Proceedings of Machine Learning Research vol 242:1122–1135, 2024

CoVO-MPC: Theoretical Analysis of Sampling-based MPC and Optimal Covariance Design

Zeji Yi* ZEJIY@ANDREW.CMU.EDU
Chaoyi Pan* CHAOYIP@ANDREW.CMU.EDU
Guanqi He GUANQIHE@ANDREW.CMU.EDU
Guannan Qu† GQU@ANDREW.CMU.EDU
Guanya Shi† GUANYAS@ANDREW.CMU.EDU
Carnegie Mellon University

Abstract

Sampling-based Model Predictive Control (MPC) has been a practical and effective approach in many domains, notably model-based reinforcement learning, thanks to its flexibility and parallelizability. Despite its appealing empirical performance, the theoretical understanding, particularly in terms of convergence analysis and hyperparameter tuning, remains absent. In this paper, we characterize the convergence property of a widely used sampling-based MPC method, Model Predictive Path Integral Control (MPPI). We show that MPPI enjoys at least linear convergence rates when the optimization is quadratic, which covers time-varying LQR systems. We then extend to more general nonlinear systems. Our theoretical analysis directly leads to a novel sampling-based MPC algorithm, CoVariance-Optimal MPC (`CoVO-MPC`) that optimally schedules the sampling covariance to optimize the convergence rate. Empirically, `CoVO-MPC` significantly outperforms standard MPPI by 43-54% in both simulations and real-world quadrotor agile control tasks. Videos and Appendices are available at https://tinyurl.com/covo-mpc-cmu.

Keywords: Sampling-based Model Predictive Control, Convergence, Optimal Control, Robotics

1. Introduction

Model Predictive Control (MPC) has achieved remarkable success and become a cornerstone in various applications such as process control, robotics, transportation, and power systems (Mayne, 2016). Sampling-based MPC, in particular, has gained significant attention in recent years due to its ability and flexibility to handle complex dynamics and cost functions and its massive parallelizability on GPUs. The effectiveness of sampling-based MPC has been demonstrated in various applications, including path planning (Helvik and Wittner, 2001; Durrant-Whyte et al., 2012; Nguyen et al., 2021; Argenson and Dulac-Arnold, 2021), and control (Chua et al., 2018; Williams et al., 2017). Particularly, thanks to its accessibility and flexibility to deal with learned dynamics and cost or reward functions, sampling-based MPC has been widely used as a subroutine in model-based reinforcement learning (MBRL) (Mannor et al., 2003; Menache et al., 2005; Ebert et al., 2018; Zhang et al., 2019; Kaiser et al., 2020; Bhardwaj et al., 2020), where the learned dynamics (often in latent space) are highly nonlinear with nonconvex cost functions.

However, there is little theoretical understanding of sampling-based MPC, especially regarding its convergence and contraction properties (e.g., whether and how fast it converges to the optimal control sequence when sampled around some suboptimal control sequence). Moreover, despite its

* † Equal contributions.

empirical success, there is no theoretical guideline for key hyperparameter tunings, especially the temperate λ and the sampling covariance Σ. For instance, one of the most popular and practical sampling-based MPC algorithms, Model Predictive Path Integral Control (MPPI, Williams et al. (2016, 2017)) uses isotropic Gaussians (i.e., there is no correlation between different time steps or different control dimensions) and heuristically tunes temperature. Further, most sampling-based MPC algorithms are unaware of the underlying dynamics and optimization landscape. This paper makes the first step in theoretically understanding the convergence and contraction properties of sampling-based MPC. Based on such analysis, we provide a novel, practical, and effective MPC algorithm by optimally scheduling the covariance matrix Σ. Our contribution is three-fold:

- For the first time, we present the convergence analysis of MPPI. When the total cost is quadratic w.r.t. the control sequence, we prove that MPPI contracts toward the optimal control sequence and precisely characterize the contraction rate as a function of Σ, λ, and system parameters. We then extend beyond the quadratic setting in three cases: (1) strongly convex total cost, (2) linear systems with nonlinear residuals, and (3) general systems.
- An immediate application of our theoretical results is a novel sampling-based MPC algorithm that optimizes the convergence rate, namely, CoVariance-Optimal MPC (`CoVO-MPC`). To do so, `CoVO-MPC` computes the optimal covariance Σ by leveraging the underlying dynamics and cost functions, either in real-time or via offline approximations.
- We thoroughly evaluate the proposed `CoVO-MPC` algorithm in different robotic systems, from Cartpole and quadrotor simulations to real-world scenarios. In particular, compared to the standard MPPI algorithm, the performance is enhanced by 43% to 54% across different tasks.

Collectively, this paper advances the theoretical understanding of sampling-based MPC and offers a practical and efficient algorithm with significant empirical advantages. This work opens avenues for further exploration and refinement of sampling-based MPC strategies, with implications for a broad spectrum of applications.

2. Preliminaries and Related Work

Notations. In our work, H represents the horizon of the MPC. At each time step h, the state is denoted as $x_h \in \mathbb{R}^n$, and the control input is $u_h \in \mathbb{R}^m$. The control input sequence over the horizon is $U = u_{1:H} \in \mathbb{R}^{mH}$, which is obtained by flattening the sequence $[U]_1 = u_1, \ldots, [U]_H = u_H$. We denote random variables using the curly letter $\mathcal{U}$, and U_i represents the i-th sample in a set of N total samples. The notation $\xrightarrow[p]{}$, as used in our work, indicates convergence in probability.

2.1. Optimal Control and MPC

Consider a deterministic optimal control framework:

$$\min_{u_{1:H}} J\left(u_{1:H}\right) = \sum_{h=1}^{H} c_h\left(x_h, u_h\right) + c_f\left(x_{H+1}\right) \quad \text{s.t.} \quad x_{h+1} = f_h\left(x_h, u_h\right), \quad 1 \le h \le H \tag{1}$$

Here $f_h : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ characterizes the dynamics, $c_h : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}_{\ge 0}$ captures the cost function, and $c_f : \mathbb{R}^n \to \mathbb{R}_{\ge 0}$ represents the terminal cost function. This formulation constitutes a constrained optimization problem focused on $J(u_{1:H})$. For a T-step control problem ($T \gg H$), at every step, MPC solves (1) in a receding-horizon manner. In detail, upon observing the current state, MPC sets the x_1 in (1) to be the current state and solves (1), which amounts to solving the optimal control only considering the costs for the next H steps. Afterward, MPC only executes the first control in the solved optimal control sequence, after which the next state can be obtained from

the system, and the procedure repeats. In other words, the H-step problem (1) is a subroutine for MPC. Throughout the theoretical parts (Secs. 3 and 4), we focus on this H-step subroutine.

The optimization in (1) is potentially highly nonconvex due to nonlinear f_h and nonconvex c_h and c_f. Non-sampling-based nonlinear MPC (NMPC) typically calls specific nonlinear programming (NLP) solvers to solve (1), which depends on certain formulations like sequential quadratic programming (SQP, Sideris and Bobrow (2005)), differentiable dynamic programming (DDP, Tassa et al. (2014)), and iterative linear quadratic regulator (iLQR, Carius et al. (2018)). While NMPC has optimality guarantees in simple settings such as linear systems with quadratic costs, its effectiveness is limited to specific systems and solvers (Song et al., 2023; Forbes et al., 2015). In the next section, we introduce sampling-based MPC, a recently popular paradigm that can mitigate these limitations.

2.2. Sampling-based MPC and Applications in Model-based Reinforcement Learning

Recently, sampling-based MPC gained popularity as an alternative to NMPC by employing the zeroth-order optimization strategy (Drews et al., 2018; Mannor et al., 2003; Helvik and Wittner, 2001). Instead of deploying specific NLP solvers to optimize (1), sampling-based MPC samples a set of control sequences from a particular distribution, evaluates the costs of all sequences by rolling them out, optimizes and updates the sampling distribution as a function of samples and their evaluated costs. Unlike NMPC, sampling-based MPC has no particular assumptions on dynamics or cost functions, resulting in a versatile approach in different systems. Moreover, the sampling nature enables massive parallelization on modern GPUs. For example, MPPI is deployed onboard for real-time robotic control (Williams et al., 2016; Pravitra et al., 2020; Sacks et al., 2023), sampling thousands of trajectories at more than 50Hz with a single GPU.

Sampling-based MPC can be categorized by what sampling-based optimizer it deploys. Covariance Matrix Adaptation Evolution Strategy (CMA-ES, Hansen et al. (2003)) adapts sampling mean and covariance using all samples to align with the desired distribution. Cross-Entropy Method (CEM, Botev et al. (2013)) deploys a similar procedure but only considers the low-cost samples. Wagener et al. (2019) unifies different sampling-based MPC approaches by connecting them to online learning with different utility functions and Bregman divergences. Among these variants, MPPI (Williams et al., 2017, 2016) is one of the most popular and practical, and it reformulates the control problem into optimal distribution matching from an information-theoretic perspective. MPPI's performance is sensitive to hyperparameters and system dynamics, leading to several variants that improve its efficiency and robustness, e.g., using Tsallis Divergence (Wang et al., 2021), imposing extra constraints when sampling (Gandhi et al., 2021; Balci et al., 2022), or using learned optimizers (Sacks et al., 2023). Compared to these works, `CoVO-MPC` optimally schedules the sampling covariance followed by our rigorous theoretical analysis.

MPPI has been particularly popular in the Model-based Reinforcement Learning (MBRL) context, an RL paradigm that first learns a dynamics model and optimizes a policy using the learned model. Specifically, largely because of its parallelizability and its flexibility to handle nonlinear and potentially latent learned models, MPPI has been a popular control and planning subroutine in MBRL (Menache et al., 2005; Szita and Lörincz, 2006; Janner et al., 2021; Lowrey et al., 2019; Hafner et al., 2019). Moreover, MPPI and CEM have been widely deployed for uncertainty-aware control and planning with probabilistic models (e.g., PETS (Chua et al., 2018), PlaNet (Hafner et al., 2019)), and integrated with learned value and policy to improve the MBRL performance (e.g., TD-MPC (Hansen et al., 2022), MBPO (Argenson and Dulac-Arnold, 2021)).

Despite its broad applications, there is little theoretical understanding of sampling-based MPC algorithms. In this paper, we aim to provide theoretical characterizations of MPPI's optimality and convergence, based on which we design an improved algorithm `CoVO-MPC`. In addition, `CoVO-MPC` can potentially be an efficient backbone for a broad class of MBRL algorithms.

3. Problem Formulation

While MPPI is an iterative approach with a receding horizon, to streamline the analysis, we consider a single step in MPPI, that is solving a trajectory optimization problem (1) using sampling-based method[1]. Specifically, MPPI rewrites the cost function in (1), as only a function of the control input (i.e., substitute the state x_h with control inputs U). To minimize $J(U)$, MPPI samples the control sequence U and outputs a weighted sum of these samples. Specifically, the control sequence is sampled from a Gaussian distribution: $\mathcal{U} \sim \mathcal{N}(U_{\text{in}}, \Sigma)$, where $U_{\text{in}} \in \mathbb{R}^{mH}$ is the mean and $\Sigma \in \mathbb{R}^{mH \times mH}$ is the covariance matrix. Given the samples, we calculate the output control sequence $U_{\text{out}} \in \mathbb{R}^{mH}$ using a softmax-style weighted sum based on each sample's cost $J(U_i)$:

$$U_{\text{out}} = \frac{\sum_{i=1}^{N} U_i \exp\left(-\frac{J(U_i)}{\lambda}\right)}{\sum_{i=1}^{N} \exp\left(-\frac{J(U_i)}{\lambda}\right)}, \tag{2}$$

where λ is the temperature parameter commonly used in statistical physics (Busetti, 2003) and machine learning. As $\lambda \to 0$, the focus intensifies on the top control sequence, while a larger λ distributes the influence over a set of relatively favorable trajectories. In the next section, we will characterize the single-step convergence properties of MPPI in different systems.

4. Main Theoretical Results

Our primary goal is to investigate the optimality and convergence rate of MPPI. We are driven by the following questions: Under what conditions does the algorithm exhibit outputs close to the optimal solution? And what is the corresponding convergence rate? Moreover, if the convergence is influenced by factors such as $U_{\text{in}}, \lambda, \Sigma$, how can we strategically design the sample covariance to expedite the convergence process?

To answer the above questions, we first analyze the convergence of the single-step MPPI (2) in a quadratic cost's environment. In Sec. 4.1, we show that the expected result of equation (2) contracts to the optimal solution in Theorem 1. Then, in Sec. 4.2, based on the contraction of the expectation output, we further give an optimal design principle for the covariance to better utilize the known dynamics. Moreover, in Sec. 4.3, we address general nonlinear environments and prove that MPPI still keeps the contraction property, though with bounded associated errors.

4.1. Convergence Analysis for Quadratic $J(U)$

We start by considering the total cost $J(U)$ of the following form,

$$J(U) = U^\top D U + U^\top d, \tag{3}$$

which is quadratic in U. One simple but general example that satisfies the above is when the dynamics is LTI or LTV, and the cost is quadratic, which we will elaborate on in Example 1. We will also discuss the generalization to non-quadratic $J(U)$ and nonlinear dynamics in Sec. 4.3.

1. We will discuss the full implementation of MPPI with a receding horizon in Sec. 5.

In this setting, our first result proves the convergence of MPPI, i.e., (2). Specifically, we show that the expected output control sequence contracts, and critically, we provide a precise contraction rate in terms of the system parameter D and the algorithm parameters Σ, λ.

Theorem 1 *Given U_{in} and the sampling distribution $\mathcal{N}(U_{\text{in}}, \Sigma)$, under the assumption that the total cost $J(U)$ is in quadratic form in (3) and D is positive semi-definite, the weighted sum of the samples U_{out}, as in (2), converges in probability to a contraction towards the optimal solution U^* when the number of sample $N \to \infty$ in the following way:*

$$\frac{||U_{\text{out}} - U^*||}{||U_{\text{in}} - U^*||} \underset{p}{\to} \left\| (\frac{2}{\lambda}\Sigma D + I)^{-1} \right\| < 1 \tag{4}$$

Similarly, the expected cost contracts to the optima J^* in the following way:

$$J(U_{\text{out}}) - J^* \underset{p}{\to} J_{\text{out}} - J^* \leq (J(U_{\text{in}}) - J^*) \left\| (I + \frac{2}{\lambda} D^{\frac{1}{2}} \Sigma D^{\frac{1}{2}})^{-2} \right\|, \tag{5}$$

where J_{out} is a constant determined by function J and U_{in}, Σ, D.

Theorem 1 means, when the number of sample N goes to infinity, MPPI enjoys a linear contraction[2] towards the optimal control sequence U^* and the optimal cost J^*. Further, from the contraction rate in both (4) and (5), we can observe that smaller λ leads to faster contraction, which we will discuss more on in Sec. 4.3. However, the faster contraction comes at the cost of higher sample complexity due to a higher variance on U_{out}, as shown by Busetti (2003).

In addition to λ, the contraction rate is also ruled by the matrix $D^{\frac{1}{2}} \Sigma D^{\frac{1}{2}}$, indicating that choosing Σ is vital for the contraction rate, and further, the optimal choice of Σ should depend on the Hessian matrix D. Note that the standard MPPI chooses $\Sigma = \sigma I$, i.e., sampling isotropically over a landscape defined by D, which could yield slow convergence. Based on this observation, we will develop an optimal design scheme of Σ in Sec. 4.2.

Example 1 (Time-variant LQR) *Here is an example of the linear quadratic regulator (LQR) setting with time-variant dynamics and costs. The dynamics is $x_{h+1} = A_h x_h + B_h [U]_h + w_h$ and the cost is given by $J(U) = \sum_{h=1}^{H} \left(x_h^\top Q_h x_h + [U]_h^\top R_h [U]_h \right)$, where $U = ([U]_1, \ldots, [U]_H)$ and $Q_h \succeq 0, R_h \succ 0$. Plugging in the states x_h as a function of U, the cost can be reformulated as $J(U) = U^\top D U + U^\top d + J_0$ where d and J_0 are constants derived from system parameters. And the Hessian is $D = M^\top \mathcal{Q} M + \mathcal{R}$, with*

$$M = \begin{bmatrix} 0 & \cdots & & \\ B_1 & 0 & \cdots & \\ \tilde{A}_{2,2} B_1 & B_2 & 0 & \cdots \\ \vdots & \vdots & & \\ \tilde{A}_{H-1,2} B_1 & \tilde{A}_{H-1,3} B_2 & \cdots & B_{H-1} \end{bmatrix}, \mathcal{Q} = \begin{bmatrix} Q_1 & 0 & \cdots & \\ 0 & Q_2 & \cdots & \\ \vdots & \vdots & & \\ 0 & 0 & \cdots & Q_H \end{bmatrix}, \mathcal{R} = \begin{bmatrix} R_1 & 0 & \cdots & \\ 0 & R_2 & \cdots & \\ \vdots & \vdots & & \\ 0 & 0 & \cdots & R_H \end{bmatrix},$$

and $\tilde{A}_{i,j} = A_i A_{i-1} \cdots A_j$ for $i \geq j$.

2. MPPI can be considered a regularized Newton's method, utilizing the Hessian Matrix D, which typically results in linear convergence (contraction). Moreover, it exhibits superlinear convergence in the proximity of the optimum. Further discussion on this topic will be provided in Appendix A.

Theorem 1 primarily addresses scenarios with quadratic $J(U)$, suitable for time-varying LQR as seen in Example 1. Even though time-varying LQR can be solved using well-known analytical methods, our theoretical exploration still holds significant value, and its underlying principles extend beyond linear systems. Firstly, implementation-wise, nonlinear systems can be linearized around a nominal trajectory, resulting in a Linear Time-Variant (LTV) system to which Theorem 1 directly applies, like in iterative Linear Quadratic Regulator (iLQR, Li and Todorov (2004)) and Differential Dynamic Programming (DDP, Mayne (1966)). Secondly, from a theory perspective, we further discuss the linearization error and more generally, non-quadratic $J(U)$ in Sec. 4.3. Lastly, the analysis within this quadratic framework elucidates the optimal covariance design principle in Sec. 4.2, which we use to design the `CoVO-MPC` algorithm in Sec. 5 that can be implemented beyond linear systems. Empirically (Sec. 6), `CoVO-MPC` demonstrates superior performance across various nonlinear problems, affirming the value of Theorem 1 beyond quadratic formulations.

4.2. Optimal Covariance Design

Theorem 1 proves that MPPI guarantees a contraction to the optima with contraction rate depending on the choice of Σ (see (5)). We now investigate the optimal Σ to achieve a faster contraction rate. Based on (5), it is evident that scaling Σ with a scalar larger than 1 brings better contraction in expectation. However, doing so is equivalent to decreasing λ, which will lead to higher sample complexity according to Sec. 4.1. In light of this, we formulate the optimal covariance Σ design problem as a constrained optimization problem: How can we design Σ to achieve an optimal contraction rate, subject to the constraint that the determinant of the covariance is upper-bounded, that is $\det \Sigma \leq \alpha$. The following theorem optimally solves this constrained optimization problem.

Theorem 2 *Under the constraint that positive semi-definite matrix Σ satisfies $\det \Sigma \leq \alpha$, the contraction rate $\left\|(I + \frac{2}{\lambda} D^{\frac{1}{2}} \Sigma D^{\frac{1}{2}})^{-2}\right\|$ in (5) is minimized when Σ has the following form: The Singular Value Decomposition (SVD) of $\Sigma = V^\top \Lambda V$ shares the same eigenvector matrix V with the SVD of the Hessian matrix $D = V^\top \mathcal{O} V$, and*

$$\frac{[\mathcal{O}]_i^2 [\Lambda_{\text{in}}]_i^2 [\Lambda]_i}{(1 + \frac{2}{\lambda}[\Lambda]_i [\mathcal{O}]_i)^3} = \text{Constant}, \forall i = 1, \cdots, mH, \tag{6}$$

where $\Lambda, \mathcal{O} \in \mathbb{R}^{mH \times mH}$, $\Lambda = diag([\Lambda]_1, [\Lambda]_2, \ldots, [\Lambda]_{mH})$,$\mathcal{O} = diag([\mathcal{O}]_1, [\mathcal{O}]_2, \ldots, [\mathcal{O}]_{mH})$. Further, $\Lambda_{\text{in}} \in \mathbb{R}^{mH}$ is the coordinates of $U_{\text{in}} - U^*$ under the basis formed by the eigenvectors V. In other words, $\Lambda_{\text{in}} = V^\top (U_{\text{in}} - U^*) = [[\Lambda_{\text{in}}]_1, [\Lambda_{\text{in}}]_2, \ldots, [\Lambda_{\text{in}}]_{mH}]^\top$. Lastly, the Constant *in the right-hand-side of (6) is selected subject to the constraint $\det \Sigma = \alpha$.*

While Theorem 2 offers insights into the optimal design of Σ, solving (6) relies on prior knowledge of U^*. In Corollary 3 below, we present an approximation to the solution of (6) in Theorem 2. This approximation is not only efficient to solve but also eliminates the need for knowing U^*. These advantages come under the assumptions of an isotropic gap between U^* and U_{in} and a small λ regime. The isotropy gap assumption is necessary as we do not have additional information about U^*[3]. A small λ regime is also practical since Theorem 1 implies a smaller λ yields faster contraction given a sufficiently large sample number N. Given modern GPU's capabilities in large-scale parallel sampling, we can generate sufficient samples for small values of λ (in practice, MPPI often uses $\lambda < 0.01$), which justifies the adoption of the small λ regime in Corollary 3. Notably, this

3. We show that the isotropy assumption is minimizing the max cost from a family of gap $U_{\text{in}} - U^*$ in the Appendix A.

approximation serves as the foundational design choice for CoVO-MPC introduced in Sec. 5, and empirical experiments consistently validate its feasibility.

Corollary 3 *Under the assumption $U_{\text{in}} - U^* \sim \mathcal{N}(0, I)$, when $\lambda \to 0$, asymptotically, the optimal solution of (6) is:*

$$\lim_{\lambda \to 0, N \to \infty} \Sigma = (\alpha \det D^{1/2})^{\frac{1}{mH}} D^{-1/2} \tag{7}$$

Note that in both Theorem 2 and Corollary 3, the choice of Σ depends on system parameters. The underlying concept of this design principle is to move away from isotropic sampling of the control sequence. Instead, the sample distribution is tailored to align with the structure of the system dynamics and cost functions. In essence, if we view the covariance matrix as an ellipsoid in the control action space, Theorem 2 and Corollary 3 implies that the optimal covariance matrix should be compressed in the direction where the total cost $J(U)$ is smoother and stretched along the direction aligned with the J's gradient. Guided by this optimal covariance matrix design principle, we introduce a practical algorithm in Sec. 5 that incorporates this optimality into the MPPI framework. This framework computes the Hessian matrix and designs the covariance matrix, thus leveraging the system structure for improved performance.

4.3. Generalization Beyond the Quadratic $J(U)$

Sec. 4.1 gives the convergence properties of MPPI when $J(U)$ is quadratic, but in practice, MPPI works well beyond quadratic $J(U)$. Therefore, in this section, we extend the convergence analysis in Sec. 4.1 to general non-quadratic total costs. Specifically, we consider three different settings, from strongly convex (not necessarily quadratic) $J(U)$, linear systems with nonlinear residuals, to general systems.

In many cases, the total cost function J demonstrates strong convexity without necessarily being quadratic. A typical example is the time-varying LQR (Example 1) with the quadratic term $x^\top Q x$ in the cost being replaced with a strongly convex function. In light of this, we now study the convergence of MPPI under a β-strongly convex total cost function $J(U)$.

Theorem 4 (Strongly convex J. Informal version of Theorem 4 in Appendix A.3) *Given a β-strongly convex function $J(U)$ which has Lipschitz continuous derivatives, that $\frac{\partial J}{\partial U}$ is L_d-Lipschitz, U_{out} converges to a neighborhood of the optimal point U^* in probability as the number of samples N tends to infinity, i.e., $U_{\text{out}} \underset{p}{\to} U_c$. Here U_c satisfies $\|U_c - U^*\| \leq \frac{\lambda}{\beta} \frac{2\|\Sigma^{-1}\|}{(1+\frac{\lambda}{\beta\|\Sigma\|})} \|U_{\text{in}} - U^*\| + \|U_{\text{error}}\|$, where $\|U_{\text{error}}\| \sim O(\frac{\sqrt{\lambda}}{\beta})$.*

Theorem 4 demonstrates MPPI maintains a contraction rate of $O(\frac{\lambda}{\beta})$ with a small λ, approaching 0 when $\lambda \ll \beta$. The remaining residual error U_{error}, also of $O(\frac{\lambda}{\beta})$, suggests MPPI's convergence to a neighborhood around the optimal point with size $O(\frac{\lambda}{\beta})$. The contraction rate in strongly convex $J(U)$ differs from the quadratic case (Sec. 4.1) due to the difference between the quadratic density function in Gaussian. The non-quadratic $J(U)$ has a slightly looser bound in terms of constant.

Beyond strongly convex $J(U)$, we next consider nonlinear dynamics. Specifically, we consider the same cost function as in Example 1 with the following dynamics: $x_{h+1} = Ax_h + B[U]_h + w + g([U]_h)$, where w is a constant, and g represents nonlinear residual dynamics. We then recast the total cost as $J(U) = U^\top D U + d^\top U + J_{\text{res}}(U)$ by retaining all higher-order terms in $J_{\text{res}}(U)$. Subsequently, we establish an exponential family using D, d, and $J_{\text{res}}(U)$ as sufficient statistics.

Given the characteristics of the residual dynamics, we can bound the variations in these three statistics relative to the quadratic total cost. Utilizing the properties of the exponential family, we then show that, with bounded residual dynamics g, contraction is still guaranteed to some extent.

Theorem 5 (Linear systems with nonlinear residuals. Informal version of Theorem 5 in Appendix A.3). *Given the residual dynamics g, U_{out} contracts in the same way as in (4) with contraction error $\|U_{\text{error}}\| \sim \frac{L_1}{\lambda}\|Q\|\left(O(\left\|A^{3H}\right\|)C_3 + O(\left\|A^{2H}\right\|)C_2\right)$ with L_1 as a Lipschitz constant from the exponential family, and C_2, C_3 as constants that coming from the bounded residual dynamics g.*

Additionally, it is also worthnoting that the Lipschitz constant L_1 is task-specific since different tasks have different sufficient statistics J_{res} within the exponential family. Given Theorem 5 for residual dynamics, a direct and straightforward corollary arises for general systems, with direct access to the cost function, specifically J_{res}.

Corollary 6 (General systems. Informal version of Theorem 6 in Appendix A.3.) *U_{out} contracts the same as (4) with a contraction error, $\|U_{\text{error}}\| \sim O(\frac{L_1}{\lambda}(\|D - D'\|))$, where D' is the Hessian under the residual dynamic g, and D is the Hessian without the residual dynamic.*

5. The CoVariance-Optimal MPC (CoVO-MPC) Algorithm

Corollary 3 in Sec. 4.2 provides the optimal covariance matrix Σ as a function of the Hessian matrix D. We define the mapping in (7) as $\Sigma = \mathbf{C}(D)$. Deploying $\mathbf{C}(\cdot)$ at every time step in the sampling-based MPC framework, we propose `CoVO-MPC` (Alg. 1), a general sampling-based MPC algorithm.

Algorithm 1: CoVO-MPC

Input: H; N; T; $c_{1:T}$; $\mathbf{C}(\cdot)$; `shift`$(\cdot)$;

1 **for** $t = 1 : T$ **do**
2 $\quad\Sigma_t \leftarrow \mathbf{C}(D_t = \nabla^2 J_t(U_{\text{in}|t}))$
3 $\quad$Sample $U_{i|t} \sim \mathcal{N}(U_{\text{in}|t}, \Sigma_t)$ for $1 \le i \le N$
4 $\quad$Compute weight $\kappa_{i|t} \leftarrow \frac{\exp(-\frac{J_t(U_{i|t})}{\lambda})}{\sum_{i=1}^{N}\exp(-\frac{J_t(U_{i|t})}{\lambda}))}$
5 $\quad U_{\text{out}|t} \leftarrow \sum_{i=1}^{N}\kappa_{i|t}U_{i|t}$
6 $\quad$Execute $u_t \leftarrow [U_{\text{out}|t}]_1$, receive x_{t+1}
7 $\quad U_{\text{in}|t+1} \leftarrow$ `shift`$(U_{\text{out}|t})$
8 **end**

As shown in Line 2 of Alg. 1, at each time step, `CoVO-MPC` will first generate an optimal sampling covariance matrix Σ_t based on the cost's Hessian matrix around the sampling mean $\nabla^2 J_t(U_{\text{in}|t})$. Next, we will follow the MPPI framework to calculate control sequences from the sampling distribution (Lines 3 to 5) and execute the first control command in the sequence (Line 6). After that, we will shift the sampling mean forward to the next time step using the `shift` operator (Line 7) and repeat the process.

The optimal covariance design (Line 2) is critical for `CoVO-MPC`, which requires computing the Hessian matrix $D_t = \nabla^2 J_t(U_{\text{in}|t}))$ in real-time. The total cost $J_t(U_{\text{in}|t}) = \sum_{h=1}^{H} c_{t+h}(x_{t+h|t}, [U_{\text{in}|t}]_h)$ is gathered by rolling out the sampling mean sequence $U_{\text{in}|t}$ with the dynamics initialized at x_t, where $x_{t+h|t}$ is the h^{th} rollout state and c_t is the running cost at time step t. To ensure D_t to be positive definite, we add a small positive value to the diagonal elements of D_t such that $D_t \succeq \epsilon I \succ 0$, which is equivalent to adding a small quadratic control penalty term to the cost function.

Offline covariance matrix approximation. Computing D_t and Σ_t in real time could be expensive. Therefore, we propose an offline approximation variant of `CoVO-MPC`. In this variant, we cache the covariance matrices across all t offline by rolling out the dynamics using a nominal controller (e.g., PID). More specifically, offline, we calculate the whole sequence of covariance matrix $\Sigma_{1:T}^{\text{off}}$ in simulation using the nominal controller. Then, online, Σ_t^{off} serves as an approximation of the optimal covariance matrix. The details can be found in Appendix B.1. Empirically, we observe that this offline approximation performs marginally worse than `CoVO-MPC` with less computation.

6. Experiments

We evaluate CoVO-MPC in three nonlinear robotic systems. To demonstrate the effectiveness of CoVO-MPC, we first compare it with the standard MPPI, in both simulation and the real world. To further understand the difference between CoVO-MPC and MPPI, we quantify their computational costs and visualize their cost distributions. Our results show that CoVO-MPC significantly outperforms MPPI with a more concentrated cost distribution.

6.1. Tasks and Implementations

We choose three tasks (illustrated as Fig.3 in Appendix B.2): (a) CartPole environment with force applied to the car as control input (Lange, 2023). (b) Quadrotor follows zig-zag infeasible trajectories. The action space is the desired thrust and body rate ($\dim(u) = 4$). (c) Quadrotor (real): same tasks as (b) but on a real-world quadrotor platform based on the Bitcraze Crazyflie 2.1 (Huang et al., 2023; Shi et al., 2021).

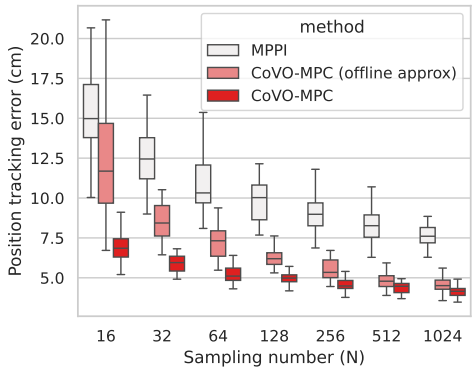

Figure 1: Quadrotor tracking errors as the sample number N increases.

For both the simulation and the real quadrotor, all results were evaluated across 3 different trajectories, each repeatedly executed 10 times. The implementation details of all tasks can be found in Appendix B.2. All tasks use the same hyperparameter (Table 3 in Appendix B.2). It is worth noting that Alg. 1 ensures the covariance matrix's determinant of CoVO-MPC is identical to that of the MPPI baseline, to keep their sampling volumes the same.

6.2. Performance and Computational Cost

Table 1 illustrates the evaluated cost for each task. CoVO-MPC outperforms MPPI across all tasks with varying performance gains from 43% to 54%. In simulations, the performance of CoVO-MPC (offline approx.) stays close to CoVO-MPC by approximating the Hessian matrix offline using simple nominal PID controllers, which implies the effectiveness of CoVO-MPC comes from the optimized non-trivial pattern of Σ_t (Fig. 4 in Appendix C) rather than its precise values. When transferred to the real-world quadrotor control, the offline approximation's performance degrades due to sim-2-real gaps but still outperforms MPPI by a significant margin (22%). The real-world tracking results are visualized in Fig. 2, where CoVO-MPC can effectively track the desired triangle trajectory while MPPI fails.

Tasks	CartPole	Quadrotor	Quadrotor (real)
CoVO-MPC	$\mathbf{0.70 \pm 0.21}$	$\mathbf{3.71 \pm 0.37}$	$\mathbf{8.36 \pm 4.86}$
CoVO-MPC (offline approx.)	0.71 ± 0.23	3.86 ± 0.34	12.09 ± 6.33
MPPI	1.52 ± 0.46	6.48 ± 0.64	15.49 ± 5.82

Table 1: The cost associated with CoVO-MPC compared with that of MPPI while tracking an infeasible zig-zag trajectory. For Quadrotor and Quadrotor (real), the value indicates tracking error in centimeters.

Besides optimality, another important aspect of sampling-based MPC is its sampling efficiency. To understand CoVO-MPC's sampling efficiency, we evaluate all methods in quadrotor simulation with various sampling numbers N. The results in Fig. 1 show that CoVO-MPC and its approximated variant with fewer samples outperforms the standard MPPI algorithm with much more samples.

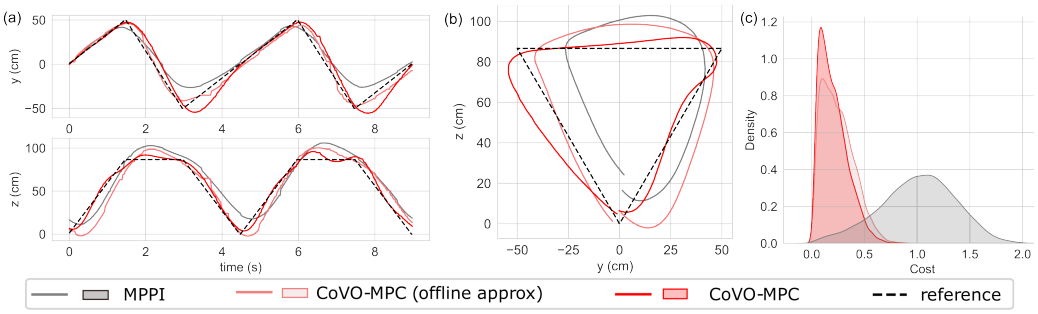

Figure 2: (a-b) Real-world quadrotor trajectory tracking results. CoVO-MPC can track the challenging infeasible triangle trajectory closer than MPPI. (c) The cost distribution of sampled trajectories at a certain time step in Quadrotor simulation. The cost of CoVO-MPC is more concentrated and has a lower mean than MPPI.

To further understand the performance difference between CoVO-MPC and MPPI, we plot the cost distributions of their sampled trajectories at a particular time step in Fig. 2(c). The cost of CoVO-MPC is more concentrated and has a lower mean than MPPI, which directly implies the effectiveness of the optimal covariance procedure of CoVO-MPC. In other words, CoVO-MPC can automatically adapt to different optimization landscapes while MPPI cannot.

As expected, the superior performance of CoVO-MPC comes with extra computational costs for the Hessian matrix D_t and the optimal covariance Σ_t. Therefore, we quantify extra computational burdens in Table 2. While CoVO-MPC indeed requires more computation, the offline approximation of CoVO-MPC has the same computational cost as MPPI but with better performance.

Algorithm	**Online Time (ms)**	**Offline Time (ms)**
CoVO-MPC	9.22 ± 0.39	0.26 ± 0.47
Offline approx.	2.03 ± 0.53	2179.59 ± 4.57
MPPI	2.17 ± 0.67	2.16 ± 0.29

Table 2: Computational time comparison.

7. Limitations and Future Work

While CoVO-MPC introduces new theoretical perspectives and shows compelling empirical performance, it currently relies on the differentiability of the system. Our future efforts will focus on adapting our theory and algorithm to more general scenarios. We also aim to investigate the finite-sample analysis of sampling-based MPC with a focus on the variance of the algorithm output. Generalizing our results to receding-horizon settings (Lin et al., 2021; Yu et al., 2020) is also interesting. Moreover, CoVO-MPC holds substantial promise for MBRL, so we plan to integrate CoVO-MPC with the MBRL framework, leveraging learned dynamics, value functions, or policies. We also recognize there exist efficient online approximations of CoVO-MPC, e.g., reusing previous Hessians or covariances, which could lead to more efficient implementations.

8. Acknowledgment

This work was supported by NSF Grants 2154171, 2339112, CMU CyLab Seed Funding and C3 AI Institute.

References

Arthur Argenson and Gabriel Dulac-Arnold. Model-Based Offline Planning, March 2021.

Isin M. Balci, Efstathios Bakolas, Bogdan Vlahov, and Evangelos Theodorou. Constrained Covariance Steering Based Tube-MPPI, April 2022.

Mohak Bhardwaj, Ankur Handa, Dieter Fox, and Byron Boots. Information Theoretic Model Predictive Q-Learning, May 2020.

Zdravko I. Botev, Dirk P. Kroese, Reuven Y. Rubinstein, and Pierre L'Ecuyer. Chapter 3 - The Cross-Entropy Method for Optimization. In C. R. Rao and Venu Govindaraju, editors, Handbook of Statistics, volume 31 of Handbook of Statistics, pages 35–59. Elsevier, January 2013. doi: 10.1016/B978-0-444-53859-8.00003-5.

Franco Busetti. Simulated annealing overview. World Wide Web URL www. geocities. com/francorbusetti/saweb. pdf, 4, 2003.

Jan Carius, René Ranftl, Vladlen Koltun, and Marco Hutter. Trajectory Optimization With Implicit Hard Contacts. IEEE Robotics and Automation Letters, 3(4):3316–3323, October 2018. ISSN 2377-3766. doi: 10.1109/LRA.2018.2852785.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep Reinforcement Learning in a Handful of Trials using Probabilistic Dynamics Models, November 2018.

Paul Drews, Grady Williams, Brian Goldfain, Evangelos A. Theodorou, and James M. Rehg. Vision-Based High Speed Driving with a Deep Dynamic Observer, December 2018.

Hugh Durrant-Whyte, Nicholas Roy, and Pieter Abbeel. Cross-Entropy Randomized Motion Planning. In Robotics: Science and Systems VII, pages 153–160. MIT Press, 2012.

Frederik Ebert, Chelsea Finn, Sudeep Dasari, Annie Xie, Alex Lee, and Sergey Levine. Visual Foresight: Model-Based Deep Reinforcement Learning for Vision-Based Robotic Control, December 2018.

Michael G. Forbes, Rohit S. Patwardhan, Hamza Hamadah, and R. Bhushan Gopaluni. Model Predictive Control in Industry: Challenges and Opportunities. IFAC-PapersOnLine, 48(8):531–538, January 2015. ISSN 2405-8963. doi: 10.1016/j.ifacol.2015.09.022.

Manan S. Gandhi, Bogdan Vlahov, Jason Gibson, Grady Williams, and Evangelos A. Theodorou. Robust Model Predictive Path Integral Control: Analysis and Performance Guarantees. IEEE Robotics and Automation Letters, 6(2):1423–1430, April 2021. ISSN 2377-3766, 2377-3774. doi: 10.1109/LRA.2021.3057563.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning Latent Dynamics for Planning from Pixels, June 2019.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal Difference Learning for Model Predictive Control, July 2022.

Nikolaus Hansen, Sibylle D. Müller, and Petros Koumoutsakos. Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES). Evolutionary Computation, 11(1):1–18, March 2003. ISSN 1063-6560. doi: 10.1162/106365603321828970.

Bjarne E. Helvik and Otto Wittner. Using the Cross-Entropy Method to Guide/Govern Mobile Agent's Path Finding in Networks. In Samuel Pierre and Roch Glitho, editors, Mobile Agents for Telecommunication Applications, Lecture Notes in Computer Science, pages 255–268, Berlin, Heidelberg, 2001. Springer. ISBN 978-3-540-44651-4. doi: 10.1007/3-540-44651-6_24.

Kevin Huang, Rwik Rana, Alexander Spitzer, Guanya Shi, and Byron Boots. Datt: Deep adaptive trajectory tracking for quadrotor control. In 7th Annual Conference on Robot Learning, 2023.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to Trust Your Model: Model-Based Policy Optimization, November 2021.

Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, Roy H. Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, Afroz Mohiuddin, Ryan Sepassi, George Tucker, and Henryk Michalewski. Model-Based Reinforcement Learning for Atari, February 2020.

Robert Tjarko Lange. Reinforcement Learning Environments in JAX, November 2023.

Weiwei Li and Emanuel Todorov. Iterative linear quadratic regulator design for nonlinear biological movement systems. In Proceedings of the First International Conference on Informatics in Control, Automation and Robotics, pages 222–229, Setúbal, Portugal, 2004. SciTePress - Science and and Technology Publications. ISBN 978-972-8865-12-2. doi: 10.5220/0001143902220229.

Yiheng Lin, Yang Hu, Guanya Shi, Haoyuan Sun, Guannan Qu, and Adam Wierman. Perturbation-based regret analysis of predictive control in linear time varying systems. Advances in Neural Information Processing Systems, 34:5174–5185, 2021.

Kendall Lowrey, Aravind Rajeswaran, Sham Kakade, Emanuel Todorov, and Igor Mordatch. Plan Online, Learn Offline: Efficient Learning and Exploration via Model-Based Control, January 2019.

Shie Mannor, Reuven Rubinstein, and Yohai Gat. The Cross Entropy Method for Fast Policy Search. Proceedings, Twentieth International Conference on Machine Learning, 2003.

David Mayne. A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems. International Journal of Control, 3(1):85–95, 1966.

David Mayne. Robust and stochastic model predictive control: Are we going in the right direction? Annual Reviews in Control, 41:184–192, January 2016. ISSN 1367-5788. doi: 10.1016/j.arcontrol.2016.04.006.

Ishai Menache, Shie Mannor, and Nahum Shimkin. Basis Function Adaptation in Temporal Difference Reinforcement Learning. Annals of Operations Research, 134(1):215–238, February 2005. ISSN 0254-5330, 1572-9338. doi: 10.1007/s10479-005-5732-z.

Tung Nguyen, Rui Shu, Tuan Pham, Hung Bui, and Stefano Ermon. Temporal Predictive Coding For Model-Based Planning In Latent Space, June 2021.

Jintasit Pravitra, Kasey A. Ackerman, Chengyu Cao, Naira Hovakimyan, and Evangelos A. Theodorou. $\mathscr{L}1$-Adaptive MPPI Architecture for Robust and Agile Control of Multirotors. In 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 7661–7666, October 2020. doi: 10.1109/IROS45743.2020.9341154.

Jacob Sacks, Rwik Rana, Kevin Huang, Alex Spitzer, Guanya Shi, and Byron Boots. Deep Model Predictive Optimization, October 2023.

Guanya Shi, Wolfgang Hönig, Xichen Shi, Yisong Yue, and Soon-Jo Chung. Neural-swarm2: Planning and control of heterogeneous multirotor swarms using learned interactions. IEEE Transactions on Robotics, 38(2):1063–1079, 2021.

A. Sideris and J.E. Bobrow. An efficient sequential linear quadratic algorithm for solving nonlinear optimal control problems. In Proceedings of the 2005, American Control Conference, 2005., pages 2275–2280 vol. 4, June 2005. doi: 10.1109/ACC.2005.1470308.

Yunlong Song, Angel Romero, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza. Reaching the limit in autonomous racing: Optimal control versus reinforcement learning. Science Robotics, 8(82):eadg1462, September 2023. doi: 10.1126/scirobotics.adg1462.

István Szita and András Lörincz. Learning tetris using the noisy cross-entropy method, 2006.

Yuval Tassa, Nicolas Mansard, and Emo Todorov. Control-limited differential dynamic programming. In 2014 IEEE International Conference on Robotics and Automation (ICRA), pages 1168–1175, May 2014. doi: 10.1109/ICRA.2014.6907001.

Nolan Wagener, Ching-an Cheng, Jacob Sacks, and Byron Boots. An Online Learning Approach to Model Predictive Control. In Robotics: Science and Systems XV. Robotics: Science and Systems Foundation, June 2019. ISBN 978-0-9923747-5-4. doi: 10.15607/RSS.2019.XV.033.

Ziyi Wang, Oswin So, Jason Gibson, Bogdan Vlahov, Manan S. Gandhi, Guan-Horng Liu, and Evangelos A. Theodorou. Variational Inference MPC using Tsallis Divergence, April 2021.

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), pages 1433–1440, May 2016. doi: 10.1109/ICRA.2016.7487277.

Grady Williams, Nolan Wagener, Brian Goldfain, Paul Drews, James M. Rehg, Byron Boots, and Evangelos A. Theodorou. Information theoretic MPC for model-based reinforcement learning. In 2017 IEEE International Conference on Robotics and Automation (ICRA), pages 1714–1721, May 2017. doi: 10.1109/ICRA.2017.7989202.

Chenkai Yu, Guanya Shi, Soon-Jo Chung, Yisong Yue, and Adam Wierman. The power of predictions in online control. Advances in Neural Information Processing Systems, 33:1994–2004, 2020.

Marvin Zhang, Sharad Vikram, Laura Smith, Pieter Abbeel, Matthew J. Johnson, and Sergey Levine. SOLAR: Deep Structured Representations for Model-Based Reinforcement Learning, June 2019.