Bregman Centroid Guided Cross-Entropy Method

Anonymous authors
Paper under double-blind review

Abstract

The Cross-Entropy Method (CEM) is a widely adopted trajectory optimizer in model-based reinforcement learning (MBRL), but its unimodal sampling strategy often leads to premature convergence in multimodal landscapes. In this work, we propose **$\mathcal{B}$regman-$\mathcal{C}$entroid Guided CEM ($\mathcal{BC}$-EvoCEM)**, a lightweight enhancement to ensemble CEM that leverages *Bregman centroids* for principled information aggregation and diversity control. BC-EvoCEM computes a performance-weighted Bregman centroid across CEM workers and updates the least contributing ones by sampling within a trust region around the centroid. Leveraging the duality between Bregman divergences and exponential family distributions, we show that BC-EvoCEM integrates seamlessly into standard CEM pipelines with negligible overhead. Empirical results on synthetic benchmarks, a cluttered navigation task, full MBRL pipelines, and a real-world quadruped robot demonstrate that BC-EvoCEM enhances both convergence and solution quality, providing a simple yet effective upgrade for CEM.

1 Introduction

The *Cross–Entropy Method* (CEM) is a derivative–free stochastic optimizer that converts an optimization problem into a sequence of rare event estimation tasks (Rubinstein & Kroese, 2004; De Boer et al., 2005). At each iteration, CEM samples N candidates $\{x_j\}_{j=1}^N$ from a parametric distribution p_{θ_t}, selects top lowest-cost samples as an elite set $\mathcal{E}_t$, and updates the parameters by maximizing the log-likelihood of these elites:

$$\theta_{t+1} \;=\; \arg\max_{\theta} \sum_{x \in \mathcal{E}_t} \log p_\theta(x), \tag{1}$$

optionally smoothed via exponential averaging for stability. Its reliance solely on cost-based ranking instead of gradient information has made CEM a widely adopted solver for high-dimensional, nonconvex optimization tasks in robotics and control (Pinneri et al., 2021; Kobilarov, 2012; Banks et al., 2020).

In model–based reinforcement learning (MBRL), an agent learns a predictive model of the environment and plans through that model to reduce costly real-world interactions (Ha & Schmidhuber, 2018; Nagabandi et al., 2018; Silver et al., 2017). Stochastic model predictive control (MPC) is a widely used planning strategy in this setting (Williams et al., 2016; Okada & Taniguchi, 2020; Chua et al., 2018; Zhang et al., 2022; Deisenroth & Rasmussen, 2011). At every decision step, MPC solves a finite–horizon trajectory optimization problem, executes only the first action, observes the next state, and replans. The CEM is often chosen as the optimizer within this loop due to its simplicity, reliance solely on cost function evaluations, and robustness to noisy or nonconvex objectives.

Despite these advantages, vanilla CEM suffers from its inherent *mode–seeking* nature: as the elites concentrate, it often collapses the search into a local optimum, which significantly limits the exploration in complex multimodal landscapes typical of RL tasks. Ensemble strategies have been proposed to mitigate this issue by running multiple CEM workers. **Centralized ensembles** merge the elite sets of all workers and fit an explicit mixture model (e.g., commonly a Gaussian mixture (Okada & Taniguchi, 2020)). Although more expressive, they introduce additional hyperparameters (number of components, importance weights) and increase computational cost due to joint expectation maximization (EM) steps. **Decentralized ensembles**

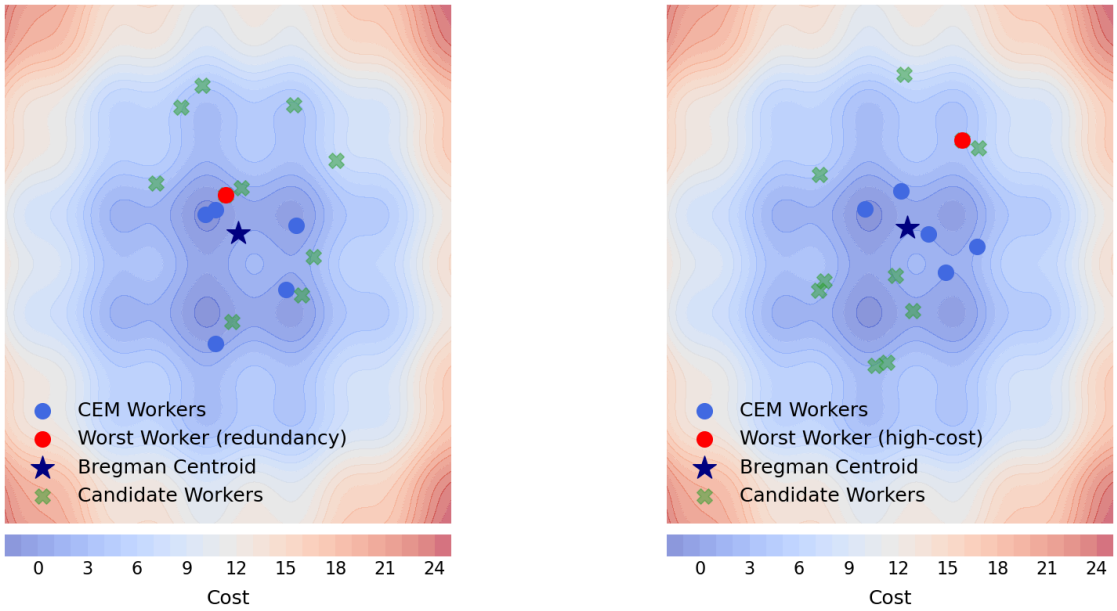


Figure 1: Illustration of $\mathcal{BC}$-EvoCEM (only *means* are shown). • Active CEM workers. • Worst workers identified due to redundancy(left) and poor quality(right). ★ Bregman Centroid as a geometric average of active workers. × Potential Candidates sampled from the trust-region.

run multiple CEM instances in parallel, keep them independent, and output the best solution at termination (Zhang et al., 2022). This approach is simple and scalable but tends to duplicate exploration effort and may reach premature consensus if poorly initialized.

Our Approach. Motivated by the trade-off between diversity preservation and computational efficiency, we introduce **$\mathcal{B}$regman-$\mathcal{C}$entroid Guided CEM ($\mathcal{BC}$-EvoCEM)**, a hybrid strategy that retains the independent updates of decentralized ensembles yet introduces a simple information–geometric coupling across workers. At each CEM iteration, **$\mathcal{BC}$-EvoCEM** computes a *performance–weighted Bregman centroid* (Nielsen & Nock, 2009) of all workers' distributions. The centroid then defines both a reference point and a *Bregman ball* trust region. Any worker whose distribution lies too close to the centroid or exhibits high cost is respawned by drawing new parameters from this trust region (see Fig. 1).

Contributions. 1) We formulate an information–geometric aggregation rule based on Bregman centroids that summarizes ensemble CEM workers with *negligible* computation cost. 2) We provide a lightweight integration into the MPC loop for MBRL, preserving the benefits of **$\mathcal{BC}$-EvoCEM** with the simplicity of a standard warm-start heuristic. 3) Through experiments on multimodal synthetic functions, cluttered navigation tasks, and full MBRL benchmarks, we demonstrate faster convergence with improved performance relative to the vanilla and decentralized CEM.

2 Bregman Divergence

We review key definitions and properties of Bregman divergences (Bregman, 1967) used throughout this paper. Let $F : \mathcal{S} \to \mathbb{R}$ be a strictly convex, differentiable potential function on a convex set $\mathcal{S}$. The *Bregman divergence* between any two points $x, y \in \mathcal{S}$ is defined as

$$\mathrm{D}_F(x\|y) \;=\; F(x) \;-\; F(y) \;-\; \big\langle x - y,\; \nabla F(y)\big\rangle, \tag{2}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product. Although D_F is not a *metric*, it retains "distance-like" and statistical properties useful in optimization and machine learning applications (Snell et al., 2017; Ahn & Chewi, 2021). In particular, its bijective correspondence with exponential families provides a range of clustering and mixture modeling techniques (Banerjee et al., 2005).

Bregman Centroid & Information Radius. Given a collection of points $\{x_i\}_{i=1}^n \subset \mathcal{S}$, the *Bregman centroid* (right-sided) is the solution to the following minimization problem (Nielsen & Nock, 2009):

$$\boldsymbol{x_c} \;=\; \arg\min_{x\in\mathcal{S}} \frac{1}{n}\sum_{i=1}^{n} \mathrm{D}_F(x_i\|x).$$

The corresponding minimized value is known as the *Information Radius* (IR) (Csiszár et al., 2004) (*Bregman Information* in (Banerjee et al., 2005)), which characterizes the *diversity* of the set $\{x_i\}$ under the geometry induced by F. Notably, for $F = \|x\|^2$, the IR coincides with the sample variance of the set. See (Banerjee et al., 2005; Nielsen & Nock, 2009; Csiszár et al., 2004) for more details.

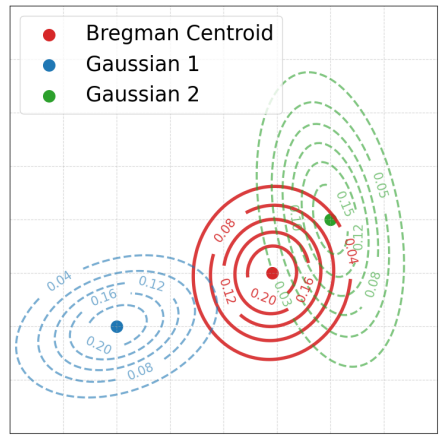


Figure 2: Illustration of the Bregman centroid of two Gaussians.

3 Method

We propose a statistical characterization of a set of distributions through a **weighted Bregman centroid** (Nielsen & Nock, 2009). Let a set of CEM distributions $\{\theta_1, \ldots, \theta_n\}$be drawn from *a parametric family* $\{p_\theta\}_{\theta\in\Theta}$, each associated with an importance weight w_i that reflects its solution quality. Formally, the weighted Bregman centroid $\boldsymbol{\theta}_c$ of the set is defined as

$$\boldsymbol{\theta}_c \;=\; \arg\min_{\theta\in\Theta} \sum_{i=1}^{n} w_i\, \mathrm{D}_F\big(\theta_i \,\big\|\, \theta\big), \tag{3}$$

where D_F is a Bregman divergence associated with a potential F. During the CEM iterations, the weights are assigned based on the performance of p_{θ_i} (e.g., $w_i \propto \exp(-\mathbb{E}_{p_{\theta_i}}[J(x)])$). Hence, the centroid serves as a *performance-weighted "geometric average"* of all CEM workers. To ensure that the ensemble remains effective in terms of both **performance and diversity**, we introduce two essential definitions:

Definition 1 (Relevance Score)**.** *Let the* score *of a worker θ_i be the weighted Bregman divergence to the centroid:*

$$\boxed{\gamma_i \;=\; \mathrm{D}_F\big(\theta_i \,\big\|\, \boldsymbol{\theta}_c\big).}$$

Definition 2 (Trust Region)**.** *For $\Delta > 0$, the trust region is defined as a* ***Bregman ball*** *centering at $\boldsymbol{\theta}_c$ with radius Δ:*

$$\boxed{\mathcal{B}_\Delta(\boldsymbol{\theta}_c) \;=\; \Big\{\theta\in\Theta \;\Big|\; \mathrm{D}_F\big(\theta \,\big\|\, \boldsymbol{\theta}_c\big) \;\le\; \Delta\}\Big\}.}$$

Interpretation of Relevance Scores. Intuitively, a low relevance score γ_i indicates that either the worker's performance weight w_i is low or it's close to the centroid. Such workers contribute minimally to both exploitation and exploration and are therefore candidates for replacement. Moreover, one can verify that γ_i is exactly θ_i's contribution to the Information Radius (IR) of the set under the probabilistic vector $\mathbf{w}$.

Role of the Trust Region. The trust region $\mathcal{B}_\Delta(\boldsymbol{\theta}_c)$ constrains where new workers may be introduced, ensuring that they remain in average proximity to the active workers. Crucially, defining the trust region as a *Bregman ball* aligns with the intrinsic geometry of the chosen parametric family, which offers theoretical insights and computational advantages when employing exponential families, as further discussed in Section 4.

Bregman Centroid Guided Evolution Strategy. Building on the above components, we propose a simple *Bregman Centroid Guided Evolution Strategy* for ensemble CEM (see Alg. 1). At each iteration, it begins with distributed CEM updates, continues with score evaluation, and finishes with an evolutionary update that replaces the lowest-scoring worker with a candidate sampled from the trust region.

Algorithm 1 $\mathcal{BC}$-evoCEM: Guided Evolution Strategy For CEM

Require: CEM distributions $\{\theta_i\}_{i=1}^n$, cost function $J(\,\cdot\,)$, iterations T

for $t = 1$ **to** T **do**
  CEM update: $\{\theta_i\} \leftarrow$ Distributed CEM$(\{\theta_i\}, J)$
  Performance weights: $w_i \propto \exp(-\mathbb{E}_{p_{\theta_i}}[J\,])$
  Centroid: $\boldsymbol{\theta}_c \leftarrow \arg\min_\theta \sum_i w_i \mathrm{D}_F(\theta_i \| \theta)$
  Scores: $\gamma_i \leftarrow \mathrm{D}_F(\theta_i \| \boldsymbol{\theta}_c)$
  Replace: $\theta_{\min} \leftarrow \arg\min_i \gamma_i$; $\theta_{\min} \leftarrow$ Sample$\big(\mathcal{B}_\Delta(\boldsymbol{\theta}_c)\big)$
end for
Return centroid $\boldsymbol{\theta}_c$ and optimized workers $\{\theta_i\}_{i=1}^n$

4 Stochastic Optimization in Exponential Families

In this section, we leverage the relationship between regular exponential families and Bregman divergences to gain statistical insight into our guided evolution strategy (Alg. 1) and achieve substantial computational savings in the CEM pipeline.

Let $\{p_\theta\}_{\theta\in\Theta}$ be a *regular, minimal* exponential family in *natural* form

$$p_\theta(x) \;\propto\; \exp\big\{\theta^\top T(x) - \Psi(\theta)\big\}, \quad \theta \in \Theta \subset \mathbb{R}^d,$$

where $T(x) \in \mathbb{R}^d$ represents the sufficient statistics and Ψ is the strictly convex cumulant. The corresponding Bregman divergence is $\mathrm{D}_\Psi(\theta\|\theta')$ (Banerjee et al., 2005). We denote the *mean parameter* by $\eta = \nabla\Psi(\theta) = \mathbb{E}_{p_\theta}[T(X)]$. Since the map $\nabla\Psi : \Theta \to \mathcal{E} = \operatorname{int}\nabla\Psi(\Theta)$ is a bijection between the natural space Θ and the mean space $\mathcal{E}$ (see (Barndorff-Nielsen, 2014; Amari, 1995)), we advocate representing and manipulating distributions in the mean space.

As the core of our method, the Bregman centroid admits a simple form in mean coordinates:

Proposition 1 (Centroid in mean coordinates)**.** *Given weights* $\mathbf{w} = (w_1, \ldots, w_n)$*,* $w_i \geq 0$*,* $\sum_i w_i = 1$*, and corresponding mean parameters* η_i*, the weighted Bregman centroid satisfies*

$$\boldsymbol{\eta}_c \;=\; \sum_{i=1}^n w_i\,\eta_i, \quad \boldsymbol{\theta}_c \;=\; (\nabla\Psi)^{-1}(\boldsymbol{\eta}_c).$$

Proof. By the *mean-as-minimizer* property of right-sided Bregman divergences (Banerjee et al., 2005; Nielsen & Nock, 2009), the optimality condition $\nabla\Psi(\boldsymbol{\theta}_c) = \boldsymbol{\eta}_c$ uniquely determines $\boldsymbol{\theta}_c$ via the bijection $\nabla\Psi\colon \Theta \to \mathcal{E}$. □

Given that CEM's likelihood evaluations (see Eq. equation 1) already yield the empirical mean $\widehat{\eta}_i = \frac{1}{N}\sum_{j=1}^N T_i(x)$, the centroid $\boldsymbol{\eta}_c$ is obtained **for free**. No extra optimization (e.g., solving equation 3) is required.

4.1 Scoring as Likelihood-based Ranking

Since only *relative* scores matter for ranking workers in Alg. 1, we may drop all terms independent of i and rewrite the *relevance score* as (see Appendix A)

$$\gamma_i = \mathrm{D}_\Psi\big(\theta_i \parallel \boldsymbol{\theta}_c\big) \;\propto\; \Big[\Psi(\theta_i) - \langle\theta_i, \boldsymbol{\eta}_c\rangle\Big] \;=\; -\,\ell(\theta_i; \boldsymbol{\eta}_c),$$

where $\ell(\theta; x) = \langle \theta, x \rangle - \Psi(\theta)$ is precisely the *per-sample* log-likelihood that the natural parameter θ_i would attain under some (hypothetical) dataset whose empirical average is $\boldsymbol{\eta}_c$. Intuitively, ranking workers by $\ell(\theta_i; \boldsymbol{\eta}_c)$ is equivalent to asking:

How well the worker θ_i explain the aggregated information collected from all workers $\{\theta_i\}_{i=1}^n$?

This yields a cheap moment-matching ranking metric that requires only inner products and evaluations of Ψ.

4.2 Efficient Trust-Region Sampling

Given the centroid $\boldsymbol{\eta}_c$, we sample candidates from the trust region $\mathcal{B}_\Delta(\boldsymbol{\theta}_c)$ by working with its dual characterization $\mathcal{S} := \{\eta \in \mathcal{E} : \mathrm{D}_{\Psi^*}(\boldsymbol{\eta}_c \parallel \eta) \leq \Delta\}$, where Ψ^* is the convex conjugate of Ψ.

Definition 3 (Radial Bregman Divergence)**.** *For $v \in \mathbb{S}^{d-1}$ (the unit sphere) define the radial Bregman divergence with respect to a fixed $\eta \in \mathcal{E}$*

$$g_v(\rho) := \mathrm{D}_{\Psi^*}\big(\eta \parallel \eta + \rho v\big), \qquad \rho \geq 0.$$

Theorem 1. *The* Turst-Region Sampler *(see Alg. 2) produces $\eta_{\text{new}} \sim \mathrm{Unif}(\mathcal{S})$ and $\theta_{\text{new}} \in \mathcal{B}_\Delta(\boldsymbol{\theta}_c)$. If Ψ is quadratic (e.g., fixed-Σ Gaussian), θ_{new} is uniformly distributed in $\mathcal{B}_\Delta(\boldsymbol{\theta}_c)$.*

Proof. See Appendix C. □

Algorithm 2 Trust-Region Sampler

Require: centroid $\boldsymbol{\eta}_c$, radius Δ
Require: radial divergence $g_v(\rho)$
1: Draw $v \sim \mathrm{Unif}(\mathbb{S}^{d-1})$
2: Root-solve $g_v(\rho_{\max}) = \Delta$
3: Sample $u \sim \mathrm{Unif}[0, 1]$
4: **Return** $\eta_{\text{new}} \leftarrow \boldsymbol{\eta}_c + u^{1/d}\, \rho_{\max}(v)\, v$

Algorithm 3 Proxy Sampler

Require: centroid $\boldsymbol{\eta}_c$, radius Δ
Require: Hessian $\mathrm{H} = \nabla^2 \Psi^*(\boldsymbol{\eta}_c)$
1: Draw $v \sim \mathrm{Unif}(\mathbb{S}^{d-1})$
2: Set $\widehat{\rho}_{\max}(v) \leftarrow \sqrt{2\Delta/(v^\top \mathrm{H} v)}$
3: Sample $t \sim \mathrm{Unif}[-\widehat{\rho}_{\max}, \widehat{\rho}_{\max}]$
4: **Return** $\eta_{\text{new}} \leftarrow \boldsymbol{\eta}_c + t\, v$

Local Proxy Sampling & Gaussian Case. While Alg. 2 is general and exact, every draw incurs a root–solving $g_v(\rho_{\max}) = \Delta$, which is expensive for high-dimensional parameterization (e.g., action sequences in MBRL). In practice, we *locally* approximate $g_v(\rho) \approx \frac{1}{2}\rho^2 v^\top \mathrm{H} v$ at the centroid $\boldsymbol{\eta}_c$ up to second order, where $\mathrm{H} = \nabla^2 \Psi^*(\boldsymbol{\eta}_c)$. This yields an *ellipsoidal* trust region $\widehat{\mathcal{S}}$ with a **closed-form** maximal radius

$$\widehat{\mathcal{S}} \;=\; \big\{\eta :\; (\eta - \boldsymbol{\eta}_c)^\top \mathrm{H}(\eta - \boldsymbol{\eta}_c) \leq 2\Delta\big\} \;\implies\; \widehat{\rho}_{\max}(v) \;=\; \sqrt{2\Delta/(v^\top \mathrm{H} v)},$$

which requires only one dot product and a square root (See the Proxy Sampler in Alg. 3).

For diagonal Gaussian action-sequence planner (common in MBRL (Chua et al., 2018; Zhang et al., 2022; Deisenroth & Rasmussen, 2011)), the Hessian $\mathrm{H} = \mathrm{diag}(h_1, \ldots, h_d)$ itself is diagonal and the resulting trust region becomes axis-aligned with principal radii $\sqrt{2\Delta/h_i}$. In such cases, sampling further reduces to simple coordinate-wise operations (see Appendix B for details).

Table 1: Major operations in the CEM loop. Here n is the number of workers, d the parameter dimension, and m the (typically small) number of iterations in the root solver of Alg. 2. All cost are worst-case. Quantities marked † are already computed in CEM.

Operation	**Cost**	**Comment**
Centroid	$O(nd)$	Weighted average of $\eta_i^\dagger$
Relevance score (γ_i)	$O(d)$ per worker	One inner product + $\Psi(\theta_i)$ (closed form)
Exact sampler ($d \lesssim 100$)	$O(d + m)$	Root solve for $\rho_{\max}$ (e.g., secant method)
Proxy sampler ($d \gg 100$)	$O(d)$	Closed-form $\widehat{\rho}_{\max}$

Summary. By operating with the mean parameterization of the exponential family, $\mathcal{BC}$**-EvoCEM**'s add-on operations incur negligible overhead. The empirical means required for the Bregman centroid are available from the CEM log–likelihood computation. All subsequent steps (see Table 1) scale linearly with the parameter dimension and remain trivial compared to environment roll–outs. Moreover, the geometric interpretation of our method is remarkably intuitive and *Euclidean-like*: the Bregman centroid coincides with a weighted arithmetic mean, and the proxy trust region resembles an ellipsoidal neighbourhood under a natural affine transformation.

5 Bregman Centroid Guided MPC for MBRL

The proposed $\mathcal{BC}$**-EvoCEM** integrates elegantly into the MPC pipeline for MBRL, where the trajectory optimization is performed iteratively in a receding horizon fashion. Instead of warm starting CEM optimizer at time $t+1$ by shifting the previous solution (Chua et al., 2018) or restarting from scratch, we use the performance-weighted Bregman centroid of the K independent CEM solutions to initialize the next iteration. To prevent ensemble collapse (i.e., IR $\to 0$), we periodically replace the least-contributing workers by candidates sampled from the trust region.

Algorithm 4 (schematic) Drop-in MPC Wrapper for MBRL.

Require: K CEM workers, buffer $\mathcal{D}$

```
for each training iteration do
    Train dynamics model f̃ on D
    Initialize Bregman Centroid
    for each control step t = 1, ..., H do
        Warm start CEM workers by BC
        Rollouts by f̃ & Update CEM workers
        Compute Bregman Centroid
        (periodic) Score & Replace
        Execute the best worker's 1st action
        Add transitions to D
    end for
end for
```

Building on the stochastic optimization techniques in Sec. 4, we implement this strategy as a *drop-in MPC wrapper for MBRL* (see Alg. 4) that

1. preserves the internal CEM update unchanged,
2. enforces performance–diversity control via the Bregman centroid,
3. and incurs only a few additional vector operations per control step.

Here, the Bregman centroid encapsulates the CEM ensemble's consensus on promising action-sequences while implicitly encoding optimality-related uncertainties in the warm start. The trust-region based replacement then reinjects diversity in regions where the model is confident. Therefore, this implementation delivers the benefits of guided evolution with the simplicity of a standard warm-start heuristic.

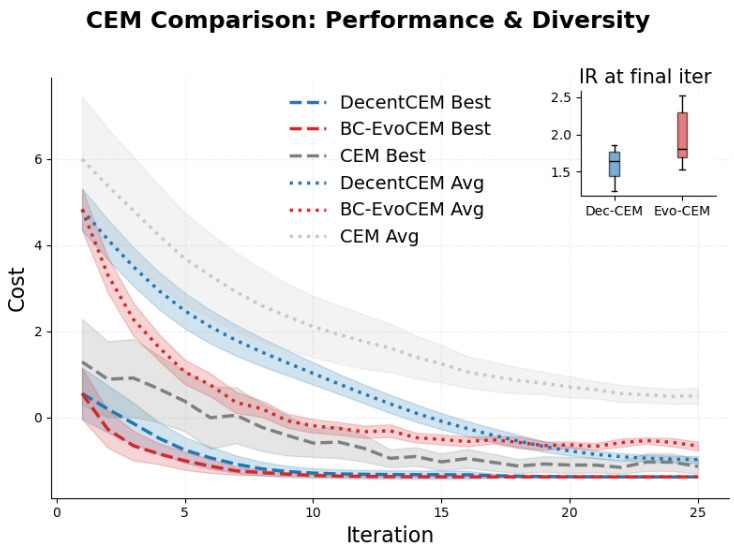


Figure 3: Performance comparison for vanilla, decentralized, and our CEM methods. Solid/dashed lines show the mean/best cost, shaded bands ±1 std. Information radius (IR) at iter 25 is shown.

6 Experimental Results

6.1 Multi-modal optimization

We first demonstrate our method on a multi-modal optimization problem with the cost function (Fig. 1 shows the cost landscape with multiple attraction basins):

$$J(\boldsymbol{x}) \;=\; \sin(3x_1) + \cos(3x_2) + 0.5\,\|\boldsymbol{x}\|_2^2.$$

We compare our method against (1) vanilla CEM and (2) decentralized CEM(DecentCEM) (Zhang et al., 2022), with the same parametric distribution $p_\theta = \mathcal{N}(\theta, 0.5^2 I)$. Our approach (red in Fig. 3) demonstrates faster convergence in both *Best* and *Average* costs. Importantly, the trust-region sampling maintains solution diversity, as shown by the final IR values (i.e., sample variance in this case).

Better worker efficiency. Our scaling study (see Fig.4) reveals a clear $\log_2(\#\text{WORKERS})$ law: beyond 8–16 workers, DecentCEM shows sharp diminishing returns. BC-EvoCEM achieves the same best-cost and diversity asymptotics with roughly *half* of the workers. It shows markedly higher ***worker efficiency*** while avoiding the quadratic rollout cost that comes with an increasingly larger ensemble.

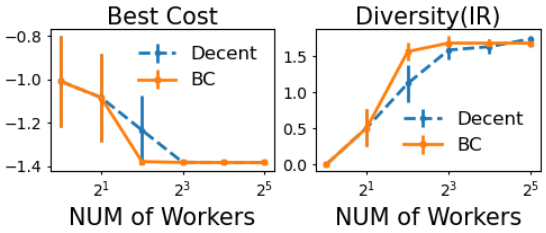


Figure 4: Scaling with workers (20 SEEDS). BC-EvoCEM needs fewer workers to hit the same best-cost.

6.2 Navigation Task

We consider a cluttered 2D navigation task with first-order dynamics and time-step $\Delta t = 0.2$. A planning horizon of $H = 200$ yields a $2H$-dimensional action sequence. We employ 5 independent diagonal-Gaussian CEM workers with identical CEM hyperparameters and initialization. To sample from the trust region in this high-dimensional space, we use the PROXYSAMPLER (Alg. 3).

Figure 5 (left) visualizes trajectories from a fully decentralized CEM, which disperse widely and frequently deviate from the start–goal line. In contrast, BC-EvoCEM (right) maintains a tight cluster of trajectories

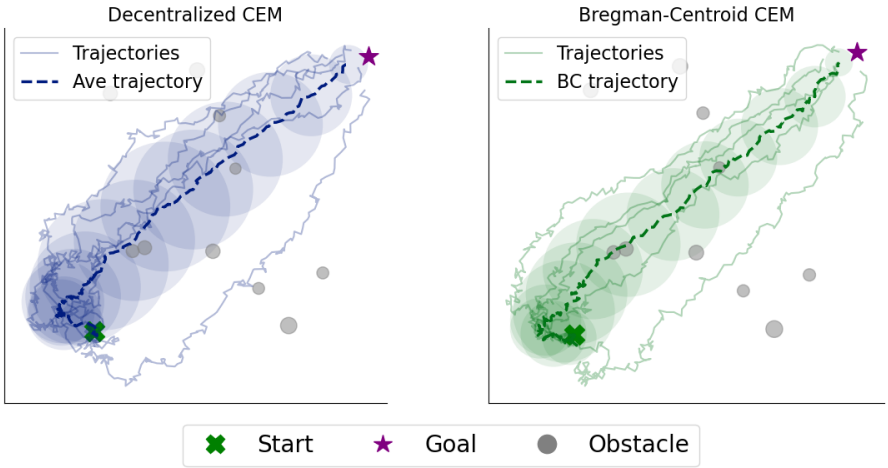


Figure 5: Trajectory distributions from decentralized CEM (left) and Bregman–centroid guided CEM (right) on a point-mass navigation task. The *representatives* of each method (dashed line) are the average and Bregman-centroid trajectory.

around the Bregman-centroid path (green dashed line), producing a more diverse and goal-directed planning. Notably, the centroid itself is not guaranteed to avoid obstacles as it serves only as an information-geometric summary of all workers. Quantitatively, BC-EvoCEM yields significant improvements in both average and best cost without incurring noticeable computational overhead.

Table 2: Normalized costs and relative drop versus decentralized CEM.

	Average cost		Best cost	
Method	Norm.	Drop (%)	Norm.	Drop (%)
Decentralized CEM	1.00	—	1.00	—
BC-evoCEM	0.18	82.4	0.55	45.3

6.3 Bregman Centroid Guided MPC in MBRL

Baselines and minimal implementation. Our study is built on the PETS framework (Chua et al., 2018) and the DecentCEM implementation of Zhang et al. (2022). Dynamics learning, experience replay, and per-worker CEM updates are left untouched. We wrap the optimizer with a *Bregman-Centroid warm-start* (see the MPC in 4). The wrapper is drop-in and lightweight, i.e. < 50 lines of Python code. This plug-and-play feature makes the method readily portable to any planning-based MBRL codebase.

Protocol. To isolate the impact of the trajectory optimizer, we keep the rest of PETS fixed and compare three variants: vanilla CEM, DecentCEM, and BC-evoCEM under two dynamics models: 1) a **deterministic** model trained by minimizing MSE, and 2) a **probabilistic ensemble** with trajectory sampling (Chua et al., 2018). Full results and hyper-parameters are in Appendix D.

Experimental results. With no additional stochasticity in the pipeline, all performance differences stem from the optimizer itself. BC-evoCEM delivers **faster early learning** and **higher asymptotic return** on four of six benchmarks (see Figure 6). Its sample-efficiency advantage is quantified in Table 3, where BC-evoCEM reaches 90% of its own final reward up to **4**× sooner than baselines. Vanilla CEM's covariance collapses quickly, and DecentCEM suffers from independent worker collapse. By contrast, the Bregman centroid keeps workers in promising regions *while preserving ensemble diversity*, translating into sustained exploration and higher returns.

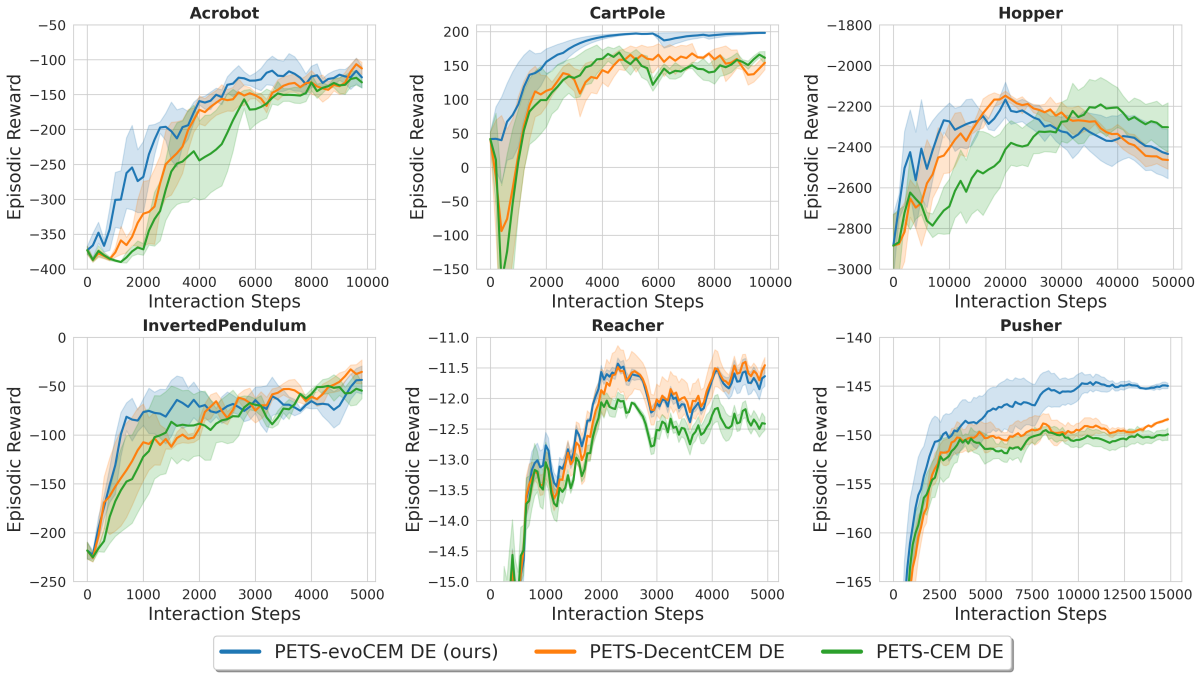


Figure 6: Training return curves across six control tasks using PETS with different CEM-based optimizers. All methods use the deterministic dynamics model. Curves show mean performance over 3 random seeds.

Table 3: Sample–efficiency *boost* (↑) of BC-EVOCEM over two CEM variants. Values > 1 mean BC-evoCEM learns faster

Environment	vs. DecentCEM	vs. vanilla CEM
Acrobot	×2.1	×2.6
CartPole	×3.4	×4.0
Hopper	×1.0	×1.1
Inverted Pendulum	×1.1	×1.2
Reacher	×1.0	×1.1
Pusher	×1.9	×2.0

When both *epistemic* and *aleatoric* uncertainty are already captured via bootstrap sampling, the performance differences among the three optimizers become statistically indistinguishable (see Appendix D.3). Here, the model's intrinsic stochasticity supplies sufficient trajectory dispersion, so further optimizer-level exploration yields diminishing returns.

Uncertainties in MBRL. The controlled study highlights two distinct yet coupled sources of uncertainty in model-based RL: *model uncertainty* and *optimality uncertainty*. Improving the dynamics model (e.g., probabilistic ensembles) addresses the former, whereas a diversity-informed optimizer (e.g., BC-EvoCEM) directly addresses the latter. Once the dynamics model approaches its performance cap (or its representational capacity is bottlenecked), optimality uncertainty predominates; in this regime, geometry-informed exploration such as BC-EvoCEM in the action space delivers a complementary boost.

6.4 Real-World Quadruped Navigation Experiments

To demonstrate that BC-evoCEM extends beyond simulation, we deploy it on a UNITREE GO2 quadruped tasked with navigating a 4×4 m cluttered arena (Fig. 7a). Both BC-evoCEM and a vanilla CEM baseline optimize a velocity policy control $= [v_{\text{linear}}, v_{\text{angular}}]$ at 20 Hz. Success is defined as reaching the goal without collision within a given number of iterations. We report five metrics in Table 4.

Table 4: Performance metrics (lower ↓ is better).

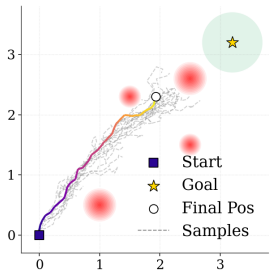
	Success	Eff. ↓	Smooth ↓	Cost ↓	t_{opt} ↓
BC-evoCEM	✓	1.158	**10.90**	**21.16**	0.462
CEM	✗	**1.079**	12.92	36.96	**0.400**

Experimental results. Figure 7 and Table 4 demonstrate that:

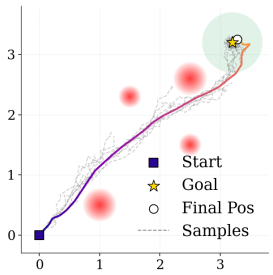
- **Planned Path** (Fig. 7). The vanilla CEM rollout drifts into the obstacle cluster and halts ∼0.9 m short of the goal (white circle), whereas BC-evoCEM generates a collision-free path that reaches the goal (yellow star). Its sample cloud (gray) remains tighter and farther from danger zones.
- **Quantitative metrics** (Table 4). Compared to vanilla CEM, BC-evoCEM yields a **smoother** control profile and cuts cumulative cost by **43%**, with only a modest +0.062 s optimization overhead.

These results validate our central claim: BC-evoCEM is a lightweight, drop-in upgrade to existing CEM implementations that delivers substantially safer, smoother, and cheaper trajectories without sacrificing real-time performance.

(a) UNITREE GO2 executing BC-evoCEM's plans.



(b) Planned path — CEM.



(c) Planned path — BC-evoCEM.

Figure 7: Real-world navigation experiment. Top: sequential onboard snapshots. Bottom: top-down planned paths from each optimizer. Quantitative metrics are reported separately in Table 4.

7 Conclusion

We introduced $\mathcal{BC}$**-EvoCEM**, a lightweight ensemble extension of the Cross-Entropy Method that has (i) principled information aggregation and (ii) diversity-driven exploration with near-zero computation overhead. Across optimization problems and model-based RL benchmarks, $\mathcal{BC}$**-EvoCEM** demonstrates faster convergence and attains higher-quality solutions than vanilla and decentralized CEM. Its plug-and-play design enables easy integration into MPC loops while preserving the algorithmic simplicity that makes CEM appealing in the first place.

Limitations In this subsection, we outline several limitations of the proposed $\mathcal{BC}$**-EvoCEM** and provide potential directions for addressing them in future work.

All information-geometric arguments (closed-form centroid, ellipsoidal trust region, likelihood-based ranking) hold only for *regular exponential-family* distributions in mean coordinates. This restriction limits the expressiveness of the CEM distributions. Future work will transfer these ideas to richer models via *geometric-preserving* transport maps (Villani et al., 2008). In addition, while we prove the centroid and repawned CEM workers remain inside a Bregman ball, the method still lacks global optimality guarantees and convergence analysis. It inherits these limitations from CEM. A promising direction is to consider the proposed $\mathcal{BC}$**-EvoCEM** in the stochastic mirror-descent framework (Ahn & Chewi, 2021), which may provide non-asymptotic convergence bounds via the primal-dual relationship.

Acknowledgments

This work was supported in part by NASA ULI (80NSSC22M0070), Air Force Office of Scientific Research (FA9550-21-1-0411), NSF CPS (2311085), NSF RI (2133656), NSF CMMI (2135925), NASA under the Cooperative Agreement 80NSSC20M0229, and NSF SLES (2331878). Marco Caccamo was supported by an Alexander von Humboldt Professorship endowed by the German Federal Ministry of Education and Research.

References

Kwangjun Ahn and Sinho Chewi. Efficient constrained sampling via the mirror-langevin algorithm. *Advances in Neural Information Processing Systems*, 34:28405–28418, 2021.

Shun-ichi Amari. Information geometry of the em and em algorithms for neural networks. *Neural networks*, 8(9):1379–1408, 1995.

Arindam Banerjee, Srujana Merugu, Inderjit S Dhillon, and Joydeep Ghosh. Clustering with bregman divergences. *Journal of machine learning research*, 6(Oct):1705–1749, 2005.

Christopher Banks, Sean Wilson, Samuel Coogan, and Magnus Egerstedt. Multi-agent task allocation using cross-entropy temporal logic optimization. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7712–7718. IEEE, 2020.

Ole Barndorff-Nielsen. *Information and exponential families: in statistical theory.* John Wiley & Sons, 2014.

Lev M Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR computational mathematics and mathematical physics*, 7(3):200–217, 1967.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018.

Imre Csiszár, Paul C Shields, et al. Information theory and statistics: A tutorial. *Foundations and Trends® in Communications and Information Theory*, 1(4):417–528, 2004.

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

Marc Deisenroth and Carl E Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pp. 465–472, 2011.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

Marin Kobilarov. Cross-entropy motion planning. *The International Journal of Robotics Research*, 31(7): 855–871, 2012.

Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 7559–7566. IEEE, 2018.

Frank Nielsen and Richard Nock. Sided and symmetrized bregman centroids. *IEEE transactions on Information Theory*, 55(6):2882–2904, 2009.

Masashi Okada and Tadahiro Taniguchi. Variational inference mpc for bayesian model-based reinforcement learning. In *Conference on robot learning*, pp. 258–272. PMLR, 2020.

Cristina Pinneri, Shambhuraj Sawant, Sebastian Blaes, Jan Achterhold, Joerg Stueckler, Michal Rolinek, and Georg Martius. Sample-efficient cross-entropy method for real-time planning. In *Conference on Robot Learning*, pp. 1049–1065. PMLR, 2021.

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, 2004.

Rolf Schneider. *Convex bodies: the Brunn–Minkowski theory*, volume 151. Cambridge university press, 2013.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.

Cédric Villani et al. *Optimal transport: old and new*, volume 338. Springer, 2008.

Tingwu Wang, Xuchan Bao, Ignasi Clavera, Jerrick Hoang, Yeming Wen, Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba. Benchmarking model-based reinforcement learning. *arXiv preprint arXiv:1907.02057*, 2019.

Grady Williams, Paul Drews, Brian Goldfain, James M Rehg, and Evangelos A Theodorou. Aggressive driving with model predictive path integral control. In *2016 IEEE international conference on robotics and automation (ICRA)*, pp. 1433–1440. IEEE, 2016.

Zichen Zhang, Jun Jin, Martin Jagersand, Jun Luo, and Dale Schuurmans. A simple decentralized cross-entropy method. *Advances in Neural Information Processing Systems*, 35:36495–36506, 2022.

A Relevance Score as Likelihood Evaluation

Recall that the Bregman divergence induced by Ψ is

$$\mathrm{D}_\Psi(\theta \parallel \boldsymbol{\theta}_c) = \Psi(\theta) - \Psi(\boldsymbol{\theta}_c) - \big\langle \nabla\Psi(\boldsymbol{\theta}_c),\, \theta - \boldsymbol{\theta}_c \big\rangle.$$

Let $\theta = \theta_i$ and denote the dual centroid $\boldsymbol{\eta}_c = \nabla\Psi(\boldsymbol{\theta}_c)$. Expand

$$\begin{aligned}\gamma_i &= \mathrm{D}_\Psi(\theta_i \parallel \boldsymbol{\theta}_c)\\ &= \Psi(\theta_i) - \Psi(\boldsymbol{\theta}_c) - \big\langle \boldsymbol{\eta}_c,\, \theta_i - \boldsymbol{\theta}_c \big\rangle\\ &= \Psi(\theta_i) - \Psi(\boldsymbol{\theta}_c) - \langle \boldsymbol{\eta}_c,\, \theta_i \rangle + \langle \boldsymbol{\eta}_c,\, \boldsymbol{\theta}_c \rangle.\end{aligned}$$

Since $\Psi(\boldsymbol{\theta}_c)$ and $\langle \boldsymbol{\eta}_c, \boldsymbol{\theta}_c \rangle$ are independent of i, they are constant across workers and can be dropped when ranking. Then, we have

$$\gamma_i \;\propto\; \Psi(\theta_i) - \langle \boldsymbol{\eta}_c,\, \theta_i \rangle \;=\; -\Big[\langle \theta_i,\, \boldsymbol{\eta}_c \rangle - \Psi(\theta_i)\Big].$$

Define the per–sample log-likelihood of the exponential family in canonical form by

$$\ell(\theta; x) \;=\; \langle \theta, x \rangle - \Psi(\theta).$$

Therefore,

$$\gamma_i \;\propto\; -\ell\big(\theta_i; \boldsymbol{\eta}_c\big).$$

B Local Proxy Sampling & Gaussian Case

To address the curse of dimensionality in the root solving step in Algorithm 2, we consider a local approximation of the (dual) trust region

$$\mathcal{S} := \{\eta \in \mathcal{E} : \mathrm{D}_{\Psi^*}(\boldsymbol{\eta}_c \parallel \eta) \le \Delta\},$$

where Ψ^* is the convex conjugate of Ψ. By the definition of the *radial Bregman Divergence* (see Def.3), we have $g_v(0) = 0$ and $\nabla_\rho g_v(0) = 0$ at $\boldsymbol{\eta}_c$. A Taylor expansion about $\rho = 0$ gives

$$g_v(\rho) \;=\; \frac{1}{2}\,\rho^2\, v^\top \underbrace{\nabla^2_\eta \mathrm{D}_{\Psi^*}\big(\boldsymbol{\eta}_c \| \eta\big)\big|_{\eta=\boldsymbol{\eta}_c}}_{=:\,\mathrm{H}}\, v \;+\; \mathcal{O}(\rho^3) \;\approx\; \frac{1}{2}\,\rho^2\, v^\top \mathrm{H}\, v, \tag{4}$$

where

$$\mathrm{H} = \nabla^2_\eta\, \mathrm{D}_{\Psi^*}(\boldsymbol{\eta}_c \| \eta)\big|_{\eta=\boldsymbol{\eta}_c} = \nabla^2\Psi^*(\boldsymbol{\eta}_c) = \big[\nabla^2\Psi(\theta_c)\big]^{-1}.$$

Substituting this quadratic approximation equation 4 into the trust region constraint $g_v(\rho) \le \Delta$ yields

$$\frac{1}{2}\,\rho^2\, v^\top \mathrm{H}\, v \;\le\; \Delta \quad\Longrightarrow\quad \rho \;\le\; \widehat{\rho}_{\max}(v) := \sqrt{\frac{2\Delta}{v^\top \mathrm{H}\, v}}.$$

Hence the proxy trust region in mean space is the Mahalanobis ball

$$\widehat{\mathcal{S}} = \big\{\eta :\; (\eta - \boldsymbol{\eta}_c)^\top \mathrm{H}\, (\eta - \boldsymbol{\eta}_c) \le 2\Delta\big\}.$$

Diagonal Gaussian Case. Consider the family $p_\theta(x) = \mathcal{N}\big(\mu, \mathrm{diag}(\sigma^2)\big)$ with natural parameters $\theta_{1i} = \mu_i/\sigma_i^2,\ \theta_{2i} = -\frac{1}{2}\sigma_i^{-2}$. Its cumulant function is given by

$$\Psi(\theta) = \sum_{i=1}^{d}\Big[-\frac{\theta_{1i}^2}{4\theta_{2i}} - \frac{1}{2}\log(-2\theta_{2i}) + \frac{1}{2}\log(2\pi)\Big],$$

and the convex dual in mean coordinates $\eta_i = \mu_i$ (fixing σ_i^2) is simply

$$\Psi^*(\eta) = \frac{1}{2}\sum_{i=1}^{d}\frac{(\eta_i - \mu_i)^2}{\sigma_i^2} + \text{const}.$$

Here, the Hessian is $\mathrm{H} = \nabla^2\Psi^*(\eta) = \mathrm{diag}\big(\sigma_1^{-2}, \ldots, \sigma_d^{-2}\big)$, so the Mahalanobis ball $\widehat{S}$ becomes *axis-aligned*:

$$\Big[\, \boldsymbol{\eta_c}^i - \sqrt{2\Delta\,\sigma_i^2},\ \boldsymbol{\eta_c}^i + \sqrt{2\Delta\,\sigma_i^2}\,\Big], \quad i = 1, \ldots, d.$$

Hence, sampling reduces to independent coordinate draws:

$$\eta_i \ \sim\ \mathrm{Unif}\Big[\, \boldsymbol{\eta_c}^i - \sqrt{2\Delta\,\sigma_i^2},\ \boldsymbol{\eta_c}^i + \sqrt{2\Delta\,\sigma_i^2}\,\Big], \quad i = 1, \ldots, d.$$

During CEM update, the empirical covariance often collapses, becoming low–rank or even singular; in other words, the *mean* component μ quickly dominates the search directions. A practical trick here is to *freeze* the diagonal variance vector σ^2 after a few iterations (or to enforce a fixed lower bound). In practice, we perform such fix-variance trick during the trust region sampling step for high-dimensional planning tasks, including the MPC implementation for MBRL in Sec. 6.3.

Because the Hessian matrix (Eq. equation 4) is block–diagonal (a diagonal sub-block for the mean and a sub-block for the variances), we can safely use the Proxy Sampler 3 to perform such *coordinate-wise* updates **exclusively** on the mean block. This also avoids numerical issues from near-singular covariances.

C Proof of Theorem 1

C.1 Preliminaries

Throughout we work on $\Theta, \mathcal{E} \subset \mathbb{R}^d$ equipped with Lebesgue measure λ^d. We write σ_{d-1} for the surface measure on the unit sphere $\mathbb{S}^{d-1} := \{v \in \mathbb{R}^d \mid \|v\|_2 = 1\}$. The following facts are used (see Villani et al. (2008); Schneider (2013)).

Fact 1 (Polar coordinates)**.** *Under the polar map $(\rho, v) \mapsto \eta = \eta_c + \rho v$ with $\rho \geq 0,\ v \in \mathbb{S}^{d-1}$, the d-dimensional Lebesgue volume element factorizes as $d\eta = \rho^{d-1}\, d\rho\, d\sigma_{d-1}(v)$.*

Fact 2 (Uniform distribution)**.** *Let $\rho_{\max} : \mathbb{S}^{d-1} \to (0, \infty)$ be measurable and define*

$$\mathcal{S} := \big\{\eta_c + \rho v : v \in \mathbb{S}^{d-1},\ 0 \leq \rho \leq \rho_{\max}(v)\big\}.$$

Then

$$\mathrm{Vol}(\mathcal{S}) = \frac{1}{d}\int_{\mathbb{S}^{d-1}} \rho_{\max}(v)^d\, d\sigma_{d-1}(v)$$

and the uniform law on $\mathcal{S}$ has a radial conditional density

$$f_{\mathcal{S}}(\rho|v) = \frac{d\,\rho^{d-1}}{\rho_{\max}(v)^d}, \qquad 0 \leq \rho \leq \rho_{\max}(v).$$

Fact 3 (Change of variables)**.** *For $\Psi \in C^2(\Theta)$ strictly convex, the gradient map $\nabla\Psi : \Theta \to \mathcal{E}$ is a C^1 diffeomorphism with Jacobian $\det \nabla^2\Psi(\theta)$. For any non-negative φ,*

$$\int_\Theta \varphi(\theta)\, d\theta = \int_{\mathcal{E}} \varphi\big(\nabla\Psi^{-1}(\eta)\big)\Big|\det \nabla^2\Psi\big(\nabla\Psi^{-1}(\eta)\big)\Big|\, d\eta.$$

C.2 Auxiliary lemma

We first show the radial Bregman Divergence (see Def. 3) is strictly increasing.

Lemma 1 (Monotonicity)**.** *Let Ψ^* be strictly convex and twice differentiable. For fixed η_0 and $v \in \mathbb{S}^{d-1}$ define $g_v(\rho) := \mathrm{D}_{\Psi^*}\big(\eta_0 \,\|\, \eta_0 + \rho v\big)$, $\rho \geq 0$. Then g_v is strictly increasing on $(0, \infty)$.*

Proof. Insert $\eta = \eta_0 + \rho v$ into D_{Ψ^*} and differentiate: $g_v'(\rho) = \big\langle \nabla\Psi^*(\eta_0 + \rho v) - \nabla\Psi^*(\eta_0), v\big\rangle$. Strict convexity implies monotonicity of $\nabla\Psi^*$; hence $g_v'(\rho) > 0$ for all $\rho > 0$. □

C.3 Main proof

Theorem 1 (Restatement)**.** *Algorithm 2 produces* $\eta_{\text{new}} \sim \text{Unif}(\mathcal{S})$ *and* $\theta_{\text{new}} \in \mathcal{B}_\Delta(\theta_c)$*. If* Ψ *is quadratic,* θ_{new} *is uniformly distributed in* $\mathcal{B}_\Delta(\theta_c)$*.*

Proof. Let g_v be defined as above.

Step 1. Boundary existence & uniqueness. By Lemma 1, g_v is strictly increasing, so $g_v(\rho) = \Delta$ has a unique root $\rho_{\max}(v) > 0$ for each v.

Step 2. Feasibility. Algorithm 2 draws $V \sim \text{Unif}(\mathbb{S}^{d-1})$ and $U \sim \text{Unif}[0,1]$, sets $\rho = \rho_{\max}(V)\, U^{1/d}$ and $\eta_{\text{new}} = \eta_c + \rho V$. Because $g_V(\rho) \leq g_V(\rho_{\max}(V)) = \Delta$, $\eta_{\text{new}} \in \mathcal{S}$ and hence $\theta_{\text{new}} := \nabla\Psi^{-1}(\eta_{\text{new}}) \in \mathcal{B}_\Delta(\theta_c)$.

Step 3. Uniformity. Conditioned on $V = v$, ρ has density $d\,\rho^{d-1}/\rho_{\max}(v)^d$ on $[0, \rho_{\max}(v)]$, which matches Fact 2; integrating over v therefore yields $\eta_{\text{new}} \sim \text{Unif}(\mathcal{S})$.

Step 4. Pull-back to Θ. By Fact 3, $f_\theta(\theta) = f_{\mathcal{S}}\big(\nabla\Psi(\theta)\big)\,\big|\det \nabla^2\Psi(\theta)\big|$. For general Ψ, $\det \nabla^2\Psi(\theta)$ varies with θ, so f_θ is not constant. If Ψ is quadratic, $\nabla^2\Psi$ is constant; hence f_θ is constant on $\mathcal{B}_\Delta(\theta_c)$, i.e. θ_{new} is uniform. $\square$

D Experimental Details

D.1 Benchmark Environment Setup

We follow the evaluation protocol of Zhang et al. (2022) to assess both our method and the baseline algorithms on the suite of robotic benchmarks introduced by Wang et al. (2019); Chua et al. (2018), including classical robotic control problems and high-dimensional locomotion and manipulation problems. Key environment parameters are summarized in Table 5. We refer the interested readers to Zhang et al. (2022) for more details, such as reward function settings, termination conditions, and other implementation specifics. For each case study, all algorithms are trained on three random seeds and evaluated on one unseen seed.

Table 5: Details of Benchmark Environments

Parameter	**Acrobot**	**CartPole**	**Hopper**	**Pendulum**	**Reacher**	**Pusher**
Train Iterations	50	50	50	50	100	100
Task Horizon	200	200	1000	100	50	150
Train Seeds	$\{1, 2, 3\}$	$\{1, 2, 3\}$	$\{1, 2, 3\}$	$\{1, 2, 3\}$	$\{1, 2, 3\}$	$\{1, 2, 3\}$
Test Seeds	$\{0\}$	$\{0\}$	$\{0\}$	$\{0\}$	$\{0\}$	$\{0\}$
Epochs per Test	3	3	3	3	3	3

D.2 Algorithms Setup

The key parameters for the proposed $\mathcal{BC}$**-EvoCEM** algorithm and all baseline methods are listed in Table 6. The dynamic model for each benchmark is parameterized as a fully connected neural network: four hidden layers with 200 units each, except for the *Pusher* task, which uses three hidden layers. All algorithms share identical training settings for learning the dynamics model; further details on model learning can be found in Chua et al. (2018) and Zhang et al. (2022).

Table 6: Details of Algorithms (**DE** and **PE**)

Parameter	**PETS-evoCEM**	**PETS-DecentCEM**	**PETS-CEM**
CEM Type	$\mathcal{BC}$-EvoCEM	DecentCEM	CEM
CEM Ensemble Size	3	3	1
CEM Population Size	100	100	100
CEM Proportion of Elites	10 %	10 %	10 %
CEM Initial Variance	0.1	0.1	0.1
CEM Internal Iterations	5	5	5
Model Learning Rate	0.001	0.001	0.001
Warm-up Episodes	1	1	1
Planning Horizon	30	30	30

D.3 Stochastic models with trajectory sampling

We adopt the probabilistic ensemble with trajectory sampling of Chua et al. (2018), which captures both *epistemic* and *aleatoric* uncertainty. Under this setting, the learning curves of all three optimizers overlap within statistical error (Fig. 8 and Fig. 9). No significant performance gaps emerge. We conjecture that the model's intrinsic stochasticity already supplies enough trajectory dispersion, so additional exploration at the optimizer level offers little incremental benefit. Moreover, the benchmark continuous-control tasks have largely *unimodal* return landscapes, which leaves less room for diversity-seeking mechanisms to improve performance beyond what stochastic trajectory sampling already provides.

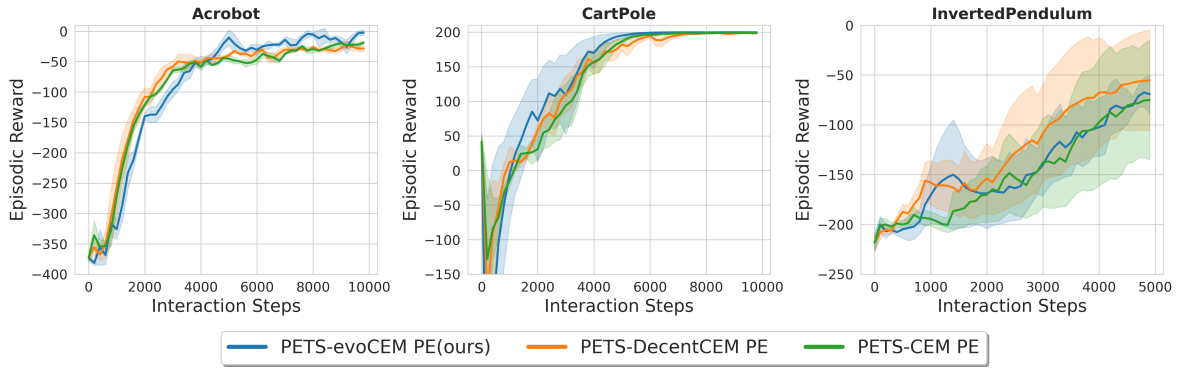


Figure 8: *Training* return curves across 3 control tasks using PETS with different CEM-based optimizers. All methods use the *probabilistic ensemble dynamics model with trajectory sampling* (Chua et al., 2018). Curves show mean performance over 3 random seeds.

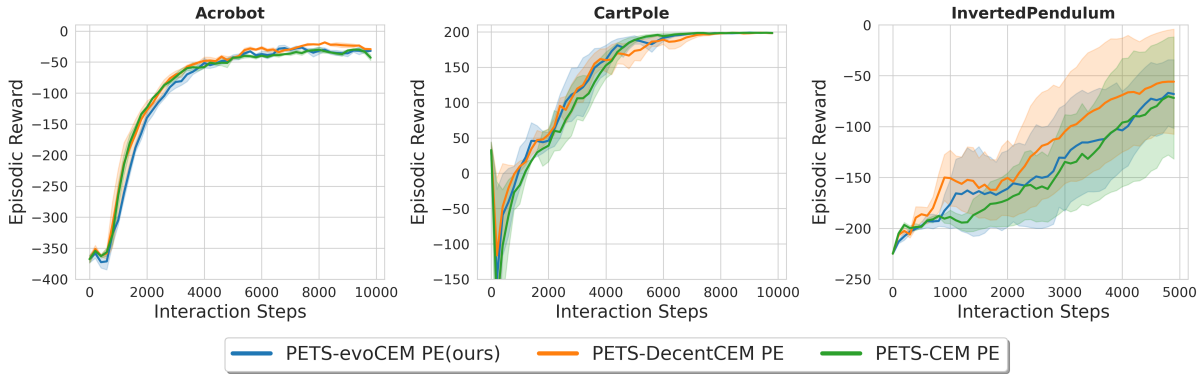


Figure 9: *Testing* return curves across 3 control tasks using PETS with different CEM-based optimizers. All methods use the *probabilistic ensemble dynamics model with trajectory sampling* Chua et al. (2018). Curves show mean performance over 3 random seeds.