=1}^L \mathbf{1}_{\{\theta_l\}}$, we obtain the following update rule for the particles $\{\theta_l\}_{l=1}^L$:

$$\theta_l^{k+1} = \theta_l^k + \hat{\phi}^*(\theta_l^k). \tag{97}$$

SM-3. Variance Reduction Analysis

The variance reduction analysis is conducted through the connection between Tsallis VI-SOC, MPPI, CEM, and SS-SOC. With the appropriate selection of the shape function, the update laws of Tsallis VI-SOC, MPPI and CEM can be derived from SS-SOC. Associated with the SS-SOC update laws are their corresponding problem formulation in stochastic optimization. For the reparameterized Tsallis VI-SOC framework, its associated problem formulation in optimization is

$$\theta^* = \arg\max_\theta \mathbb{E}_{p(u;\theta)}\left[\exp\left(\frac{1}{r-1}\log\left(1-\frac{J}{\gamma}\right)\right)\right]. \tag{98}$$

Note that the scenario where $J \geq \gamma$ is ignored since the Taylor series expansion and analysis will be around $0 < J < \gamma$. Similarly, the optimization problem formulation corresponding to MPPI is

$$\theta^* = \arg\max_\theta \mathbb{E}_{p(u;\theta)}\left[\exp(-\lambda^{-1}J)\right]. \tag{99}$$

For CEM, the corresponding problem formulation is

$$\theta^* = \arg\max_\theta \mathbb{E}_{p(u;\theta)}\left[\mathbf{1}_{J\leq\gamma}\right]. \tag{100}$$

To analyze degree of risk aversion of each problem formulation, we can look at its Taylor series expansion. For a cost likelihood function $c(J_\theta)$ where J_θ is a stochastic cost term parametrized by the parameter θ, the Taylor series expansion around $\mathbb{E}[J_\theta] \coloneqq \tilde{J}_\theta$ has the form:

$$c(J_\theta) = c(\tilde{J}_\theta) + c'(\tilde{J}_\theta)\left(J_\theta - \tilde{J}_\theta\right) + \frac{1}{2}c''(\tilde{J}_\theta)\left(J_\theta - \tilde{J}_\theta\right)^2 + O(J_\theta^3). \tag{101}$$

Taking the expectation:

$$\mathbb{E}[c(J_\theta)] \approx c(\tilde{J}_\theta) + \frac{1}{2}c''(\tilde{J}_\theta)\mathrm{Var}(J_\theta). \tag{102}$$

On the other hand, if we define the certainty equivalent cost $J_{\theta CE}$ such that $c(J_{\theta CE}) = \mathbb{E}[c(J_\theta)]$, we have that

$$f(J_{\theta CE}) = c(\tilde{J}_\theta) + c'(\tilde{J}_\theta)\left(J_{\theta CE} - \tilde{J}_\theta\right) + O(J_{\theta CE}^2). \tag{103}$$

Hence,

$$c'(\tilde{J}_\theta)\left(J_{\theta CE} - \tilde{J}_\theta\right) \approx \frac{1}{2}c''(\tilde{J}_\theta)\mathrm{Var}(J_\theta). \tag{104}$$

Defining $\pi_A \coloneqq \tilde{J}_\theta - J_{\theta CE}$ as the *Absolute Risk Premium* (which is a **negative** quantity for **cost** minimization as opposed to a **positive** quantity for **utility/reward** maximization), we now have that

$$\pi_A \approx \frac{-1}{2}\frac{c''(\tilde{J}_\theta)}{c'(\tilde{J}_\theta)}\mathrm{Var}(J_\theta) \tag{105}$$

$$= \frac{1}{2}A(\tilde{J}_\theta)\mathrm{Var}(J_\theta), \tag{106}$$

where $A(\cdot)$ is the ARA coefficient, defined by

$$A(J) = -\frac{c''(J)}{c'(J)}. \tag{107}$$

The absolute risk premium measures the difference between the mean cost and average cost scaled by the cost transform as a function of the cost function variance. Therefore, the ARA coefficient indicates the degree of risk aversion with respect to the cost function variance.

The ARA coefficients for Tsallis VI-SOC and MPPI can be easily computed from equation 107 as

$$A_{\text{Tsallis}}(J) = -\frac{r-2}{(r-1)(\gamma - J)} \tag{108}$$

$$A_{\text{MPPI}}(J) = \frac{1}{\lambda}. \tag{109}$$

Since CEM's problem formulation includes the non-differentiable indicator function, we cannot directly apply the same analysis. However, by taking the indicator function as the limit of the sigmoid function σ

$$1_{\{J \leq \gamma\}} = \lim_{k \to \infty} \sigma(-k(J - \gamma)), \tag{110}$$

we can state the CEM problem formulation as

$$\theta^* = \arg\max_{\theta} \mathbb{E}_{p(u;\theta)} \left[\lim_{k \to \infty} \sigma(-k(J - \gamma)) \right]. \tag{111}$$

With this, the ARA coefficient can be computed as

$$A_{\text{CEM}}(J) = \lim_{k \to \infty} -k \tanh\left(\frac{1}{2}k(\gamma - J)\right). \tag{112}$$

SM-4. Simulation Details

A table of the system parameters is shown in Table V.

A. Planar Navigation

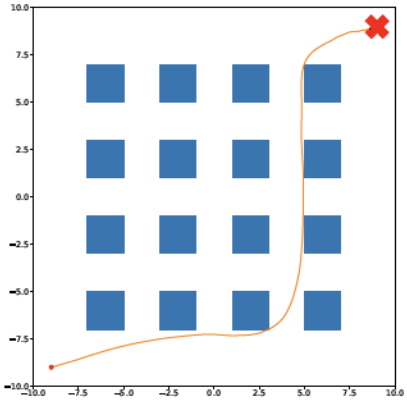

Fig. 5: Setup for the Planar Navigation task. The goal is for the robot (orange dot) to reach the goal location (red cross) while avoiding the obstacles (blue squares) in the middle. A crash cost of 10000 is incurred for crashing into the obstacles.

We consider the task of navigating a $2D$ double integrator through an obstacle field (see Fig. 5). The state consists of $(x_t, \dot{x}_t)$, while the control is $u_t = \ddot{x}_t$, where $x_t, \dot{x}_t, \ddot{x}_t \in \mathbb{R}^2$ represent the position, velocity and acceleration of the system. For the initial position $x_0 \coloneqq (\begin{bmatrix}-9 & -9\end{bmatrix}^{\mathrm{T}}, \begin{bmatrix}0 & 0\end{bmatrix}^{\mathrm{T}})$ and goal position $x_{goal} \coloneqq (\begin{bmatrix}9 & 9\end{bmatrix}^{\mathrm{T}}, \begin{bmatrix}0 & 0\end{bmatrix}^{\mathrm{T}})$, we defined the quadratic cost function:

$$J(X, U) = 10000 c_{\text{crash}} + (x_T - x_{\text{goal}})^{\mathrm{T}} Q_f (x_T - x_{\text{goal}}) + \sum_{t=0}^{T-1} (x_t - x_{\text{goal}})^{\mathrm{T}} Q_t (x_t - x_{\text{goal}}) + u_t^{\mathrm{T}} R u_t, \tag{113}$$

where c_{crash} is the indicator variable for crashing into an obstacle, and

$$Q_t \coloneqq \text{diag}(\begin{bmatrix}0.5 & 0.5 & 0.2 & 0.2\end{bmatrix}) \tag{114}$$

$$Q_T \coloneqq \text{diag}(\begin{bmatrix}0.25 & 0.25 & 1 & 1\end{bmatrix}) \tag{115}$$

$$R \coloneqq \text{diag}(\begin{bmatrix}0.01 & 0.01\end{bmatrix}). \tag{116}$$

B. Quadrotor

The quadrotor task includes navigating a 3D obstacle course with 35 randomly placed obstacles while trying to reach a target location and hover. The control of the quadrotor is done through angular rates and a thrust command, we assume a low level tracking controller that is common for flight systems. The state of the quadrotor x is composed of the 3 coordinates x, y, z angular position represented in quaternions and linear and angular velocities for a total of 13 states. The location of target is represented in cartersian coordinates by x_{target} and is located at (25,25,5). The initial location x_0 of the quadrotor is at (0,0,5). For this task, we use the following quadratic cost function:

$$J(X,U) = \sum_{t=0}^{T-1} 40\|x_{target} - x\| + 10\|\dot{x}\| + 2\|\omega\| + 1e7 c_{crash}, \tag{117}$$

where x is the cartesian coordinates of the quadrotor, $\dot{x}$ is the linear velocity, ω is the angular rates, and c_{crash} is the indicator function that is 1 if the following conditions are met, and 0 otherwise

1) If the vehicle gets within 0.75 of an obstacle.
2) If the vehicle goes above 10 meters or below 0 meters in the z axis.

C. Franka

The Franka manipulator is modified to have 7-DOF by removing the last joint and fixing the fingers in place. For state $x := (e, \dot{e})$ and control $u := \dot{q}$, where $e, \dot{e} \in \mathbb{R}^3$ are the position and velocity of the end-effector, and $\dot{q} \in \mathbb{R}^7$ are the joint velocities, the objective is to reach the goal position x_{goal} with obstacles between the starting position of the end effector and the goal. For this task, we use the following quadratic cost function:

$$J(X,U) = \sum_{t=0}^{T-1} 5\|e_t - x_{\text{goal}}\| + 0.1\|\dot{e}_t\|. \tag{118}$$

D. Ant

For the Ant, we take inspiration from the cost function for the HalfCheetah from [5] and modify the cost function to reward forward velocity. For the instantaneous x velocity v_t, torso height h_t and angle to the vertical axis α, the cost function J is defined as

$$J(X,U) = \sum_{t=0}^{T-1} -v_t \mathbf{1}_{\{h_t \leq 0.7\}} \mathbf{1}_{\{|\alpha| \geq 0.5\}} + 0.005\|u_t\|^2. \tag{119}$$

E. Humanoid

Similar to the Ant, we modify the cost function to reward forward velocity only if the height of the agent's root is not too close or far from the ground and the humanoid is oriented upwards. For the instantaneous x velocity v_t, torso height h_t and angle to the vertical axis α,

$$J(X,U) = \sum_{t=0}^{T-1} -v_t \mathbf{1}_{\{0.85 \leq h_t \leq 1.4\}} \mathbf{1}_{\{|\alpha| \geq 0.5\}} + 0.005\|u_t\|^2. \tag{120}$$

TABLE V: System parameters

System	Episode Length	Trials	State Space ($x_t \in$)	Control Space ($u_t \in$)	Timestep (Δt)	System Noise (σ)
Planar Navigation	300	100	$\mathbb{R}^4$	$\mathbb{R}^2$	0.01	1.0
Quadcopter	400	100	$\mathbb{R}^{13}$	$\mathbb{R}^4$	0.015	6.67
Franka	80	8	$\mathbb{R}^{14}$	$\mathbb{R}^7$	0.005	0.5
Ant	128	8	$\mathbb{R}^{29}$	$\mathbb{R}^8$	0.05	0.005
Humanoid	128	8	$\mathbb{R}^{55}$	$\mathbb{R}^{21}$	0.033	0.05

TABLE VI: Optimization Hyperparameters Per Dynamics

System	Warmup Iterations	Iterations per Timestep	Number of Samples	MPC Horizon T
Planar Navigation	8	1	256	96
Quadcopter	64	1	1024	150
Franka	32	2	256	20
Ant	32	4	256	20
Humanoid	32	4	1024	20

TABLE VII: Hyperparameters for Unimodal Gaussian

System	Optimizer	Control Std Dev (σ)	λ^{-1}	Elite Fraction	r
Planar Navigation	MPPI	18.000	0.015		
	CEM	7.780		0.098	
	Tsallis	18.667		0.070	1.796
Quadcopter	MPPI	0.934	3.970		
	CEM	1.056		0.015	
	Tsallis	1.450		0.140	6.276
Franka	MPPI	0.720	54.794		
	CEM	0.629		0.038	
	Tsallis	0.790		0.167	1.989
Ant	MPPI	0.130	54.000		
	CEM	0.500		0.050	
	Tsallis	0.536		0.044	5.179
Humanoid	MPPI	0.587	19.559		
	CEM	0.378		0.019	
	Tsallis	0.540		0.018	5.933

TABLE VIII: Hyperparameters for Gaussian Mixture Model

System	Optimizer	Number of Mixtures L	λ^{-1}	Elite Fraction	r
Planar Navigation	MPPI	4	3.051		
	CEM	4		0.770	
	Tsallis	4		0.036	33.294
Quadcopter	MPPI	4	0.650		
	CEM	4		0.015	
	Tsallis	4		0.010	1.040
Franka	MPPI	4	37.031		
	CEM	4		0.027	
	Tsallis	4		0.450	1.150
Ant	MPPI	4	16.943		
	CEM	4		0.050	
	Tsallis	4		0.024	5.000
Humanoid	MPPI	4	65.221		
	CEM	4		0.013	
	Tsallis	4		0.037	6.281

TABLE IX: Hyperparameters for Stein Policy

System	Optimizer	Number of Particles N	Kernel Bandwidth Multiplier	λ^{-1}	Elite Fraction	r
Planar Navigation	MPPI	8	1.030	71.200		
	CEM	8	0.462		0.023	
	Tsallis	8	9.444		0.382	3.420
Quadcopter	MPPI	8	2.869	3.829		
	CEM	8	12.902		0.071	
	Tsallis	8	2.320		0.047	2.077
Franka	MPPI	8	2.569	50.160		
	CEM	8	11.188		0.229	
	Tsallis	8	7.100		0.370	2.000
Ant	MPPI	8	6.683	8.160		
	CEM	8	7.100		0.150	
	Tsallis	8	7.100		0.400	2.500
Humanoid	MPPI	16	9.157	63.527		
	CEM	16	14.816		0.021	
	Tsallis	16	6.000		0.100	3.000

SM-5. ADDITIONAL RESULTS

In this section we include the results of different framework's sensitivity with respect to its hyperparpameters on the planar navigation example with fixed variance unimodal Gaussian policy in Table X. We also include the cumulative cost comparison plots of the simulator systems. All plots in this section show error bars of ± 1 standard deviation of the cost.

TABLE X: Sensitivity Results for hyperparameters on planar navigation example with fixed variance unimodal Gaussian policy. Tsallis mean variation results are bolded. Note that Tsallis hyperparameters demonstrate lower cost sensitivity when perturbed from their baseline values.

	MPPI		CEM		Tsallis			
	λ		γ		γ		r	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
+10%	30 294.44	3499.19	35 557.07	1753.46	28 806.34	531.61	28 873.70	572.90
Baseline	30 023.73	3644.07	34 617.47	2523.34	28 714.70	570.20	28 714.70	570.20
-10%	30 143.85	4273.51	34 427.06	2654.51	28 571.64	591.37	28 535.40	657.00
Mean Variation (%)	0.65	6.65	1.08	−12.66	**−0.09**	**−1.53**	**−0.04**	**7.85**

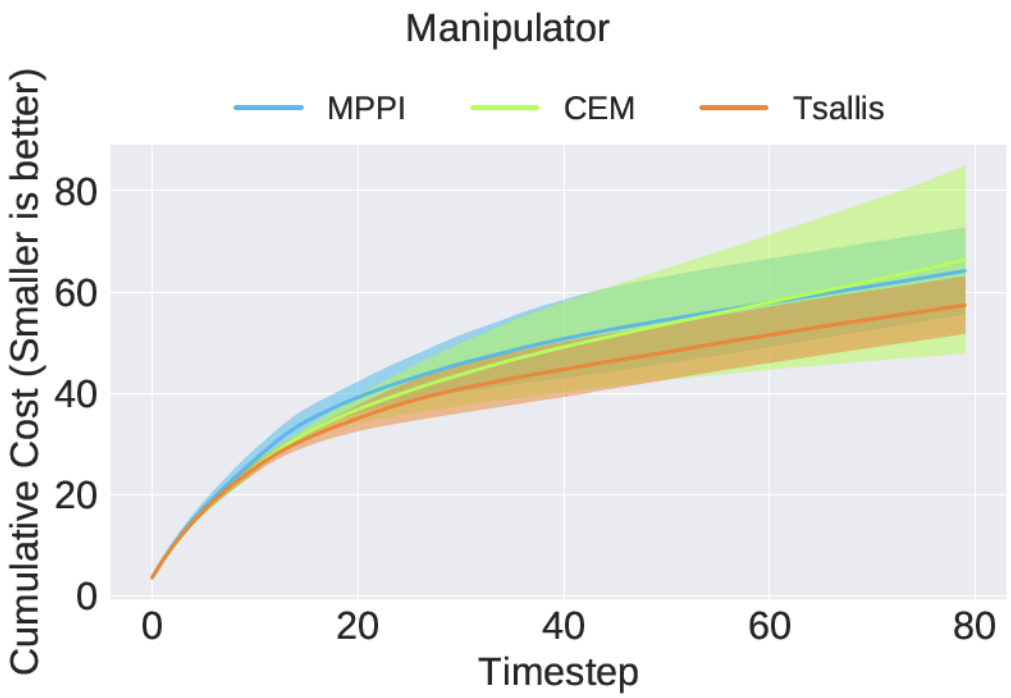

Fig. 6: Unimodal Gaussian on Manipulator

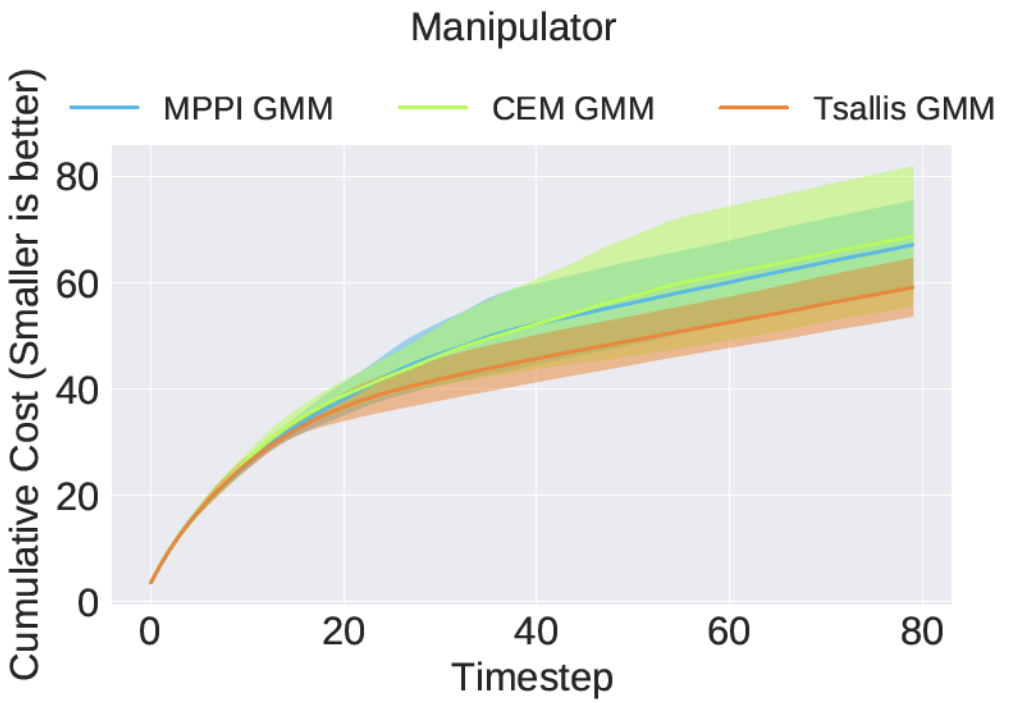

Fig. 7: Gaussian Mixture on Manipulator

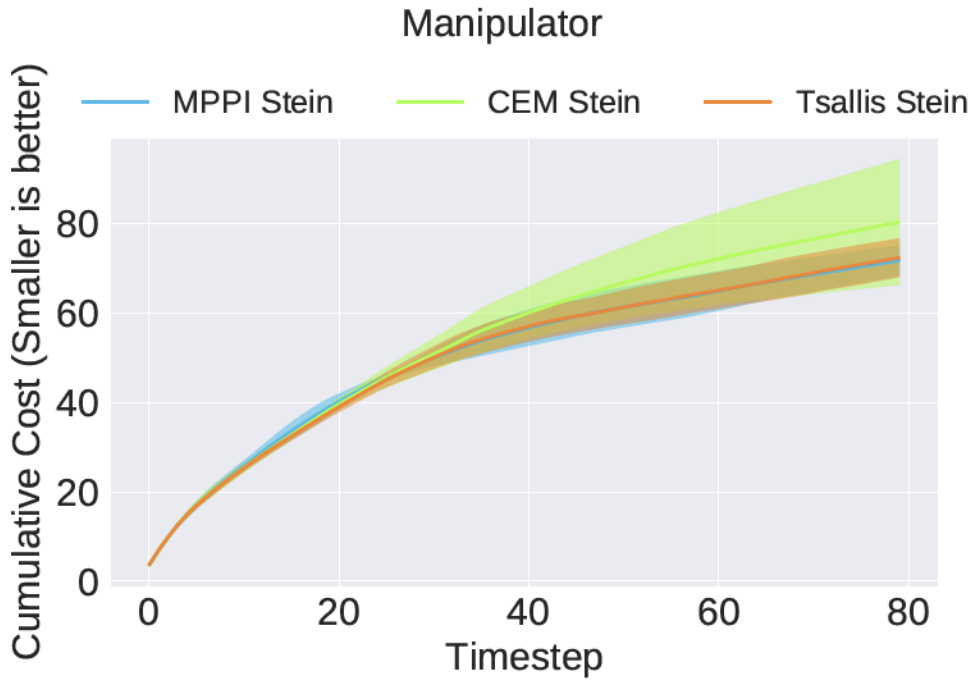

Fig. 8: Nonparametric Stein Policy on Manipulator

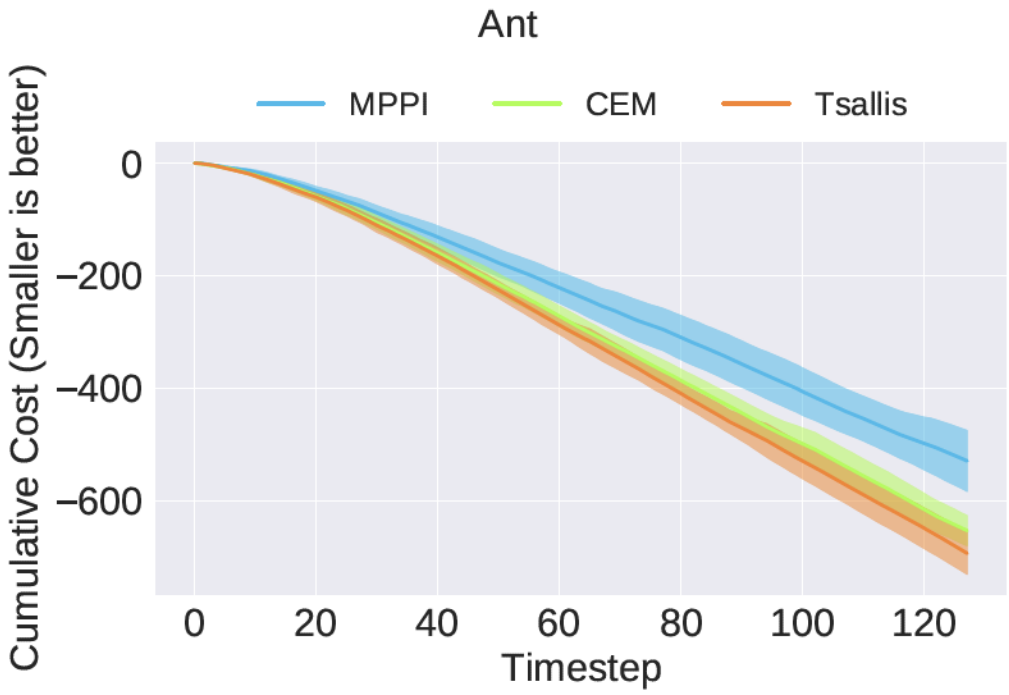

Fig. 9: Unimodal Gaussian on Ant

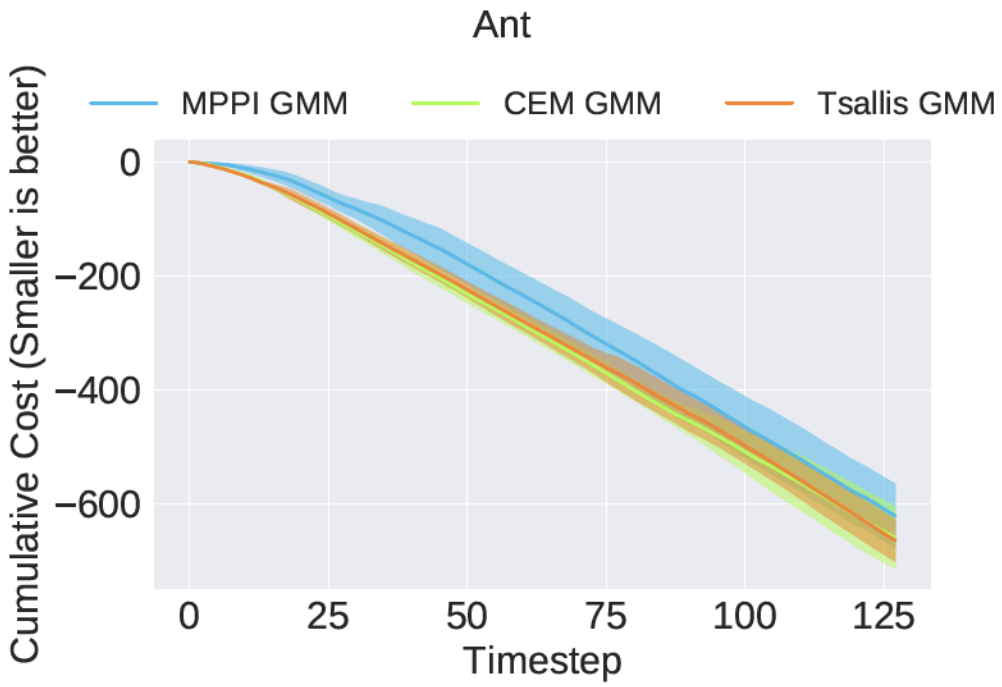

Fig. 10: Gaussian Mixture on Ant

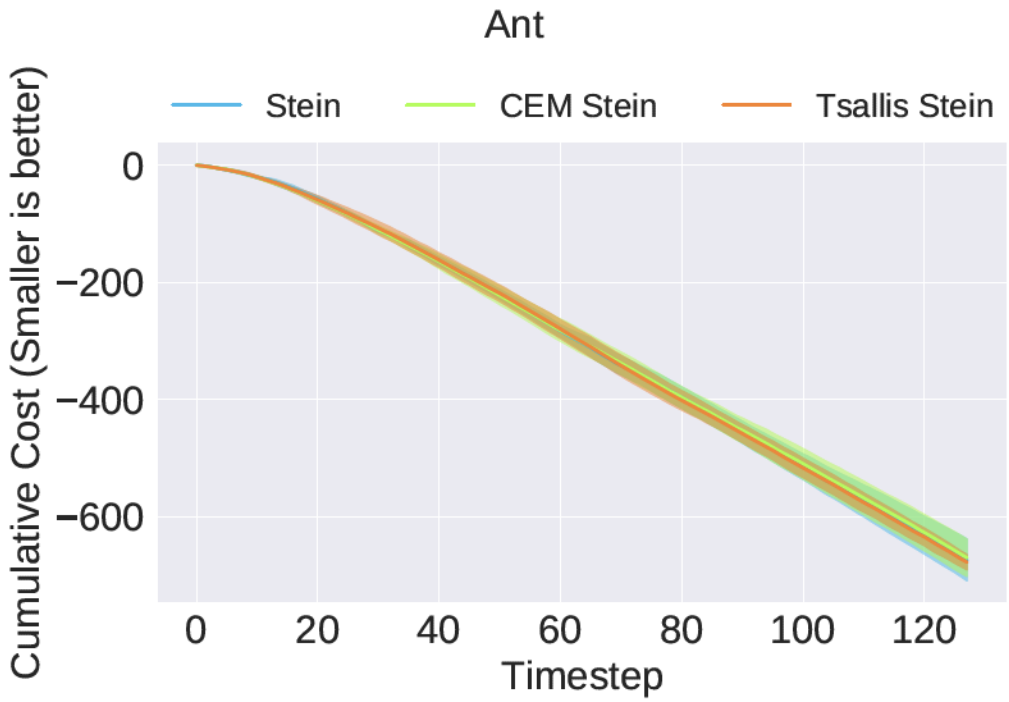

Fig. 11: Nonparametric Stein Policy on Ant

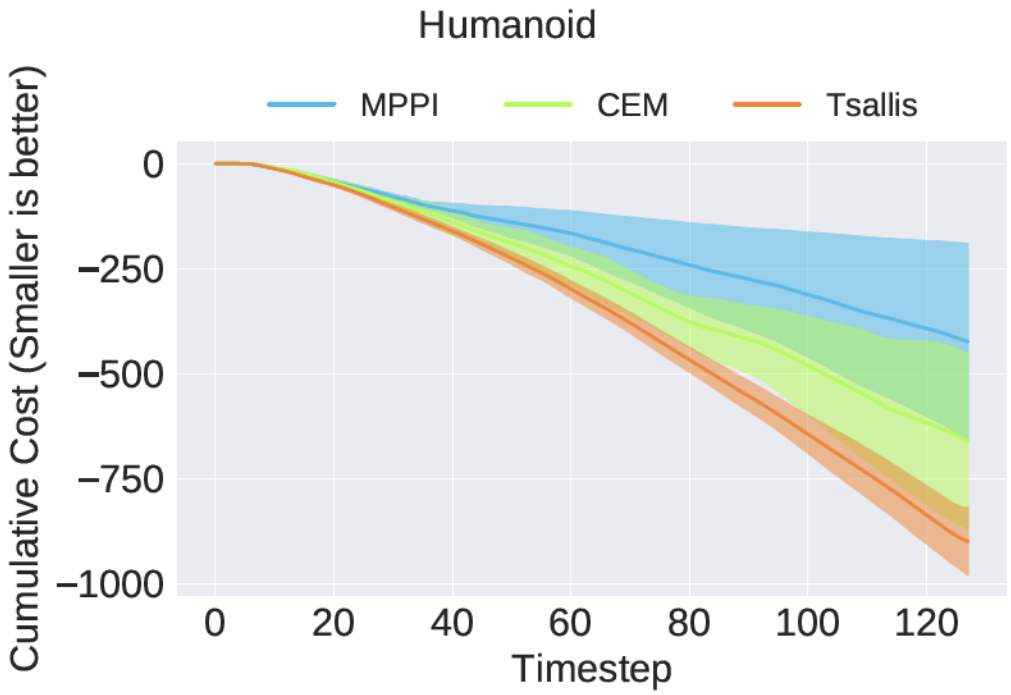

Fig. 12: Unimodal Gaussian on Humanoid

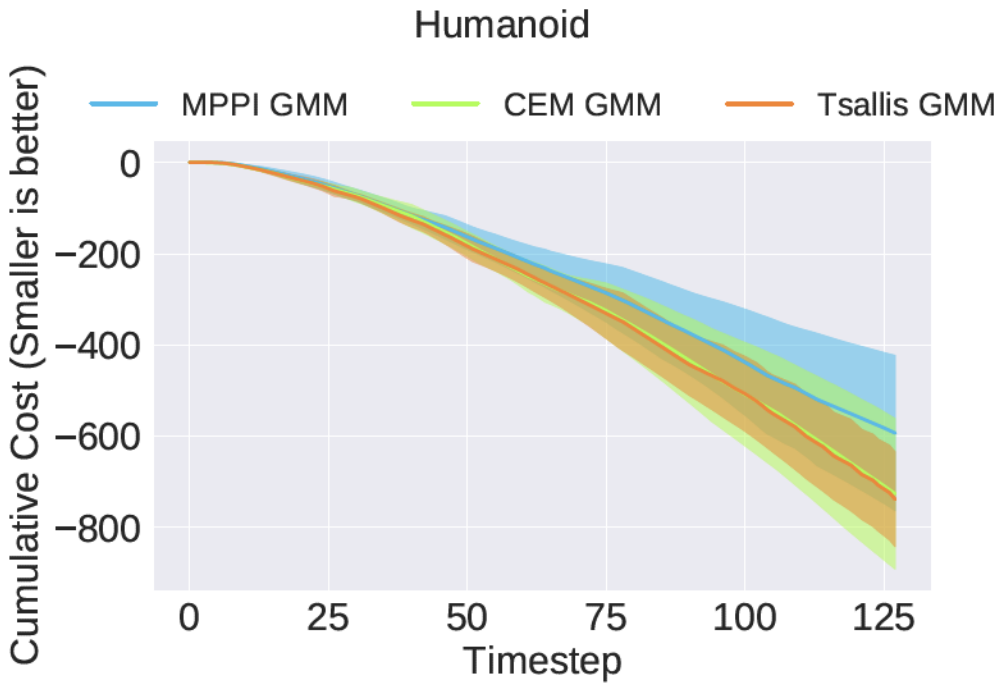

Fig. 13: Gaussian Mixture on Humanoid

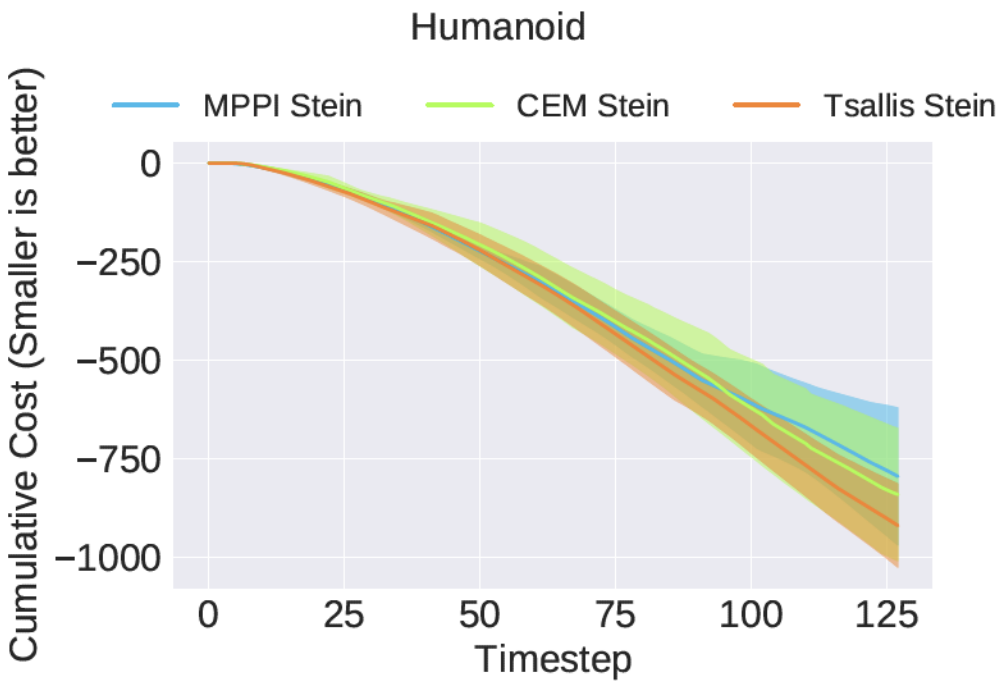

Fig. 14: Nonparametric Stein Policy on Humanoid