Data-driven Acceleration of MPC with Guarantees

Agustin Castellano ACASTE11@JHU.EDU

Shijie Pan SPAN34@JHU.EDU

Enrique Mallada MALLADA@JHU.EDU

Dept. of Electrical and Computer Engineering
Johns Hopkins University

Editors: G. Sukhatme, L. Lindemann, S. Tu, A. Wierman, N. Atanasov

Abstract

Model Predictive Control (MPC) is a powerful framework for optimal control but can be too slow for low-latency applications. We present a data-driven framework to accelerate MPC by replacing online optimization with a nonparametric policy constructed from offline MPC solutions. Our policy is greedy with respect to a constructed upper bound on the optimal cost-to-go, and can be implemented as a nonparametric lookup rule that is orders of magnitude faster than solving MPC online. Our analysis shows that under sufficient coverage condition of the offline data, the policy is recursively feasible and admits provable, bounded optimality gap. These conditions establish an explicit trade-off between the amount of data collected and the tightness of the bounds. Our experiments show that this policy is between 100 and 1000 times faster than standard MPC, with only a modest hit to optimality, showing potential for real-time control tasks.

Keywords: Explicit MPC, Approximate MPC, Nonlinear systems, Nonparametric methods.

1. Introduction

Model Predictive Control (MPC) is a powerful framework for optimal control of constrained, high-dimensional dynamical systems (Mayne, 2014). Born from the process control industry in the late 1970s (Richalet et al., 1978; Cutler and Ramaker, 1980), it has since matured and been applied to myriad of different industries and applications, including aerospace (Di Cairano and Kolmanovsky, 2018), automotive (Hrovat et al., 2012), thermal control in buildings (Drgoňa et al., 2020), and more (Forbes et al., 2015). At the core of MPC is the solution approach of iteratively solving a receding-horizon constrained optimization problem (Mayne et al., 2000). Trade-offs between the problem horizon and the quality of the solutions have been extensively studied (Grüne et al., 2010; Reble and Allgöwer, 2012; Worthmann, 2012). Notwithstanding, one of the core challenges of MPC—even more-so for embedded applications—is how to get *fast, high-quality* control in an online setting.

Ways of speeding up MPC have been explored for decades (Garcia et al., 1989). In the case of linear systems with quadratic costs and polytopic constraints, it is a well-known fact that the optimal controller is piecewise affine (Bemporad et al., 2002). There is a substantial body of work that leverages this fact in *explicit* MPC (Alessio and Bemporad, 2009), where one seeks to learn the optimal controller for each polyhedral region, by combining neural networks with projection schemes (Chen et al., 2018) or with multiparametric quadratic programming (Maddalena et al., 2020). It is out of the scope of this work to present a comprehensive review of MPC. Popular surveys include Garcia et al. (1989) and the more recent work of Alessio and Bemporad (2009).

© 2026 A. Castellano, S. Pan & E. Mallada.

In this work, we use offline MPC to build a data-driven, nonparametric policy that is guaranteed feasible and achieves a desired optimality gap. The core idea to establish feasibility will be to solve a more conservative MPC problem during the offline phase. Our policy will enjoy by design good performance in the vecinity of each offline solution. Specifically, we make the following contributions:

1. We present a novel nonparametric policy that approximately solves MPC problems. This policy is built with offline data from a more constrained MPC solution.
2. We establish rigorous requirements on the offline data that ensure recursive feasibility over the whole domain and bounds on the optimality gap.
3. Empirically, we show that this policy can be implemented efficiently on a GPU, and during inference is orders of magnitude faster than online MPC.

The rest of the paper is organized as follows. **Section** 2 reviews standard MPC. **Section** 3 presents the conservative problem we solve. **Section** 4 defines the data-driven nonparametric policy, and establishes sufficient data-coverage conditions that guarantee feasibility and a desired performance. We present two algorithm in **Section** 5: one based on stochastic sampling (Algorithm 1) and another one based on sequential splitting of the state-space domain (Algorithm 2), that upon termination provide guarantees of sequential feasibility and suboptimality. Experiments in **Section** 6 show our verification algorithm in action and empirically contrast trade-offs between performance and controller latency both for our method and standard MPC.

2. Preliminaries

We are interested in solving the following problem:

$$J(\mathbf{x}_0) \triangleq \min_{\mathbf{u}_{0:T-1}} \sum_{t=0}^{T-1} \gamma^t c(\mathbf{x}_t, \mathbf{u}_t) + F(\mathbf{x}_T) \tag{1a}$$

$$\text{subject to: } \mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t), \qquad t = 0, \dots, T-1 \tag{1b}$$

$$\mathbf{x}_t \in \mathbb{X}, \qquad t = 1, \dots, T-1 \tag{1c}$$

$$\mathbf{u}_t \in \mathbb{U}, \qquad t = 0, \dots, T-1 \tag{1d}$$

where $\mathbf{u}_{0:T-1} \triangleq [\mathbf{u}_0\ \mathbf{u}_1 \dots \mathbf{u}_{T-1}]$ is the sequence of controls, the problem horizon satisfies $1 \leq T \leq \infty$, and $\gamma \in (0, 1]$ is a discount factor. Stage costs $c(\cdot, \cdot)$ are nonnegative, the feasible sets are compact and satisfy $\mathbb{X} \subseteq \mathbb{R}^n$, $\mathbb{U} \subseteq \mathbb{R}^m$. Terminal state constraints (if any) are encoded via $F : \mathbb{X} \to \mathbb{R}_{\geq 0} \cup \{+\infty\}$. Without loss of generality, we assume that the origin is an equilibrium point of f, i.e. $\mathbf{0} = f(\mathbf{0}, \mathbf{0}) \in \mathbb{X}$. Further, we make the following additional assumptions.

Assumption 1 (Lipschitz dynamics) *The map f is L_f-Lipschitz in $\mathbf{x}$ and L_u-Lipschitz in $\mathbf{u}$:*

$$\left|f(\mathbf{x}, \mathbf{u}) - f(\mathbf{x}', \mathbf{u}')\right| \leq L_f \left\|\mathbf{x} - \mathbf{x}'\right\|_{\mathbb{X}} + L_u \left\|\mathbf{u} - \mathbf{u}'\right\|_{\mathbb{U}} \quad \forall \mathbf{x}, \mathbf{x}' \in \mathbb{X}, \ \ \forall \mathbf{u}, \mathbf{u}' \in \mathbb{U}\ , \tag{2}$$

where $\|\cdot\|_{\mathbb{X}}$ and $\|\cdot\|_{\mathbb{U}}$ are appropiate norms on $\mathbb{X}$ and $\mathbb{U}$.

We omit the dependence of $\|\cdot\|_{\square}$ on $\mathbb{X}$ and $\mathbb{U}$ when it is clear from context.

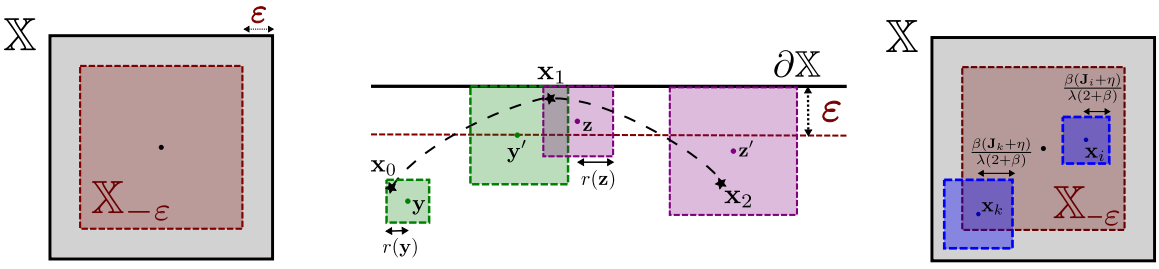

Figure 1: Left: Original constraint set $\mathbb{X}$ and its erosion $\mathbb{X}_{-\varepsilon}$. Middle: Feasibility certificates under our framework. The trajectory $(\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \ldots)$ marked with '$\star$' is produced by our policy. Optimal transitions $\mathbf{y} \overset{\pi^\star}{\to} \mathbf{y}'$ and $\mathbf{z} \overset{\pi^\star}{\to} \mathbf{z}'$ are precomputed offline (by solving (4)) and stored in a dataset $\mathcal{D}$. The control associated with each state (e.g. $\mathbf{y}$) in the dataset is also feasible in a neighborhood of that point (the ball with radius $r(\mathbf{y})$, see Prop. 2). Right: Performance guarantees for our policy (Theorem 2). Each triplet $(\mathbf{x}_i, \mathbf{u}_i, \mathbf{J}_i)$ in the dataset certifies a ball $\mathbb{B}\left(\mathbf{x}_i, \frac{\beta(\mathbf{J}_i+\eta)}{\lambda(2+\beta)}\right)$, wherein $\frac{J^\pi(\mathbf{x})-J(\mathbf{x},\varepsilon)}{J(\mathbf{x},\varepsilon)+\eta} \leq \beta$ for any $\mathbf{x}$ in that ball.

Assumption 2 (Stationarity of optimal solutions) *The optimal policy $\pi^\star : \mathbb{X} \to \mathbb{U}$ for (1) is stationary.*

It follows from Assumption 2 that the optimal cost-to-go $J(\cdot)$ is also stationary, and satisfies the Bellman Equation (Bellman, 1954; Bertsekas, 2012):

$$J(\mathbf{x}) = c\left(\mathbf{x}, \pi^\star(\mathbf{x})\right) + \gamma \cdot J\left(f(\mathbf{x}, \pi^\star(\mathbf{x}))\right). \tag{3}$$

The preceeding assumption is not overly restrictive: it captures (among other cases) infinite-horizon, linear-quadratic control (LQR) (Bertsekas, 2012, Ch. 3) and shortest path problems with a positive weighted graph (Bertsekas, 2012, Ch. 2).

Typical MPC solution scheme and limitations

In Model Predictive Control, the standard approach is to solve (1) over a smaller horizon $H \ll T$, obtain the optimal sequence of controls $\mathbf{u}_0, \ldots, \mathbf{u}_{H-1}$, apply only the first one ($\mathbf{u}_0$) to the system. Then, this procedure is repeated from the new state. This approach is also called receding horizon control (Mattingley et al., 2011; Rawlings and Muske, 2002). It provides a trade-off between computation time (smaller H) at the expense of solution quality (larger H). It always produces a stationary controller (Mayne et al., 2000, Section 3.8.1.), although one of its main drawbacks is that, by its own, it does not guarantee recursive feasibility (Löfberg, 2012). It is common in the MPC literature to add a terminal constraint $\mathbf{x}_H \in \mathbb{X}_H$, with $\mathbb{X}_H$ being a constraint-satistfying control invariant set (Chen and Allgöwer, 1998; Mayne et al., 2000), or to carefully design a terminal cost (like $F(\cdot)$ in (1a)) that yields recursive feasibility (e.g. using a Control Lyapunov function as $F(\cdot)$ (Jadbabaie et al., 2002)). However, computing control invariant sets or Lyapunov functions for general non-linear systems is a hard problem in and of itself (Blanchini, 1999).

In this work, we overcome these limitations by using offline, precomputed solutions to a slightly conservative version of (1). These solutions are used to define our data-driven policy, which will enjoy—by design and with sufficient data—recursive feasibility.

3. Offline solution strategy

During the offline phase, we collect data by solving a more conservative version of Problem (1), which we describe now. Let $\mathbb{B}_\varepsilon$ be the ε-ball in $\mathbb{R}^n$ centered at the origin. We define the *erosion* of $\mathbb{X}$ at level ε as:

$$\mathbb{X}_{-\varepsilon} \triangleq \mathbb{X} \ominus \mathbb{B}_\varepsilon = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{x} + \mathbb{B}_\varepsilon \subseteq \mathbb{X}\} \ ,$$

where $\ominus$ denotes Pontryagin/Minkowski difference (Blanchini et al., 2008, Ch. 3) (see Figure 1). Consider the following generalization of problem (1):

$$J(\mathbf{x}_0, \varepsilon) \triangleq \min_{\mathbf{u}_{0:T-1}} \text{(1a)} \tag{4a}$$

$$\text{subject to: (1b), (1d)} \tag{4b}$$

$$\mathbf{x}_t \in \mathbb{X}_{-\varepsilon}, \qquad t = 1, \dots, T-1 \tag{4c}$$

The cost-to-go $J(\cdot, \cdot)$ above is a mapping from $\mathbb{X} \times \mathbb{R}_{\geq 0}$ to $\mathbb{R}_{\geq 0} \cup \{+\infty\}$. Note that $J(\cdot, 0)$ reduces to (1). We make two additional assumptions.

Assumption 3 (Shrunk problem is feasible) *There exists a small, positive ε such that Problem (4) is feasible for all $\mathbf{x}_0 \in \mathbb{X}$.*

We highlight that the assumption above is for any $\mathbf{x}_0 \in \mathbb{X}$, and not for any $\mathbf{x}_0 \in \mathbb{X}_{-\varepsilon}$. This means initial conditions $\mathbf{x}_0 \in \mathbb{X} \setminus \mathbb{X}_{-\varepsilon}$ are required to "jump in" to $\mathbb{X}_{-\varepsilon}$ in one step[1]. The trajectory $\mathbf{z} \to \mathbf{z}'$ in Figure 1 is an example of this behavior.

Assumption 4 (Cost-to-go of conservative problem is Lipschitz)

1. *There exists $L > 0$ such that, for any ε satisfying the assumption above, we have:*

$$J(\mathbf{x}_0, \varepsilon) - J(\mathbf{x}_0, 0) \leq L \cdot \varepsilon \quad \forall\, \mathbf{x}_0 \in \mathbb{X} \ .$$

2. *There exists $L_J > 0$ such that for any ε satisfying the assumption above, we have:*

$$J(\mathbf{x}_0, \varepsilon) - J(\mathbf{x}_0', \varepsilon) \leq L_J \|\mathbf{x}_0 - \mathbf{x}_0'\| \quad \forall \mathbf{x}_0, \mathbf{x}_0' \in \mathbb{X} \ .$$

Assumption 4.1. and connections as to whether the map $(\mathbf{x}_0, \varepsilon) \mapsto \mathbf{u}^\star_{0:T-1}$ is locally Lipschitz are concepts related to perturbation analysis of optimization problems (Rockafellar and Wets, 1998) and notions of *strong stability* of solutions, see e.g. Section 5 in Bonnans and Shapiro (1998). The middle panel in Figure 1 shows trajectories under our policy: if the dataset $\mathcal{D}$ has optimal transitions coming from (4) (i.e. satisfying $\mathbf{x}_t \in \mathbb{X}_{-\varepsilon}, \forall t \geq 1$), then our policy is guaranteed feasible for (1) (i.e. $\mathbf{x}_t \in \mathbb{X}$, $\forall t \geq 1$). Assumption 4.2. asks for the optimal cost-to-go to be Lipschitz continuous in the state variable. This is not overly restrictive, as shown below:

Proposition 1 (Sufficient conditions for Assumption 4.2. Lemma 3 in Buşoniu et al. (2018)) *Suppose Assumption 1 holds, the stage cost $c(\cdot, \cdot)$ is L_c-Lipschitz and $\gamma \max\{L_f, L_u\} < 1$. Then Assumption 4.2. holds with $L_J \leq \frac{L_c}{1 - \gamma \max\{L_f, L_u\}}$.*

1. This condition can be relaxed by enforcing that the system enters $\mathbb{X}_{-\varepsilon}$ after at most K steps, i.e. changing (4c) to $\mathbf{x}_t \in \mathbb{X}_{-\varepsilon} \forall t \geq K$, for some $K : 1 \leq K < T$. For the sake of clarity we focus on the case $K = 1$, but all our results are easily generalized to $K > 1$.

4. Nonparametric policy

In this work we propose a nonparametric policy based on *offline* (i.e. precomputed) solutions to (4). Optimal tuples $(\mathbf{x}_i, \mathbf{u}_i, \mathbf{J}_i)$ from the *conservative* problem (4) are stored in a dataset $\mathcal{D}$:

$$\mathcal{D} \triangleq \{(\mathbf{x}_i, \mathbf{u}_i, \mathbf{J}_i)\}_i \quad \text{where } \mathbf{u}_i = \pi^\star(\mathbf{x}_i),\ \mathbf{J}_i = J(\mathbf{x}_i, \varepsilon)\ . \tag{5}$$

We implement a policy that takes a "close" action in the dataset, as defined next.

Definition 1 (Nonparametric policy) *Given a dataset $\mathcal{D}$ as in (5) and a parameter $\lambda > 0$, define:*

$$\pi_{\mathcal{D}} : \mathbb{X} \to \mathbb{U} \quad \pi_{\mathcal{D}}(\mathbf{x}) = \mathbf{u}_\iota,\ \textit{where}\ \ \iota = \underset{1 \le i \le |\mathcal{D}|}{\arg\min} \left\{ \mathbf{J}_i + \lambda \cdot \|\mathbf{x} - \mathbf{x}_i\| \right\}\ .$$

A precise value for λ will be given later. This policy can be thought of as the one-nearest-neighbor regressor (Hastie et al., 2009, Ch. 13) based on the data $\{(\mathbf{x}_i, \mathbf{u}_i)\}_i$ from $\mathcal{D}$, with a regularization factor $\frac{1}{\lambda}\mathbf{J}_i$. This policy will *accelerate* MPC because inference will be done at a much lower latency (details deferred until Section 6), with only a modest hit on performance.

Remark 1 (Policy is built from the conservative problem) *We higlight that the dataset $\mathcal{D}$ defined above, and hence the policy, come from solving offline the* conservative *problem (4), and not the original one (1). Requiring the states $\mathbf{x}_t$ to be in $\mathbb{X}_{-\varepsilon}$ for all $t > 0$ will allow us to guarantee recursive feasibility of the nonparametric policy. Even though we solve a more conservative problem, we do so for the full horizon T (instead of using the lookahead horizon H).*

We are interested in ensuring that $\pi_{\mathcal{D}}$ achieves: *(i)* recursive feasibility and *(ii)* bounded suboptimality. We study these conditions next.

4.1. Guaranteeing recursive feasibility

The main idea to establish recursive feasibility of our policy is by leveraging the fact that we are solving the *conservative* problem over (4) $\mathbb{X}_{-\varepsilon}$, meaning optimal trajectories are separated from the boundary $\partial\mathbb{X}$. We establish this condition after the following definition.

Definition 2 (One-step feasibility) *$(\mathbf{x}, \mathbf{u})$ is one-step feasible with respect to (4) (respectively, to (1)) if $\mathbf{x} \in \mathbb{X}, \mathbf{u} \in \mathbb{U}$ and $f(\mathbf{x}, \mathbf{u}) \in \mathbb{X}_{-\varepsilon}$ (resp. $f(\mathbf{x}, \mathbf{u}) \in \mathbb{X}$).*

Proposition 2 (Local feasibility) *If $(\mathbf{x}, \mathbf{u})$ is one-step feasible w.r.t. (4) and $\mathbf{x}' = f(\mathbf{x}, \mathbf{u})$, then $(\mathbf{x}_0, \mathbf{u})$ is one-step feasible w.r.t. (1) for all $\mathbf{x}_0 \in \mathbb{B}\big(\mathbf{x}, r(\mathbf{x})\big) \cap \mathbb{X}$, where:*

$$r(\mathbf{x}) \triangleq \frac{\operatorname{dist}\left(f(\mathbf{x}, \mathbf{u}), \partial\mathbb{X}\right)}{L_f} \ge \frac{\varepsilon}{L_f} \tag{6}$$

The proof of the proposition above is in Appendix B.1, and relies on the Lipschitzness of $f(\cdot, \cdot)$ (Assumption 1). The inequality in (6) comes from the assumption that the conservative problem (4) is feasible over $\mathbf{x} \in \mathbb{X}$. This proposition can be visualized in the middle panel of Figure 1: trajectories under the same $\mathbf{u}$ cannot go too far apart in one step. Conditions for recursive feasibility over the whole domain follow naturally.

Proposition 3 (Recursive feasibility for our policy) *If any of the following conditions hold, then $\pi_{\mathcal{D}}$ is recursively feasible for (1) over $\mathbb{X}$:*

1. *The states $\{\mathbf{x}_i\}_i$ in $\mathcal{D}$ forms an $\frac{\varepsilon}{L_f}$-cover of $\mathbb{X}$*[2].
2. *$\bigcup_{i=1}^{|\mathcal{D}|} \mathbb{B}(\mathbf{x}_i, r(\mathbf{x}_i)) \supseteq \mathbb{X}$, where the radii $r(\mathbf{x}_i)$ are defined in Proposition 2.*

Note that the former condition may be overly conservative, since it considers the smallest possible feasibility radius for any point in the dataset (see transition $\mathbf{y} \to \mathbf{y}'$ in Figure 1). Since our policy always picks actions in the dataset $\mathcal{D}$, the constraint $\mathbf{u}_t \in \mathbb{U}$ is guaranteed by design.

4.2. Bounded suboptimality

We now turn to bounding the optimality gap of policy $\pi_{\mathcal{D}}$. The key idea developed in this section is that, with a proper choice of regularization λ (see Definition 1), our policy's cost-to-go can be lower bounded. First, recall that under Assumption 4.2. the cost-to-go for the perturbed problem is L_J-Lipschitz, meaning:

$$J(\mathbf{x}_0, \varepsilon) \leq J(\mathbf{x}_0', \varepsilon) + L_J \|\mathbf{x}_0 - \mathbf{x}_0'\| \ .$$

We use this to establish a global upper bound for $J(\cdot, \varepsilon)$, based on the data in $\mathcal{D}$.

Definition 3 (Nonparametric upper & lower bounds on J) *Let $\mathcal{D}$ be the dataset in (5). For any $\lambda > 0$ define:*

$$J_{\text{ub}}^{\lambda} : \mathbb{X} \to \mathbb{R} : \quad J_{\text{ub}}^{\lambda}(\mathbf{x}) \triangleq \min_{1 \leq i \leq |\mathcal{D}|} \left\{ \mathbf{J}_i + \lambda \|\mathbf{x} - \mathbf{x}_i\| \right\}$$
$$J_{\text{lb}}^{\lambda} : \mathbb{X} \to \mathbb{R} : \quad J_{\text{lb}}^{\lambda}(\mathbf{x}) \triangleq \max_{1 \leq i \leq |\mathcal{D}|} \left\{ \mathbf{J}_i - \lambda \|\mathbf{x} - \mathbf{x}_i\| \right\}$$

It follows from the Lipschitz condition on $J(\cdot, \varepsilon)$ that $J_{\text{lb}}^{\lambda}(\mathbf{x}) \leq J(\mathbf{x}, \varepsilon) \leq J_{\text{ub}}^{\lambda}(\mathbf{x})$ for all $\mathbf{x} \in \mathbb{X}$, as long as $\lambda \geq L_J$. Our key finding is that, under appropiate choice of λ, our policy is better than $J_{\text{ub}}^{\lambda}(\cdot)$, that is: $J^{\pi}(\mathbf{x}) \leq J_{\text{ub}}^{\lambda}(\mathbf{x})$.

Theorem 1 (Policy evaluation inequality) *Assume we are in any of the conditions of Proposition 3, $\gamma L_f < 1$, and that the dataset $\mathcal{D}$ contains* trajectories*, i.e., for any $(\mathbf{x}_i, \mathbf{u}_i) \in \mathcal{D}$, there exists $j \leq |\mathcal{D}| : \mathbf{x}_j = f(\mathbf{x}_i, \mathbf{u}_i) \in \mathcal{D}$. Then:*

$$\lambda \geq \left(\frac{1 + \gamma L_f}{1 - \gamma L_f} \right) L_J \implies J^{\pi}(\mathbf{x}) \leq J_{\text{ub}}^{\lambda}(\mathbf{x}) \quad \forall \mathbf{x} \in \mathbb{X}. \tag{7}$$

Proof The full proof is in Appendix B.2; here we provide a sketch. The key idea is establishing that $\left(\mathcal{T}^{\pi} J_{\text{ub}}^{\lambda}\right)(\mathbf{x}) \leq J_{\text{ub}}^{\lambda}(\mathbf{x}) \quad \forall \mathbf{x} \in \mathbb{X}$, where $\mathcal{T}^{\pi}$ is the Bellman operator under policy π (Bertsekas et al., 2011, Ch. 1). Then, the monotonicity of this operator ($J_1 \preceq J_2 \implies T^{\pi} J_1 \preceq T^{\pi} J_2$), together with the fact that its unique fixed point is J^{π} yields the thesis. ■

The preceeding theorem establishes that our policy is at least as good as a conservative upper bound on the optimal cost-to-go. We wish to emphasize that $J_{\text{ub}}^{\lambda}(\cdot)$ is a data-dependent bound that *improves*

2. A collection of points $\{w_i\}_{i=1,2,\ldots,|\mathcal{D}|}$ forms an r-cover of a normed space $(\mathbb{W}, \|\cdot\|)$ if $\mathbb{W} \subseteq \bigcup_i \mathbb{B}(w_i, r)$.

Algorithm 1: Data collector

Input: Conservative threshold $\varepsilon > 0$. Budget N.

Initialize: $\mathcal{D} = \emptyset$

for $i = 1, \dots, N$ **do**

  Sample $\mathbf{x}_i \sim \text{Uniform}(\mathbb{X})$.

  Get trajectory $\tau_i = \{(\mathbf{x}_k, \mathbf{u}_k, \mathbf{J}_k)\}_{k=i}^{T+i-1}$ by solving (4) from $\mathbf{x}_i$, add each transition to $\mathcal{D}$.

end

Output: Nonparametric policy $\pi_{\mathcal{D}}$ with guarantees (Prop. 4)

(i.e. becomes smaller) with more data, which is expected to improve the performance of policy J^{π}. Our work builds on Castellano et al. (2025), where the authors study unconstrained optimization problems in the context of Reinforcement Learning, which we adapt here to constrained settings. We close this section by establishing performance guarantees for policy $\pi_{\mathcal{D}}$.

Theorem 2 (Performance guarantees) *Let $\beta > 0$ and $0 < \eta \ll 1$. Assume policy $\pi_{\mathcal{D}}$ is recursively feasible and that the conditions of Theorem 1 are satisfied. If the dataset $\mathcal{D}$ has* sufficient coverage*, in the sense that:*

$$\forall \mathbf{x} \in \mathbb{X} \;\; \exists\, \mathbf{x}_i \in \mathcal{D} : \|\mathbf{x} - \mathbf{x}_i\| \leq \frac{\beta}{(2+\beta)\lambda} \left(\mathbf{J}_i + \eta\right), \tag{8}$$

then:

$$\sup_{\mathbf{x} \in \mathbb{X}} \frac{J^{\pi}(\mathbf{x}) - J(\mathbf{x}, \varepsilon)}{J(\mathbf{x}, \varepsilon) + \eta} \leq \beta \,. \tag{9}$$

Moreover, if $\eta > L\varepsilon$:

$$\sup_{\mathbf{x} \in \mathbb{X}} \frac{J^{\pi}(\mathbf{x}) - J(\mathbf{x})}{J(\mathbf{x}) + \eta} \leq \frac{\beta\eta + L\varepsilon}{\eta - L\varepsilon} \,. \tag{10}$$

The proof is in Appendix B.3. We highlight that (9) bounds the *relative suboptimality* of our policy with respect to the optimal solution of the *conservative* problem (4). By contrast, (10) quantifies the *relative gap* to the optimal solution of the *original* problem (1). The parameter η is introduced to keep the denominators bounded away from zero, so that the relative tolerance β is well-defined and attainable even in neighborhoods where $J(\mathbf{x}) = 0$ or $J(\mathbf{x}, \varepsilon) = 0$. Recall all costs in our original formulation (1a) are non-negative, so we always have $J(\mathbf{x}, \varepsilon) \geq J(\mathbf{x}) \geq 0$.

5. Algorithms

We now present two algorithms to drive the data-collection process for dataset $\mathcal{D}$. The first one (Algorithm 1) assumes initial conditions can be sampled uniformly from $\mathbb{X}$. Offline MPC is then run from each sampled point to give an optimal trajectory for (4). We establish PAC bounds for this algorithm next.

Proposition 4 (Sample complexity for Algorithm 1) *Let β be a small positive constant, and define $r \triangleq \min\left\{\frac{(\beta(\eta - L\varepsilon) - L\varepsilon)\eta}{2\lambda(2\eta + \beta(\eta - L\varepsilon) - L\varepsilon)}, \frac{\varepsilon}{2L_f}\right\}$. With probability at least $1 - \delta$, Algorithm 1 outputs both a recursively feasible and β-optimal policy over $\mathbb{X}$ after at most:*

$$\mathcal{O}\left(\mathrm{N}_{cover}\left(\mathbb{X}; r\right) \log \mathrm{N}_{cover}\left(\mathbb{X}; r\right) \log \frac{1}{\delta}\right)$$

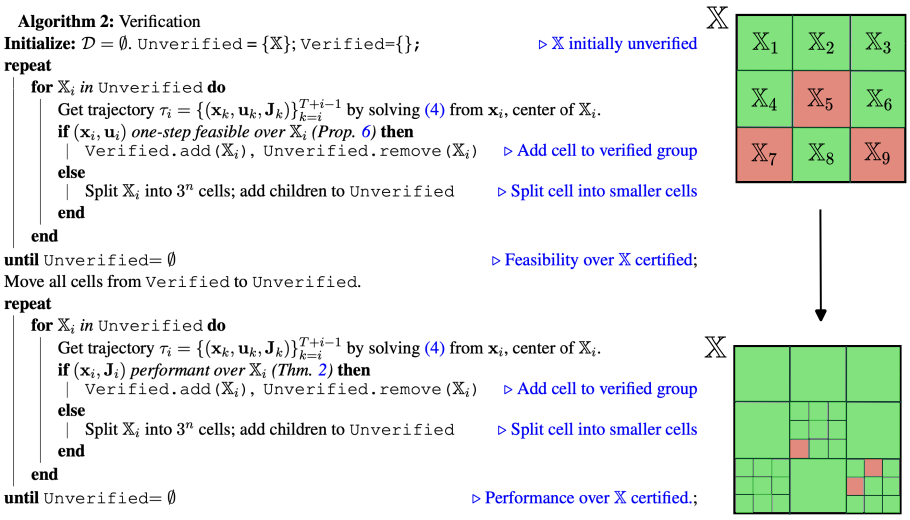

Figure 2: Algorithm 2 and the visualization of the cell verification/splitting method. Each cell $\mathbb{X}_1, \ldots, \mathbb{X}_9$ is tested against two criteria—(i) one-step feasibility and (ii) performance. Cells that pass are shown in green and kept in `Verified`; those that fail are shown in red and are split into 3^n child cells that are yet to be verified. The children are then re-verified using the same criteria and are either accepted (green) and moved to `Verified`, or split again (red), proceeding sequentially.

iterations, where $\mathrm{N}_{cover}(\mathbb{X}; r)$ *is the covering number*[3] *of* $\mathbb{X}$ *with balls of radius* r*, and "*β*-optimal policy" means* $\sup_{\mathbf{x}\in\mathbb{X}} \frac{J^\pi(\mathbf{x}) - J(\mathbf{x})}{J(\mathbf{x})+\eta} \leq \beta$*, i.e., the gap with respect to the original problem (1).*

See Appendix B.4 for a detailed proof of Proposition 4. Algorithm 1 enjoys the guarantees above but may be inefficient due to stochastic sampling. We present another algorithm that recursively partitions $\mathbb{X}$ into disjoint cells, and verifies both feasibility (Prop. 6) and desired performance (Theorem 2) for each cell, splitting each cell until verification is achieved. We highlight this procedure in Algorithm 2, and relegate a detailed description (Algorithm 2) to Appendix B.5. In what follows we assume $\mathbb{X}$ is a hypercube containing the origin and $\|\cdot\|_{\mathbb{X}} = \|\cdot\|_\infty$. This can easily be generalized to more complex shapes of $\mathbb{X}$ by taking inner covers of $\mathbb{X}$ with L_∞-balls. See Figure 2 for a visualization of the algorithm.

6. Experiments

The purpose of the following numerical simulations are two-fold. First, we want to establish a trade-off between the speedup of our method and its performance gap. As one would expect, more data will improve performance but at the cost of higher latency. Second, we show Algorithm 2 in action, recursively verifying a domain $\mathbb{X}$ by the cell-splitting method.

On "accelerating" MPC: It should be noted that one of the potential bottlenecks of implementing our policy $\pi_{\mathcal{D}}$ is that inference requires querying distances $\|\mathbf{x} - \mathbf{x}_i\|$ of a test point $\mathbf{x}$ to each point $\mathbf{x}_i$ in the dataset. We use `FAISS` (Johnson et al., 2019; Douze et al., 2024), a GPU-enabled library for fast similarity search, allowing us to enjoy a substantial speed-up with respect to standard MPC.

3. Formally defined as $\min\left\{n > 0 : \exists \mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{X} : \bigcup_{i\leq n} \mathbb{B}(\mathbf{x}_i, r) \supseteq \mathbb{X}\right\}$

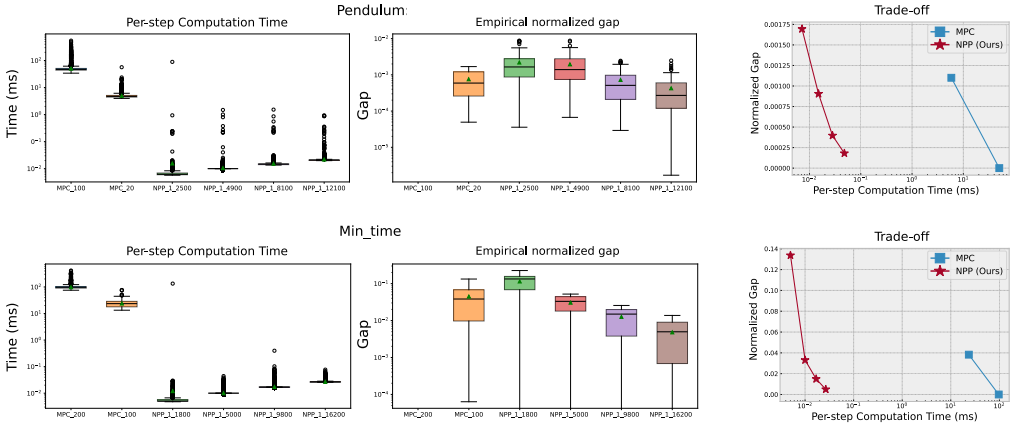

Figure 3: Statistics for the inverted pendulum (top) and minimum time problem (bottom) over 100 trajectories. Left: Per-step latency (in ms) for each controller. Our controllers (`NN_1_XXXX`) are ordered left to right from smallest to largest dataset $\mathcal{D}$. Middle: Distribution of the relative optimality gap. Boxes correspond to the interquartile range $(25\% - 75\%)$, black line shows the median and the green arrow corresponds to the mean. Right: Trade-off between computation time and relative gap for our method (red $\star$'s) and MPC (blue $\square$'s). Our method is substantially faster than MPC and, with sufficient data, outperforms MPC with shorter lookahead horizons.

Trade-offs between latency and performance: We study trade-offs between latency (small computation time) and performance (small optimality gap) on two different benchmarks: an **inverted pendulum**, in which we wish to stabilize the pendulum near the unstable equilibrium, and a **minimum time**, unstable LTI system that we wish to drive to the origin with penalized control effort. Due to space limitations, we relegate the details of these environments to Appendix C. For these two environments, we consider different datasets $\mathcal{D}$ obtained by uniformly partitioning the state space into a grid, with $g \in \{3, 5, 7, 9, 11\}$ grid points per dimension. Then, (4) is solved for each point in the grid, and horizon $T = 100$ trajectories are added to the dataset. Then, the performance of both MPC controllers (with varying lookahead horizon $H \leq T$) and our nonparamaetric policies (labelled `NPP`) are evaluated on $M = 100$ trajectories from random initial conditions. These trajectories are used to build the plots in Figure 3, where we show the results for the pendulum (top) and minimum time problem (bottom). The boxplots in the left and middle panels show, in color, the interquartile range (25% - 75%) of the empirical distribution over the $M = 100$ trajectories; median values are shown as a solid black horizontal line, means are shown in green, and the whiskers extend to 1.5 times the interquartile range. Outliers are shown as black circles. The left panel shows the time taken to get one control action (in ms), the middle one shows the empirical normalized gap $\frac{J^\pi(\mathbf{x}) - J(\mathbf{x})}{J(\mathbf{x})}$ for the different controllers across the M trajectories. The right panel shows the trade-off between the normalized gap and the time taken to get controls, where each marker corresponds to the median value for that particular controller. Our nonparametric policy strikes a good balance between good performance (low gap) with remarkably lower computation time (many orders of magnitude faster).

Verification We test Algorithm 2 on a discrete LQR problem. Details on the setup and on the hyperparameters of our method are in Appendix C. Figure 4 shows Algorithm 2 in action: it recur-

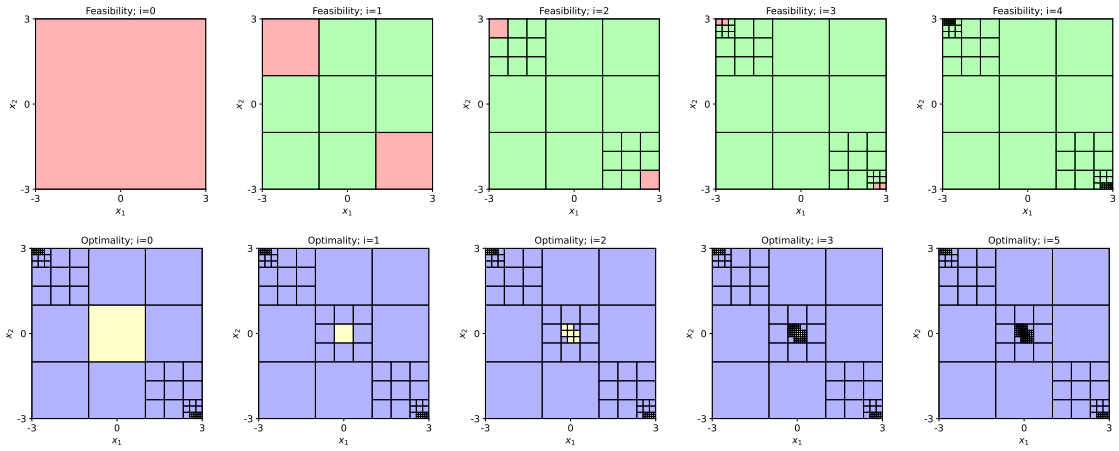

Figure 4: Algorithm 2 in action: feasibility (top row) and optimality (bottom row) certificates for the LQR problem. Iterations are ordered from left to right. Top: for each figure, we show in red the cells that don't satisfy the feasibility condition (Prop. 6). The algorithm recursively splits each cell and runs trajectories from the center points. Verified cells are shown in green. Bottom: After all cells have been deemed feasible, the algorithm verifies the optimality gap (Theorem 2). Guaranteed suboptimal cells are shown in blue. At termination (bottom right panel) the algorithm has verified the whole state space $\mathbb{X}$.

sively partitions the state space into smaller balls until feasibility can be verified (top row of Fig. 4), then it focuses on guaranteeing optimality for each cell—with more splitting if needed (bottom row of Fig. 4). Note that feasibility is harder to verify near the boundary of $\mathbb{X}$ (requiring more splitting). This is to be expected from our guarantees on a feasibility radius (Prop. 6). A target gap, however, is harder to verify near the origin. This is explained since $J(\mathbf{x}) \approx 0$ near the origin, and our target gap (9) is relative.

7. Conclusions & Future work

We proposed a nonparametric, data-driven scheme to accelerate MPC by reusing offline-computed trajectories: online control queries pick stored controls by trading off recorded cost-to-go and state proximity. Under mild Lipschitz and coverage assumptions the policy enjoys recursive feasibility and has explicit performance bounds; with sufficient dataset coverage the relative suboptimality can be driven arbitrarily small. Inference is extremely fast and we achieve a few orders of magnitude of speed-up with respect to standard MPC. Future work includes testing and scalable implementations for high-dimensional systems and embedded control applications.

Acknowledgments

A.C. is grateful to Pedro Izquierdo for insightful discussions and suggestions during the first stages of this manuscript.

References

Karun Adusumilli. Neyman allocation is minimax optimal for best arm identification with two arms. *arXiv preprint arXiv:2204.05527*, 2022.

Alessandro Alessio and Alberto Bemporad. A survey on explicit model predictive control. In *Nonlinear model predictive control: towards new challenging applications*, pages 345–369. Springer, 2009.

Richard Bellman. The theory of dynamic programming. *Bulletin of the American Mathematical Society*, 60(6):503–515, 1954.

Alberto Bemporad, Manfred Morari, Vivek Dua, and Efstratios N Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20, 2002.

Dimitri Bertsekas. *Dynamic programming and optimal control: Volume I*, volume 4. Athena scientific, 2012.

Dimitri P Bertsekas et al. Dynamic programming and optimal control 3rd edition, volume ii. *Belmont, MA: Athena Scientific*, 1, 2011.

Franco Blanchini. Set invariance in control. *Automatica*, 35(11):1747–1767, 1999.

Franco Blanchini, Stefano Miani, et al. *Set-theoretic methods in control*, volume 78. Springer, 2008.

J Frédéric Bonnans and Alexander Shapiro. Optimization problems with perturbations: A guided tour. *SIAM review*, 40(2):228–264, 1998.

Lucian Buşoniu, Előd Páll, and Rémi Munos. Continuous-action planning for discounted infinite-horizon nonlinear optimal control with lipschitz values. *Automatica*, 92:100–108, 2018.

Agustin Castellano, Sohrab Rezaei, Jared Markowitz, and Enrique Mallada. Nonparametric policy improvement in continuous action spaces via expert demonstrations. In *Reinforcement Learning Conference*, 2025. URL https://openreview.net/pdf?id=aj9jJvdFDR.

Hong Chen and Frank Allgöwer. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. *Automatica*, 34(10):1205–1217, 1998.

Steven Chen, Kelsey Saulnier, Nikolay Atanasov, Daniel D Lee, Vijay Kumar, George J Pappas, and Manfred Morari. Approximating explicit model predictive control using constrained neural networks. In *2018 Annual American control conference (ACC)*, pages 1520–1527. IEEE, 2018.

CR Cutler and BL Ramaker. Dynamic matrix control. In *A computer control algorithm. In joint automatic control conference*, volume 17, page 72, 1980.

Stefano Di Cairano and Ilya V Kolmanovsky. Real-time optimization and model predictive control for aerospace and automotive applications. In *2018 annual American control conference (ACC)*, pages 2392–2409. IEEE, 2018.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The faiss library. *arXiv*, 2024.

Ján Drgoňa, Javier Arroyo, Iago Cupeiro Figueroa, David Blum, Krzysztof Arendt, Donghun Kim, Enric Perarnau Ollé, Juraj Oravec, Michael Wetter, Draguna L Vrabie, et al. All you need to know about model predictive control for buildings. *Annual reviews in control*, 50:190–232, 2020.

Riccardo Fogliato, Pratik Patil, Mathew Monfort, and Pietro Perona. A framework for efficient model evaluation through stratification, sampling, and estimation. In *European Conference on Computer Vision*, pages 140–158. Springer, 2024.

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, 2015.

Carlos E Garcia, David M Prett, and Manfred Morari. Model predictive control: Theory and practice—a survey. *Automatica*, 25(3):335–348, 1989.

Aurélien Garivier and Emilie Kaufmann. Optimal best arm identification with fixed confidence. In *Conference on Learning Theory*, pages 998–1027. PMLR, 2016.

Lars Grüne, Jürgen Pannek, Martin Seehafer, and Karl Worthmann. Analysis of unconstrained nonlinear mpc schemes with time varying control horizon. *SIAM Journal on Control and Optimization*, 48(8):4938–4962, 2010.

Trevor Hastie, Robert Tibshirani, Jerome Friedman, et al. The elements of statistical learning, 2009.

Davor Hrovat, Stefano Di Cairano, H Eric Tseng, and Ilya V Kolmanovsky. The development of model predictive control in automotive industry: A survey. In *2012 IEEE International Conference on Control Applications*, pages 295–302. IEEE, 2012.

Ali Jadbabaie, Jie Yu, and John Hauser. Unconstrained receding-horizon control of nonlinear systems. *IEEE Transactions on Automatic Control*, 46(5):776–783, 2002.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2019.

Ilja Kuzborskij and Nicolo Cesa-Bianchi. Locally-adaptive nonparametric online learning. *Advances in Neural Information Processing Systems*, 33:1679–1689, 2020.

Johan Löfberg. Oops! i cannot do it again: Testing for recursive feasibility in mpc. *Automatica*, 48 (3):550–555, 2012.

Emilio Tanowe Maddalena, CG da S Moraes, Gierri Waltrich, and Colin N Jones. A neural network architecture to learn explicit mpc controllers from data. *IFAC-PapersOnLine*, 53(2):11362–11367, 2020.

Jacob Mattingley, Yang Wang, and Stephen Boyd. Receding horizon control. *IEEE Control Systems Magazine*, 31(3):52–65, 2011.

David Q Mayne. Model predictive control: Recent developments and future promise. *Automatica*, 50(12):2967–2986, 2014.

David Q Mayne, James B Rawlings, Christopher V Rao, and Pierre OM Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

Rémi Munos. Optimistic optimization of a deterministic function without the knowledge of its smoothness. *Advances in neural information processing systems*, 24, 2011.

James B Rawlings and Kenneth R Muske. The stability of constrained receding horizon control. *IEEE transactions on automatic control*, 38(10):1512–1516, 2002.

Marcus Reble and Frank Allgöwer. Unconstrained model predictive control and suboptimality estimates for nonlinear continuous-time systems. *Automatica*, 48(8):1812–1817, 2012.

Jacques Richalet, André Rault, JL Testud, and J Papon. Model predictive heuristic control. *Automatica (journal of IFAC)*, 14(5):413–428, 1978.

R Tyrrell Rockafellar and Roger JB Wets. *Variational analysis*. Springer, 1998.

Roy Siegelmann, Yue Shen, Fernando Paganini, and Enrique Mallada. Stability analysis and data-driven verification via recurrent lyapunov functions. *IEEE Transactions on Automatic Control*, 7, 2025.

Sean Sinclair, Tianyu Wang, Gauri Jain, Siddhartha Banerjee, and Christina Yu. Adaptive discretization for model-based reinforcement learning. *Advances in Neural Information Processing Systems*, 33:3858–3871, 2020.

Karl Worthmann. Estimates of the prediction horizon length in mpc: A numerical case study. *IFAC Proceedings Volumes*, 45(17):232–237, 2012.

Appendix A.

Proposition 5 (Bounding trajectories) *For all* $\mathbf{x} \in \mathbb{X}$ *and for all* $\mathbf{u} \in \mathbb{U}$ *we have:*

$$\|f(\mathbf{x}, \mathbf{u})\| \leq L_f \|\mathbf{x}\| + L_u \|\mathbf{u}\| \ . \tag{11}$$

Remark 2 (On the need for global Lipschitz constants) *For the previous result to make sense for any* $\mathbf{x} \in \mathbb{X}, \mathbf{u} \in \mathbb{U}$ L_f *and* L_u *must be global Lipschitz constants, since we are comparing* $(\mathbf{x}, \mathbf{u})$ *to* $(\mathbf{0}, \mathbf{0})$.

Bounding trajectories

We use $\mathbf{x}_t = \phi(\mathbf{x}_0, \mathbf{u}_{0:t-1})$ to denote the solution at time t starting from $\mathbf{x}_0$, under control law $\mathbf{u}_{0:t-1} = [\mathbf{u}_0, \mathbf{u}_1, \dots \mathbf{u}_{t-1}]$. Similarly, let $\mathbf{x}'_t = \phi(\mathbf{x}'_0, \mathbf{u}'_{0:t-1})$. We have the following result:

$$\left\|\mathbf{x}_t - \mathbf{x}'_t\right\| \leq L_f^k \cdot \left\|\mathbf{x}_0 - \mathbf{x}'_0\right\| + L_u \sum_{\ell=0}^{t-1} L_f^{t-1-\ell} \left\|\mathbf{u}_\ell - \mathbf{u}'_\ell\right\| \quad \forall t \geq 0 \ . \tag{12}$$

In particular, for two different states $\mathbf{x}_0, \mathbf{x}'_0$ under the same control $\mathbf{u}_0$:

$$\left\|\mathbf{x}_1 - \mathbf{x}'_1\right\| \leq L_f \left\|\mathbf{x}_0 - \mathbf{x}'_0\right\| \ . \tag{13}$$

A.1. Certifying recursive feasibility

Let $(\mathbf{x}, \mathbf{u}, \mathbf{x}')$ be a triplet with $\mathbf{x}' = f(\mathbf{x}, \mathbf{u})$. We want to study whether applying the same control $\mathbf{u}$ for a different state $\mathbf{x}_0$ (close to $\mathbf{x}$) is feasible.
We define the *radius* of the feasible set $\mathbb{X}$ as:

$$R \triangleq \sup \{r \geq 0 : \mathbb{B}_{\mathbb{X}}(\mathbf{0}, r) \subseteq \mathbb{X}\} \ . \tag{14}$$

We say a tuple $(\mathbf{x}, \mathbf{u})$ is *one-step feasible* if $\mathbf{x} \in \mathbb{X}, \mathbf{u} \in \mathbb{U}, f(\mathbf{x}, \mathbf{u}) \in \mathbb{X}$.

Proposition 6 (One-step feasibility)

1. *If* $(\mathbf{x}, \mathbf{u})$ *is feasible, then* $(\mathbf{x}_0, \mathbf{u})$ *is feasible for all* $\mathbf{x}_0 \in \mathbb{B}\big(\mathbf{x}, r(\mathbf{x}, \mathbf{u})\big)$, *where:*

$$r(\mathbf{x}, \mathbf{u}) \triangleq \max \left\{0, \frac{R - L_u \|\mathbf{u}\|}{L_f} - \|\mathbf{x}\|\right\} \tag{15}$$

2. *If* $(\mathbf{x}, \mathbf{u})$ *is feasible and* $\mathbf{x}' = f(\mathbf{x}, \mathbf{u})$, *then* $(\mathbf{x}_0, \mathbf{u})$ *is feasible for all* $\mathbf{x}_0 \in \mathbb{B}\big(\mathbf{x}, r(\mathbf{x}')\big)$, *where:*

$$r(\mathbf{x}') \triangleq \frac{R - \|\mathbf{x}'\|}{L_f} \leq \frac{\operatorname{dist}(\mathbf{x}', \partial\mathbb{X})}{L_f} \tag{16}$$

Appendix B. Proofs

B.1. Proposition 6

Proof

1. Let $\mathbf{x}' = f(\mathbf{x}, \mathbf{u})$. By virtue of (11):

$$\|\mathbf{x}'\| \leq L_f \|\mathbf{x}\| + L_u \|\mathbf{u}\| \tag{17}$$

Let $\mathbf{x}_0 \in \mathbb{X} : \|\mathbf{x}_0 - \mathbf{x}\| \leq r$, and $\mathbf{x}_0' = f(\mathbf{x}_0, \mathbf{u})$ be the successor state under the same $\mathbf{u}$. We have:

$$\|\mathbf{x}_0'\| - \|\mathbf{x}'\| \leq \|\mathbf{x}_0' - \mathbf{x}'\| \leq L_f r \implies \tag{18}$$

$$\|\mathbf{x}_0'\| \leq \|\mathbf{x}'\| + L_f r \leq L_f \left(\|\mathbf{x}\| + r\right) + L_u \|\mathbf{u}\|, \tag{19}$$

where the first inequality in (18) follows from the reverse triangle inequality, and the second one from (12) (for two successor states under same control $\mathbf{u}$). In (19) we rearrange terms and use (17).

Imposing the right hand side of (19) be smaller than R:

$$L_f \left(\|\mathbf{x}\| + r\right) + L_u \|\mathbf{u}\| \leq R \implies r \leq \frac{R - L_u \|\mathbf{u}\|}{L_f} - \|\mathbf{x}\| \,, \tag{20}$$

as desired.

2. Re-using the first inequality in (19), and imposing that it be upper bounded by R:

$$\|\mathbf{x}_0'\| \leq \|\mathbf{x}'\| + L_f r \leq R \implies r \leq \frac{R - \|\mathbf{x}'\|}{L_f} \,. \tag{21}$$

For the remaining inequality, note:

$$\operatorname{dist}\left(\mathbf{x}', \partial\mathbb{X}\right) \geq R - \|\mathbf{x}'\| \quad \forall \mathbf{x}' \in \mathbb{X} \,. \tag{22}$$

■

B.2. Theorem 1

Proof Notice that, by assumption, we are in the conditions of Proposition 3. This implies policy $\pi_{\mathcal{D}}$ is feasible, and hence:

$$J^{\pi}(\mathbf{x}) < +\infty \quad \forall \mathbf{x} \in \mathbb{X}.$$

Let $\mathcal{T}^{\pi} : \mathcal{J} \to \mathcal{J}$ be the Bellman operator (Bertsekas et al., 2011, Ch. 1) of policy π, mapping costs-to-go $J \in \mathcal{J}$ onto $\mathcal{J}$, defined by:

$$\left(T^{\pi} J\right)(\mathbf{x}) = c(\mathbf{x}, \pi(\mathbf{x})) + \gamma \cdot J\big(f(\mathbf{x}, \pi(\mathbf{x}))\big)$$

The following are two well-known facts of $\mathcal{T}^{\pi}$(Bertsekas et al., 2011, Lemma 1.1.1; Prop 1.2.1):

1. For any policy π, $\mathcal{T}^\pi$ is monotone, i.e.

$$J_1(\mathbf{x}) \leq J_2(\mathbf{x}) \quad \forall \mathbf{x} \implies \left(\mathcal{T}^\pi J_1\right)(\mathbf{x}) \leq \left(\mathcal{T}^\pi J_2\right)(\mathbf{x}) \quad \forall \mathbf{x}$$

2. J^π is the unique fixed point of $\mathcal{T}^\pi$:

$$\lim_{k\to\infty} \overbrace{\left(\mathcal{T}^\pi \circ \mathcal{T}^\pi \circ \dots \mathcal{T}^\pi\right)}^{k \text{ times}} J = J^\pi \quad \forall J \in \mathcal{J}.$$

The combination of these two facts leads to the following lemma:

Lemma 1 *If $J : \mathbb{X} \to \mathbb{R}$ satisfies:*

$$\left(T^\pi J\right)(\mathbf{x}) \leq J(\mathbf{x}) \quad \forall \mathbf{x} \in \mathbb{X},$$

then:

$$J^\pi(\mathbf{x}) \leq J(\mathbf{x}) \quad \forall \mathbf{x} \in \mathbb{X}\,.$$

Then, to prove Theorem 1, all that remains is to show J^λ_{ub} satisfies the hypothesis of the lemma above.
Fix $\mathbf{x} \in \mathbb{X}$ and let

$$\mathbf{u}_i = \pi(\mathbf{x}),$$

where i is the solution of:

$$\arg\min_{1\leq i\leq |\mathcal{D}|} \left\{\mathbf{J}_i + \lambda \cdot \|\mathbf{x} - \mathbf{x}_i\|\right\}\,.$$

Define as well:

$$\mathbf{x}'_i = f(\mathbf{x}_i, \mathbf{u}_i), \quad \mathbf{x}' = f(\mathbf{x}, \mathbf{u}_i)\,.$$

Then, the following string of (in)equalities hold, as explained below:

$$\begin{aligned}
\mathcal{T}^\pi J^\lambda_{\text{ub}}(s) - J^\lambda_{\text{ub}}(s) &= c(\mathbf{x}, \mathbf{u}_i) + \gamma J^\lambda_{\text{ub}}(\mathbf{x}') - J^\lambda_{\text{ub}}(\mathbf{x}) && (23)\\
&= J(\mathbf{x}, \mathbf{u}_i) - \gamma J(\mathbf{x}') + \gamma J^\lambda_{\text{ub}}(\mathbf{x}') - J^\lambda_{\text{ub}}(\mathbf{x}) && (24)\\
&= J(\mathbf{x}, \mathbf{u}_i) - \gamma J(\mathbf{x}') + \gamma J^\lambda_{\text{ub}}(\mathbf{x}') - \mathbf{J}_i - \lambda\|\mathbf{x} - \mathbf{x}_i\| && (25)\\
&\leq \mathbf{J}_i + L_J\|\mathbf{x} - \mathbf{x}_i\| + \gamma\Big(J^\lambda_{\text{ub}}(\mathbf{x}') - J(\mathbf{x}')\Big) - \mathbf{J}_i - \lambda\|\mathbf{x} - \mathbf{x}_i\| && (26)\\
&\leq (L_J - \lambda)\|\mathbf{x} - \mathbf{x}_i\| + \gamma\big(\mathbf{J}_j + \lambda\|\mathbf{x}' - \mathbf{x}_j\| - \mathbf{J}_j + L_J\|\mathbf{x}' - \mathbf{x}_j\|\big) && (27)\\
&= (L_J - \lambda)\|\mathbf{x} - \mathbf{x}_i\| + \gamma(\lambda + L_J)\|\mathbf{x}' - \mathbf{x}_j\| && (28)\\
&\leq (L_J - \lambda)\|\mathbf{x} - \mathbf{x}_i\| + \gamma(\lambda + L_J)L_f\|\mathbf{x} - \mathbf{x}_i\| && (29)\\
&\leq 0 \iff && (30)\\
&\iff (L_J - \lambda) + \gamma(\lambda + L_J)L_f \leq 0 && (31)\\
&\iff L_J + \gamma L_J L_f \leq \lambda(1 - \gamma L_f) && (32)\\
&\iff \lambda \geq \frac{1 + \gamma L_f}{1 - \gamma L_f} L_J\,, && (33)
\end{aligned}$$

completing the proof. ■

B.3. Theorem 2

Proof Note that (9) is equivalent to

$$(1+\beta)\left(J(\mathbf{x},\varepsilon)+\eta\right) \;\geq\; J^{\pi}(\mathbf{x})+\eta. \tag{34}$$

Fix any $\mathbf{x}\in\mathbb{X}$. By the Definition 3,

$$(1+\beta)\left(J(\mathbf{x},\varepsilon)+\eta\right) \;\geq\; (1+\beta)(\max_{1\leq i\leq|\mathcal{D}|}\left\{\mathbf{J}_i+\eta-\lambda\|\mathbf{x}-\mathbf{x}_i\|\right\}). \tag{35}$$

Letting $\mathbf{x}_i \in \arg\max_{1\leq i\leq|\mathcal{D}|}\left\{\mathbf{J}_i+\eta-\lambda\|\mathbf{x}-\mathbf{x}_i\|\right\}$. Combining with the $J^{\lambda}_{\mathrm{ub}}$ in Definition 3, a sufficient condition for (34) is,

$$\begin{aligned}(1+\beta)(J(\mathbf{x},\varepsilon)+\eta) &\geq (1+\beta)(J^{\lambda}_{\mathrm{lb}}(\mathbf{x})+\eta)\\ &= (1+\beta)(\max_{1\leq i\leq|\mathcal{D}|}\left\{\mathbf{J}_i+\eta-\lambda\|\mathbf{x}-\mathbf{x}_i\|\right\})\\ &= (1+\beta)\left(\mathbf{J}_i+\eta-\lambda\|\mathbf{x}-\mathbf{x}_i\|\right),\end{aligned}$$

and, by taking the same $\mathbf{x}_i$ to get the upper bound of $J^{\pi}(\mathbf{x})+\eta$,

$$\begin{aligned}J^{\pi}(\mathbf{x})+\eta &\leq J^{\lambda}_{\mathrm{ub}}(\mathbf{x})+\eta\\ &= \min_{1\leq j\leq|\mathcal{D}|}\left\{\mathbf{J}_j+\eta+\lambda\|\mathbf{x}-\mathbf{x}_j\|\right\}\\ &\leq \mathbf{J}_i+\eta+\lambda\|\mathbf{x}-\mathbf{x}_i\|.\end{aligned}$$

So, we have the sufficient condition is,

$$\begin{aligned}(1+\beta)\left(\mathbf{J}_i+\eta-\lambda\|\mathbf{x}-\mathbf{x}_i\|\right) &\geq \mathbf{J}_i+\eta+\lambda\|\mathbf{x}-\mathbf{x}_i\|,\\ \beta\left(\mathbf{J}_i+\eta-\lambda\|\mathbf{x}-\mathbf{x}_i\|\right) &\geq 2\lambda\|\mathbf{x}-\mathbf{x}_i\|,\\ \beta(\mathbf{J}_i+\eta) &\geq (2+\beta)\lambda\|\mathbf{x}-\mathbf{x}_i\|.\end{aligned} \tag{36}$$

Hence,

$$\|\mathbf{x}-\mathbf{x}_i\| \;\leq\; \frac{\beta}{(2+\beta)\lambda}(\mathbf{J}_i+\eta). \tag{37}$$

This proves (8). To get (10), we have the following decomposition,

$$\begin{aligned}
\sup_{\mathbf{x}\in\mathbb{X}} \frac{J^\pi(\mathbf{x}) - J(\mathbf{x})}{J(\mathbf{x})+\eta} &= \sup_{\mathbf{x}\in\mathbb{X}} \left\{ \frac{J^\pi(\mathbf{x}) - J(\mathbf{x},\varepsilon)}{J(\mathbf{x},\varepsilon)+\eta} \frac{J(\mathbf{x},\varepsilon)+\eta}{J(\mathbf{x})+\eta} + \frac{J(\mathbf{x},\varepsilon) - J(\mathbf{x})}{J(\mathbf{x})+\eta} \right\} \\
&= \sup_{\mathbf{x}\in\mathbb{X}} \left\{ \left(\frac{J^\pi(\mathbf{x}) - J(\mathbf{x},\varepsilon)}{J(\mathbf{x},\varepsilon)+\eta} + 1 \right) \frac{J(\mathbf{x},\varepsilon)+\eta}{J(\mathbf{x})+\eta} - 1 \right\} \\
&\overset{(i)}{\leq} \sup_{\mathbf{x}\in\mathbb{X}} \left\{ \left(\sup_{\mathbf{x}\in\mathbb{X}} \left\{ \frac{J^\pi(\mathbf{x}) - J(\mathbf{x},\varepsilon)}{J(\mathbf{x},\varepsilon)+\eta} \right\} + 1 \right) \frac{J(\mathbf{x},\varepsilon)+\eta}{J(\mathbf{x})+\eta} - 1 \right\} \\
&\overset{(ii)}{\leq} \sup_{\mathbf{x}\in\mathbb{X}} \left\{ (\beta+1) \frac{J(\mathbf{x},\varepsilon)+\eta}{J(\mathbf{x})+\eta} - 1 \right\} \\
&= (\beta+1) \sup_{\mathbf{x}\in\mathbb{X}} \left\{ \frac{J(\mathbf{x},\varepsilon)+\eta}{J(\mathbf{x})+\eta} \right\} - 1 \\
&\overset{(iii)}{\leq} (\beta+1) \sup_{\mathbf{x}\in\mathbb{X}} \left\{ \frac{J(\mathbf{x},\varepsilon)+\eta}{J(\mathbf{x},\varepsilon)+\eta-L\varepsilon} \right\} - 1 \\
&= (\beta+1) \left(1 + \sup_{\mathbf{x}\in\mathbb{X}} \left\{ \frac{L\varepsilon}{J(\mathbf{x},\varepsilon)+\eta-L\varepsilon} \right\} \right) - 1 \\
&= (\beta+1) \left(1 + \frac{\frac{L\varepsilon}{\eta}}{1-\frac{L\varepsilon}{\eta}} \right) - 1,
\end{aligned}$$

where (i) follows from the nonnegativity $\frac{J^\pi(\mathbf{x})-J(\mathbf{x},\varepsilon)}{J(\mathbf{x},\varepsilon)+\eta} \geq 0$ and $\frac{J(\mathbf{x},\varepsilon)+\eta}{J(\mathbf{x})+\eta} \geq 1$. Using (9) yields (ii). For (iii), if ε is small enough to let $J(\mathbf{x},\varepsilon)+\eta-L\varepsilon > 0$, $\forall \mathbf{x} \in \mathbb{X}$, we invoke the first item of Assumption 4 to lower bound the denominator in $\frac{J(\mathbf{x},\varepsilon)+\eta}{J(\mathbf{x})+\eta}$. Finally, by rearranging the last term, we finished the proof. ■

Remark 3 (Upper bound on relative error.) *From (36), we have a sufficient upper bound of the relative error as,*

$$\mathrm{Err} = \frac{2\lambda\|\mathbf{x}-\mathbf{x}_i\|}{\mathbf{J}_i + \eta - \lambda\|\mathbf{x}-\mathbf{x}_i\|}, \tag{38}$$

where it is defined as Err. *It will be used later on Algorithm 2.*

B.4. Proposition 4

Proof Recall the Theorem 2, we call the β appears in (8) and (9) as β' to clearify the difference between the relative error tolerance on (9) and (10). Then, if a cover over $\mathbb{X}$ satisfies (8) and $\eta > L\varepsilon$, then a $\frac{\beta'\eta+L\varepsilon}{\eta-L\varepsilon}$-original optimality holds in (10). Now, given a error tolerance β on (10), we could solve the maximum β' to let β-*optimal policy* hold is,

$$\begin{aligned}
\frac{\beta'\eta + L\varepsilon}{\eta - L\varepsilon} &\leq \beta \\
\beta' &\leq \frac{\beta(\eta - L\varepsilon) - L\varepsilon}{\eta},
\end{aligned} \tag{39}$$

where the result means if a specific β' satisfies (39), then the cover satisfies (8) with this β' satisfies β-*optimal policy*. Besides, because $g(\beta') = \frac{\beta'}{\lambda(2+\beta')} = \frac{1}{\lambda} - \frac{2}{\lambda(2+\beta')}$ is monotonically increasing regards to β', the loosest bound on $\|\mathbf{x} - \mathbf{x}_i\|$ to ensure β-*optimal policy* is,

$$\begin{aligned}\|\mathbf{x} - \mathbf{x}_i\| &\leq \frac{\frac{\beta(\eta - L\varepsilon) - L\varepsilon}{\eta}}{\lambda\left(2 + \frac{\beta(\eta - L\varepsilon) - L\varepsilon}{\eta}\right)}\eta \\ &= \frac{(\beta(\eta - L\varepsilon) - L\varepsilon)\eta}{\lambda\left(2\eta + \beta(\eta - L\varepsilon) - L\varepsilon\right)},\end{aligned} \tag{40}$$

where we erase the dependence of $\mathbf{J}_i$ in (8) because we want a sufficient homogeneous bound (Algorithm 1 samples points uniformly) over $\mathbb{X}$. We denoting the half in RHS of (40) as r_β for the $2r_\beta$-covering on $\mathbb{X}$ because of the following triangular inequality. For any $\mathbf{x} \in \mathbb{X}$, there always exists a ball with radius r_β to let, $\exists \mathbf{x}_i \in \mathcal{D}$, $\mathbf{x}_i, \mathbf{x} \in \mathbb{B}_{r_\beta}$. Suppose this ball is centered at $\mathbf{c}$, then,

$$\|\mathbf{x} - \mathbf{x}_i\| \leq \|\mathbf{x} - \mathbf{c}\| + \|\mathbf{x}_i - \mathbf{c}\| \leq 2r_\beta, \tag{41}$$

which ensures (40) holds. Consequently, with probability $1 - \delta$, the Algorithm 1 will output a β-optimal policy at most after:

$$\mathcal{O}\left(\mathrm{N}_{\text{cover}}\left(\mathbb{X}; r_\beta\right) \log \mathrm{N}_{\text{cover}}\left(\mathbb{X}; r_\beta\right) \log \frac{1}{\delta}\right), \tag{42}$$

iterations. Additionally, from Proposition 3, we know that for one-step feasibility. Then by the same reason as (41), it is sufficient to cover $\mathbb{X}$ by balls with radius $\frac{\varepsilon}{L_f}$. Hence, the covering complexity under the same probability threshold $1 - \delta$ is,

$$\mathcal{O}\left(\mathrm{N}_{\text{cover}}\left(\mathbb{X}; \frac{\varepsilon}{2L_f}\right) \log \mathrm{N}_{\text{cover}}\left(\mathbb{X}; \frac{\varepsilon}{2L_f}\right) \log \frac{1}{\delta}\right). \tag{43}$$

At last, by denoting $r \triangleq \min\left\{\frac{(\beta(\eta - L\varepsilon) - L\varepsilon)\eta}{2\lambda(2\eta + \beta(\eta - L\varepsilon) - L\varepsilon)}, \frac{\varepsilon}{2L_f}\right\}$, we take the higher order complexity between (42) and (43). Here we finished the proof. ■

B.5. Considerations on Algorithm 2

Assumption 5 (Regularity of $\mathbb{X}$ and $\|\cdot\|_{\mathbb{X}}$) *$\mathbb{X}$ is a hypercube containing the origin and $\|\cdot\|_{\mathbb{X}} = \|\cdot\|_\infty$.*

Algorithm 2 mainly follows the idea of Neyman allocation (Garivier and Kaufmann, 2016; Adusumilli, 2022; Fogliato et al., 2024) and deterministic non-parametric optimization (Munos, 2011; Kuzborskij and Cesa-Bianchi, 2020; Sinclair et al., 2020). We construct a refinement tree whose nodes are axis–aligned hypercubes (ℓ_∞-balls) $\mathbb{X}^{k,j} \subseteq \mathbb{X}$, where k denotes the tree layer (depth) and j indexes the nodes at the layer k of the current $\mathrm{Tree}(t)$. A layer-k hypercube has radius h_k (side length $2h_k$). Splitting replaces a node by 3^n children, each of radius $h_{k+1} = h_k/3$. This splitting rule follows the logic of Siegelmann et al. (2025, Algorithm 2), we split each edge into $\frac{1}{3}$

because the central point still remains a central point in the new subgrid $\mathcal{H}_{k,j}$[4]. Specifically, a list $I = \left\{ I_1, I_2, ..., I_{\left\lceil \log_3\left(\frac{h_0 L_f}{\varepsilon}\right)\right\rceil} \right\}$ is defined initial to keep on track of the number of leaves in different layer k in current $\mathrm{Tree}(t)$. The length of list is $\left\lceil \log_3\left(\frac{h_0 L_f}{\varepsilon}\right)\right\rceil$ because h_0 can split at most this time to reach a sufficient small radius $\frac{\varepsilon}{L_f}$ in Proposition 3. The Algorithm 2 proceeds in two stages:

(a) we first construct a nonparametric policy that guarantees one-step feasibility;

(b) given a budget N, we adaptively split cells to achieve β-optimality relative to the best achievable performance.

Initialization. We initialize with a uniform cover $\mathcal{H}_0 = \{\mathbb{X}^{1,i}\}_{i=1}^{N_0}$ of common radius h_0 and cardinality N_0. For each cell $\mathbb{X}^{k,j}$ with center $\mathbf{x}_{k,j}$, we evaluate the dynamics along the horizon $m = i, \ldots, i+T-1$ by solving (4), obtaining the control inputs $\mathbf{u}_{k,m}$ and successor states $\mathbf{J}_{k,m}$. The resulting optimal trajectory is $\tau_{k,i} = \left\{(\mathbf{x}_{k,m},\, \mathbf{u}_{k,m},\, \mathbf{J}_{k,m})\right\}_{m=i}^{T+i-1}$.

Acceptance/Refinement rule. A cell $\mathbb{X}^{k,j}$ is *accepted* if (i) it is one-step feasible for stage (a), or (ii) it is β-optimal policy for stage (b) at its center-trajectory $\tau_{k,i} = \{(\mathbf{x}_{k,m}, \mathbf{u}_{k,m}, \mathbf{J}_{k,m})\}_{m=i}^{i+T-1}$, respectively. If both (i) and (ii) hold simultaneously, the cell $\mathbb{X}^{k,j}$ is *permanently kept*. Otherwise, $\mathbb{X}^{k,j}$ is *refined*: we split it into with subgrid $\mathcal{H}_{k,j}$ with 3^n uniform children of radius $h_{k+1} = h_k/3$ and replace the parent in the current tree $\mathrm{Tree}(t)$ by these children (**Block 1**). The central trajectory $\tau_{k+1,i}$ for each child cell $\mathbb{X}_{k+1,i}$ in $\mathcal{H}_{k,j}$ is computed and collected in the data set $\mathcal{D}$.

In stage (b), whenever a cell is refined, we update the evaluation budget by $K \leftarrow K + 3^n$.

Iteration and stopping. At iteration t, we traverse all current leaves of $\mathrm{Tree}(t)$ and apply the acceptance/refinement rule to each leaf; the current iteration t ends once all leaves have been processed. The procedure terminates in stage (a) as soon as the resulting tree is one-step feasible. For stage (b), the algorithm terminates when either the evaluation budget is exhausted, i.e., $N - K < 3^n$, or every leaf in current $\mathrm{Tree}(t)$ is already β-optimal policy.

In stage (b), if the remaining budget $N - K$ is insufficient to refine all non–β-optimalleaves in $\mathrm{Tree}(t)$, we refine only $\left\lfloor \frac{N-K}{3^n} \right\rfloor$ leaves—specifically, those with the largest relative error bounds $\mathrm{RErr}_{k,j}$ as defined in (38).

Remark 4 (Cover a regular domain (Assumption 5).) *In Assumption 5, we assume $\mathbb{X}$ is a regular hypercube to ensure a uniform grid to cover $\mathbb{X}$ without overlapping or omission. This assumption here is just to ease the clarfication of our presentation on Algorithm 2. The Algorithm 2 could be extend to any shape of $\mathbb{X}$ easily by consider a feasible outer cover $\mathbb{X}_0$ ($\forall \mathbf{x} \in \mathbb{X}_0,\ J(\mathbf{x}, \varepsilon) < +\infty$) constructed by non-overlapping same radius ℓ_∞-balls.*

Appendix C. Details on the experiments

C.1. Inverted Pendulum

We consider the inverted pendulum with angle x_1 and angular velocity x_2, with equations of motion given by:

$$\dot{\mathbf{x}} = \frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{g}{l}\sin x_1 \end{bmatrix} + \frac{1}{ml^2}\begin{bmatrix} 0 \\ 1 \end{bmatrix} u\ , \tag{44}$$

4. For example, if we split each edge of the cell $\mathbb{X}^{k,j}$ by half, then the central point, $\mathbf{x}_{k,j}$, is not usable for any children cell $\mathbb{X}^{k+1,i}$ in the new subgrid $\mathcal{H}_{k,j}$.

where $u \in \mathbb{R}$ is the scalar torque applied to the axis of the pendulum. We consider the discrete-time dynamics of (44) with sampling time $\delta t = 0.05s$, we let $\mathbf{u}_t \in \mathbb{U} = [-5, 5], m = 1kg, l = 1m, g = 9.82\frac{m}{s^2}$. The stage cost is given by:

$$c(\mathbf{x}_t, \mathbf{u}_t) = \mathbf{x}_t^\top \begin{bmatrix} 1 & 0 \\ 0 & 0.1 \end{bmatrix} \mathbf{x}_t + 0.01\|\mathbf{u}_t\|^2.$$

We consider a time horizon $T = 100$, and obtain different datasets $\mathcal{D}$ by performing a uniform grid of the state space $\mathbb{X} = [-2, 2]^2$ with $G \in \{5, 7, 9, 11\}$ points per dimension, and run offline MPC to get full length trajectories starting from those points.

C.2. Minimum time with control regularization

$$\dot{\mathbf{x}} = \frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0.2 & 1 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \tag{45}$$

We consider the discrete-time version of (45) with sampling time $\delta t = 0.1s$, stage cost:

$$c(\mathbf{x}_t, \mathbf{u}_t) = 1 + 10 \cdot \|\mathbf{u}_t\| \ . \tag{46}$$

and terminal constraint $\mathbf{x}_T = 0$. Even though this cost functions are not captured by our theory above, we want to show that our policy still achieves good performance in this case. We consider the discrete-time dynamics of (45) with sampling time $\delta t = 0.1s$, horizon $T = 200$ and do uniform gridding of the state space $\mathbb{X} = [-2, 2]^2$ with the same grids as for the pendulum, $\mathbb{U} = [-1, 1]$.

C.3. Verification on constrained LQR

$$A = \begin{bmatrix} 1 & 0.1 \\ 0 & 1 \end{bmatrix}; \qquad B = \begin{bmatrix} 0.15 \\ 1 \end{bmatrix}, \tag{47}$$

We consider stage cost $c(\mathbf{x}, \mathbf{u}) = \|\mathbf{x}\|^2 + \|\mathbf{u}\|^2$, constraint sets $\mathbb{X} = [-3, 3]^2, \mathbb{U} = [-2, 2]$ and a horizon $T = 10$. For our policy we used $L_f = 1.1, \lambda = 1, \beta = 5, \eta = 0.01$. This parameters were chosen in an ad-hoc manner to get a convergent result in 5 iterations or less—both for feasibility and performance—so that Figure 4 was illustrative. With tighter requirements (e.g. $\beta = 0.1, \eta = 1e{-}6$) Algorithm 2 necessitates many more iterations.

Algorithm 2: Local Adaptive Data collector

Input: Conservative threshold $\varepsilon > 0$. Stopping time T. Initial covering radius h_0. Desired gap β. Slack η.

Initialize: $\mathcal{D} = \emptyset$. $\text{Tree}(0) = \mathbb{X}$. $I = \mathbf{0}_{1\times\left\lceil \log_3\left(\frac{h_0 L_f}{\varepsilon}\right)\right\rceil}$.

Sample $\mathbf{x}$ through the uniform grid $\{\mathbb{X}^{1,1}, \mathbb{X}^{1,1}, \mathbb{X}^{1,2}, ..., \mathbb{X}^{1,N_0}\} = \mathcal{H}_0$ with radius h_0.

Get trajectory $\tau_{1,i} = \{(\mathbf{x}_{1,m}, \mathbf{u}_{1,m}, \mathbf{J}_{1,m})\}_{m=i}^{T+i-1}$ by solving (4) from $\mathbf{x}_{1,i} \in \mathbb{X}^{1,i} \subseteq \mathcal{H}_0$.

$\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{x}_{1,m}, \mathbf{u}_{1,m}, \mathbf{J}_{1,m})\}$ **for** $(\mathbf{x}_{1,m}, \mathbf{u}_{1,m}, \mathbf{J}_{1,m})$ **in** $\tau_{1,i}$. $I_1 \leftarrow N_0$. $\text{Tree}(1) \leftarrow \mathcal{H}_0$. $t \leftarrow 1$.

while $\exists\, \mathbb{X}^{k,j} \subseteq \text{Tree}(t)$ *infeasible,* **do**

  for $\mathbb{X}^{k,j} \subseteq \text{Tree}(t)$ **do**

    Check the one-step feasibility (Equation (6), Theorem 2).

    if $\mathbb{X}^{k,j}$ *one-step infeasible,* **then**

      Block 1:

        Split the cell into the uniform grid $\{\mathbb{X}^{k+1,I_{k+1}+1}, ..., \mathbb{X}^{k+1,I_{k+1}+3^n}\} = \mathcal{H}_{k,j}$.

        $\text{Tree}(t) \leftarrow \text{Tree}(t) \cup \mathcal{H}_{k,j}/\mathbb{X}^{k,j}$. $I_{k+1} = I_{k+1} + 3^n$.

        for $\mathbb{X}^{k+1,i} \subseteq \mathcal{H}_{k,j}$ **do**

          Sample $\mathbf{x}_{k+1,i}$ as the central point of $\mathbb{X}^{k+1,i}$.

          Get trajectory $\tau_{k+1,i} = \{(\mathbf{x}_{k+1,m}, \mathbf{u}_{k+1,m}, \mathbf{J}_{k+1,m})\}_{m=i}^{T+i-1}$ by solving (4) from $\mathbf{x}_{k+1,i} \in \mathbb{X}^{k+1,i} \subseteq \mathcal{H}_{k,j}$.

          $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{x}_{k+1,m}, \mathbf{u}_{k+1,m}, \mathbf{J}_{k+1,m})\}$ **for** $(\mathbf{x}_{k+1,m}, \mathbf{u}_{k+1,m}, \mathbf{J}_{k+1,m})$ **in** $\tau_{k+1,i}$.

        end

      End Block 1

    end

  end

  $\text{Tree}(t+1) \leftarrow \text{Tree}(t)$. $t \leftarrow t+1$.

end

$K \leftarrow 0$.

while $N \geq K$, **do**

  Compute the number of cells $\mathbb{X}^{k,j} \subseteq \text{Tree}(t)$ not β-optimal as $N_{\text{Tree}(t)}$.

  if $3^n N_{\text{Tree}(t)} \leq N - K$, **then**

    for $\mathbb{X}^{k,j} \subseteq \text{Tree}(t)$ **do**

      Check the β-optimality (Equation (8), Theorem 2).

      if $\mathbb{X}^{k,j}$ *not* β*-optimal policy,* **then**

        Run **Block 1**. $K \leftarrow K + 3^n$.

      end

      if $\exists\, \mathbb{X}^{k,j} \subseteq \text{Tree}(t)$ *not* β*-optimal,* **then**

        $\text{Tree}(t+1) \leftarrow \text{Tree}(t)$. $t \leftarrow t+1$.

      else

        End All

      end

    end

  else

    Compute the relative error $\text{RErr}_{k,j}$ (Equation (38)) **for** $\mathbb{X}^{k,j} \subseteq \text{Tree}(t)$ not β-optimal.

    Run **Block 1** for $\lfloor \frac{N-K}{3^n} \rfloor$ cells with largest $\text{RErr}_{k,j}$.

    $K \leftarrow K + \left(\lfloor \frac{N-K}{3^n} \rfloor + 1\right) 3^n$.

  end

end

Output: Nonparametric policy $\pi_{\mathcal{D}}$ for $\mathbb{X}$, feasible over $\mathbb{X}$, with performance guarantees.