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 conditions 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. New solutions can be incorporated straightforwardly without the need for retraining, enabling continual improvement. 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 is how to get *fast, high-quality* controls 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). A substantial body of work 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). These projection schemes, however, are costly and require solving additional optimization problems. Other works enable explicit (nonlinear) MPC via neural networks (Hertneck et al., 2018), or by nonparametric methods, for example by using set membership approximation (Canale et al., 2009), kernel

regression (Carnerero et al., 2023; Huang et al., 2023), quasi-interpolation (Ganguly and Chatterjee, 2025), nonlinear piecewise approximations (Trinh et al., 2016) and tube MPC (Bayer et al., 2016). One drawback of neural network methods (Chen et al., 2018; Hertneck et al., 2018) is that they are difficult to adapt to new data, often requiring retraining from scratch. Some of the nonparametric methods cited (Canale et al., 2009) only guarantee stability/performance *asymptotically*, while others (Carnerero et al., 2023; Trinh et al., 2016) provide no guarantees or rely on too strong assumptions on the system dynamics (Bayer et al., 2016).

In contrast with the afforementioned literature, we propose a novel framework that uses offline solutions to derive a nonparametric policy with **non-asymptotic performance and feasibility guarantees**. New data modifies our policy locally, enabling **continual improvement**: new data is incorporated with no retraining needed and without appreciable performance degradation. 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. This policy can learn continually, with new solutions added without any retraining.
3. We establish conditions on the offline data that ensure recursive feasibility and bounds on the optimality gap over the whole domain.
4. Empirically, we show that this policy can be implemented efficiently as a lookup rule on a CPU , 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 algorithms 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 recursive feasibility and suboptimality. Experiments in **Section** 6 show our verification algorithm in action and empirically contrast trade-offs between performance and controller latency for our method versus 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 $\mathbf{u}$:*

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

where $\|\cdot\|_{\mathbb{X}}$ and $\|\cdot\|_{\mathbb{U}}$ are appropriate 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.

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 preceding assumption is not overly restrictive: it captures (among other cases) infinite-horizon LQR control (Bertsekas, 2012, Ch. 3) and shortest path problems (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 $\mathbf{u}_0,\dots,\mathbf{u}_{H-1}$, 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-satisfying 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:

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

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

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

$$\text{subject to: (1b), (1d)}, \mathbf{x}_t\in\mathbb{X}_{-\varepsilon}, \qquad t=1,\dots,T-1 \tag{4b}$$

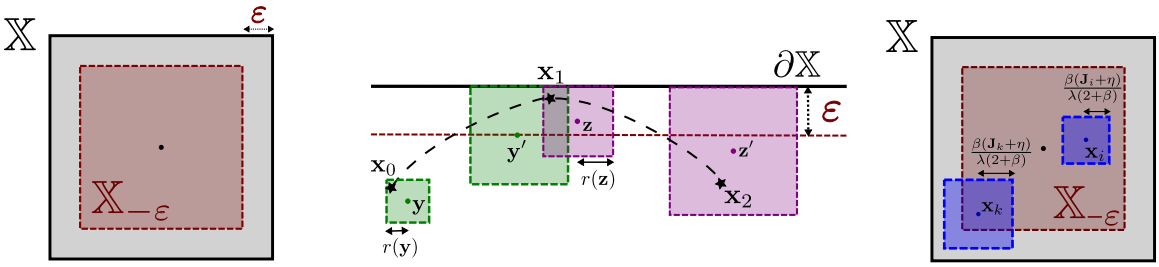

Figure 1: Left: Original constraint set $\mathbb{X}$ and its erosion $\mathbb{X}_{-\varepsilon}$. Middle: Feasibility certificates under our framework. The trajectory $(\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \ldots)$ marked with '$\star$' is produced by our policy. Optimal transitions $\mathbf{y} \overset{\pi^\star}{\to} \mathbf{y}'$ 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.

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. The trajectory $\mathbf{z} \to \mathbf{z}'$ in Figure 1 is an example of this behavior.[1]

Assumption 4 (Cost-to-go of conservative problem is Lipschitz) *There exists $L > 0$ and $L_J > 0$ such that for any ε satisfying the assumption above, we have:*

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

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 assumption can be relaxed by enforcing that the system enters $\mathbb{X}_{-\varepsilon}$ after at most K steps, i.e. changing (4b) to $\mathbf{x}_t \in \mathbb{X}_{-\varepsilon}\ \forall t \geq K$, for some $K : 1 \leq K < T$. Our results can be easily generalized to the case $K > 1$, which is related to notions of *control recurrence* (Shen et al., 2022; Sibai and Mallada, 2026). For ease of exposition we focus on the case $K = 1$ and defer the general one for future work.

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 highlight 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 study conditions on $\pi_{\mathcal{D}}$ for *(i)* recursive feasibility and *(ii)* bounded suboptimality 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. $\bigcup_{i=1}^{|\mathcal{D}|} \mathbb{B}\left(\mathbf{x}_i, r(\mathbf{x}_i)\right) \supseteq \mathbb{X}$*, where* $r(\mathbf{x}_i) = \frac{\operatorname{dist}(f(\mathbf{x}_i,\mathbf{u}_i),\partial\mathbb{X})}{L_f}$

Note that the former condition may be overly conservative, since it considers the smallest possible feasibility radius for any point in the dataset (like 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: $J(\mathbf{x}_0, \varepsilon) \leq J(\mathbf{x}_0', \varepsilon) + L_J\|\mathbf{x}_0 - \mathbf{x}_0'\|$. 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* $J^{\lambda}_{\text{ub}} : \mathbb{X} \to \mathbb{R}$, $J^{\lambda}_{\text{lb}} : \mathbb{X} \to \mathbb{R}$*:*

$$J^{\lambda}_{\text{ub}}(\mathbf{x}) \triangleq \min_{1\leq i\leq|\mathcal{D}|} \left\{\mathbf{J}_i + \lambda\|\mathbf{x} - \mathbf{x}_i\|\right\} \qquad J^{\lambda}_{\text{lb}}(\mathbf{x}) \triangleq \max_{1\leq i\leq|\mathcal{D}|} \left\{\mathbf{J}_i - \lambda\|\mathbf{x} - \mathbf{x}_i\|\right\}$$

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

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:*

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

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

The preceding 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 $J^{\lambda}_{\text{ub}}(\cdot)$ is a data-dependent bound that *improves* (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, adapted here to constrained settings. We close this section by establishing performance guarantees for policy $\pi_{\mathcal{D}}$.

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 $\tau_i = \{(\mathbf{x}_k, \mathbf{u}_k, \mathbf{J}_k)\}_{k=i}^{T+i-1}$ solving (4) from $\mathbf{x}_i$, add each transition to $\mathcal{D}$.
end
Output: Nonparametric policy $\pi_{\mathcal{D}}$ with feasibility & performance guarantees (Prop. 4)

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

$$\sup_{\mathbf{x}\in\mathbb{X}} \frac{J^{\pi}(\mathbf{x}) - J(\mathbf{x}, \varepsilon)}{J(\mathbf{x}, \varepsilon) + \eta} \leq \beta\ . \tag{8}$$

Moreover, if $\eta > L\varepsilon$ then: $\sup_{\mathbf{x}\in\mathbb{X}} \frac{J^{\pi}(\mathbf{x}) - J(\mathbf{x})}{J(\mathbf{x}) + \eta} \leq \frac{\beta\eta + L\varepsilon}{\eta - L\varepsilon}$.

The proof is in Appendix B.3. We highlight that (8) bounds the *relative suboptimality* of our policy with respect to the optimal solution of the *conservative* problem (4). By contrast, the next equation quantifies the *relative gap* to the optimal solution of the *original* problem (1).

On bounding Lipschitz constants Our nonparametric policy (and the upper- and lower-bounds of J) depends on a hyperparameter $\lambda > 0$. Sufficient conditions for a value of λ that yields the results of theorems 1 and 2 in turn rely on the Lipschitz constants L_f and L_J (or at least upper bounds on them). We recognize this as a limitation of our work but also remark that these constants can be learned from trajectories, see e.g. Knuth et al. (2021).

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 (Haussler and Warmuth, 2018) for this algorithm next, since people are fond of them.

Proposition 4 (Sample complexity for Algorithm 1) *Let $\beta > 0$, $\eta > L\varepsilon(1 + \frac{1}{\beta})$ and $\lambda > 0$ satisfying (7). Pick any $\delta \in (0, 1)$. Define $r \triangleq \frac{1}{2}\min\left\{\frac{\eta}{\lambda} \cdot \frac{\beta\eta - L\varepsilon(1+\beta)}{(2+\beta)\eta - L\varepsilon(1+\beta)}, \frac{\varepsilon}{L_f}\right\}$. With probability at least $1 - \delta$, Algorithm 1 outputs both a recursively feasible and β-optimal policy over $\mathbb{X}$ after at most:*

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

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).

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

Algorithm 2: Verification

Initialize: $\mathcal{D} = \emptyset$. `Unverified` = $\{\mathbb{X}\}$; `Verified`=$\{\}$; ▷ $\mathbb{X}$ initially unverified
repeat
  for $\mathbb{X}_i$ *in* `Unverified` **do**
    Get trajectory $\tau_i = \{(\mathbf{x}_k, \mathbf{u}_k, \mathbf{J}_k)\}_{k=i}^{T+i-1}$ by solving (4) from $\mathbf{x}_i$, center of $\mathbb{X}_i$.
    if $(\mathbf{x}_i, \mathbf{u}_i)$ *one-step feasible over* $\mathbb{X}_i$ *(Prop. 2)* **then**
      Move $\mathbb{X}_i$ to `Verified`.
    else
      Split $\mathbb{X}_i$ into 3^n cells; add children to `Unverified`.
    end
  end
until `Unverified`$= \emptyset$ ▷ Feasibility over $\mathbb{X}$ certified;
Move all cells from `Verified` to `Unverified`.
repeat
  for $\mathbb{X}_i$ *in* `Unverified` **do**
    Get trajectory $\tau_i = \{(\mathbf{x}_k, \mathbf{u}_k, \mathbf{J}_k)\}_{k=i}^{T+i-1}$ by solving (4) from $\mathbf{x}_i$, center of $\mathbb{X}_i$.
    if $(\mathbf{x}_i, \mathbf{J}_i)$ *performant over* $\mathbb{X}_i$ *(Thm. 2)* **then**
      Move $\mathbb{X}_i$ to `Verified`.
    else
      Split $\mathbb{X}_i$ into 3^n cells; add children to `Unverified`.
    end
  end
until `Unverified`$= \emptyset$ ▷ Performance over $\mathbb{X}$ certified.;

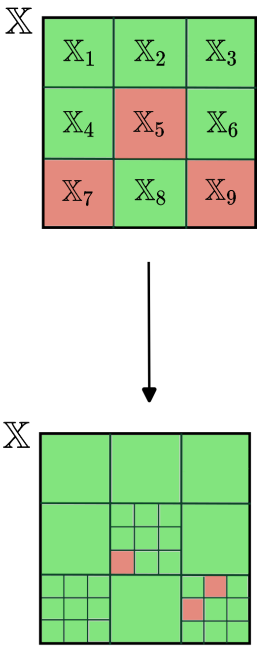

Figure 2: Algorithm 2 and the visualization of the cell verification/splitting method. Each cell is tested against two criteria: (i) one-step feasibility and (ii) performance. Cells that pass are shown in green and kept in `Verified`; those that fail are shown in red and are split into 3^n children cells. The children are then re-verified using the same criteria, proceeding recursively.

See Appendix B.4 for a proof of Proposition 4. Algorithm 1 enjoys the guarantees above but may be inefficient due to stochastic sampling. An alternative verification procedure is provided in Algorithm 2, which recursively partitions $\mathbb{X}$ into disjoint cells until feasibility and a desired performance is guaranteed. A detailed description can be found in Appendix C.

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: One potential bottleneck 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 . We run `FAISS` in CPU-only mode. Instead of fully solving $\iota(\mathbf{x}) = \arg\min_{1 \le i \le |\mathcal{D}|} \left\{\mathbf{J}_i + \lambda \cdot \|\mathbf{x} - \mathbf{x}_i\|\right\}$, we invoke `FAISS` to get the k-nearest-neighbors to a query point $\mathbf{x}$. Call these points $\mathcal{N}_k(\mathbf{x})$. Then, the minimization done at each step is approximated by: $\iota(\mathbf{x}) \approx \arg\min_{i:\ \mathbf{x}_i \in \mathcal{N}_k(\mathbf{x})} \left\{\mathbf{J}_i + \lambda \cdot \|\mathbf{x} - \mathbf{x}_i\|\right\}$, which amounts to taking the smallest value of a k-dimensional array. A large enough value of k will ensure that $\mathcal{N}_k(\mathbf{x})$ contains the true minimizer, but renders the look-up slower. We use $k = 100$.

Hardware/Software Setup We use `do-mpc` (Fiedler et al., 2023), an open-source python module for MPC to conduct the experiments. This software interfaces with `CasADi` (Andersson et al., 2019), which uses a symbolic framework to build each problem, and runs automatic differentiation to obtain trajectories and controls. The optimization solver running in the background is `IPOPT` (Wächter and Biegler, 2006; Biegler and Zavala, 2009), with a tolerance of 10^{-8}. All experiments are run on a Macbook Air M2 with 8GB of RAM, without GPUs.

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 D. 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. The performance of both MPC controllers (with varying lookahead horizon $H \leq T$) and our nonparametric policies (labeled `MINT`, for `M`onotonically `I`mproving bound-based `N`onparametric policy based on `T`rajectories) are evaluated on $M = 100$ trajectories from random initial conditions. Figure 3, shows the result of these trajectories. Boxes show the interquartile range (25% - 75%) of the empirical distribution over the M trajectories; black lines correspond to the median values, means are shown in green, whiskers extend to 1.5 times the inter-quartile 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. The right panels shows the trade-off between the median normalized gap and the median time taken to get controls. Our nonparametric policy strikes a good balance between good performance (low gap) with lower computation time.

Verification We test Algorithm 2 on a discrete LQR problem. Details on the setup and on the hyperparameters of our method are in Appendix D. Figure 4 shows Algorithm 2 in action: it recursively partitions the state space into smaller cells 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). Feasibility is harder to verify near the boundary of $\mathbb{X}$ (requiring more splitting), to be expected from our guarantees on a feasibility radius (Prop. 2). A target gap is harder to verify near the origin, since our target gap (8) is relative and $J(\mathbf{x}) \approx 0$ in that vicinity.

7. Conclusions & Future work

We proposed a nonparametric, data-driven scheme to accelerate MPC by reusing offline-computed trajectories. Our policy picks stored controls by trading off cost-to-go and state proximity. Under mild Lipschitz and coverage assumptions, the controller enjoys recursive feasibility and has explicit performance bounds; with sufficient dataset coverage the relative suboptimality can be made arbitrarily small, establishing an explicit trade-off between desired performance and memory requirements. 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.

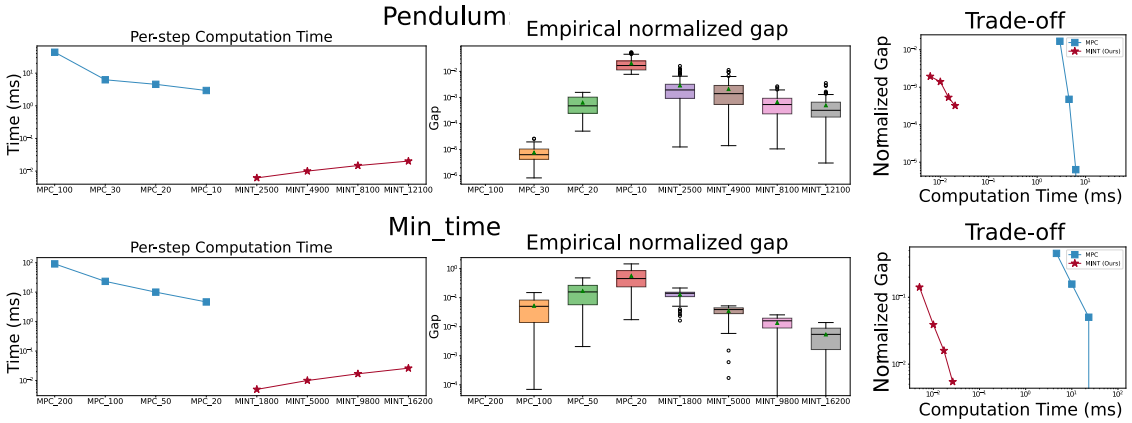

Figure 3: Statistics for the inverted pendulum (top) and minimum time problem (bottom) over 100 trajectories. Left: Per-step latency (in ms) for each controller. Our controllers (`MINT_XXXX`, red $\star$'s) are ordered left to right from smallest to largest dataset $\mathcal{D}$, MPC controllers (blue $\square$'s) ordered from largest to smallest horizon. 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 the controllers. Our method is substantially faster than MPC and, with sufficient data, outperforms MPC with shorter lookahead horizons.

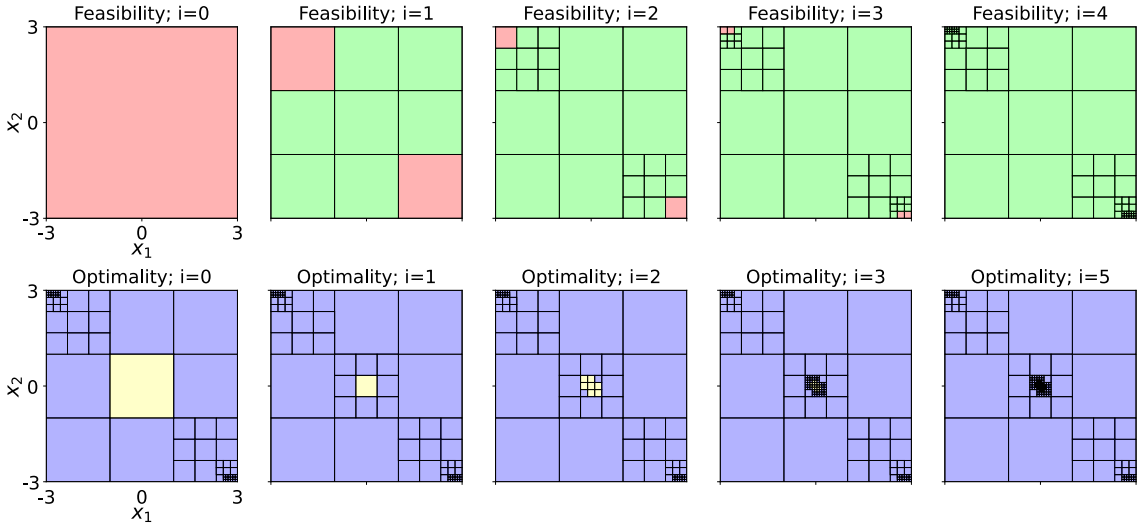

Figure 4: Algorithm 2 in action: feasibility (top row) and optimality (bottom row) certificates for LQR, iterations ordered left to right. Top: Shown in red are cells that don't satisfy the feasibility condition (Prop. 2). The algorithm recursively splits each cell until verification, shown in green. Bottom: verification of gap (Theorem 2). Guaranteed suboptimal cells are shown in blue. At termination the algorithm has verified the whole state space $\mathbb{X}$.

Acknowledgments

A.C. is grateful to Pedro Izquierdo Lehmann for insightful discussions and suggestions during the first stages of this manuscript. This work was supported by the NSF Global Centers program under Grant No. 2330450 and by the DOE Office of Science (ASCR) under Award No. 826565.

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.

Joel A E Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl. CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36, 2019. doi: 10.1007/s12532-018-0139-4.

Florian A Bayer, Florian D Brunner, Mircea Lazar, Marc Wijnand, and Frank Allgöwer. A tube-based approach to nonlinear explicit mpc. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 4059–4064. IEEE, 2016.

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.

Lorenz T Biegler and Victor M Zavala. Large-scale nonlinear programming using ipopt: An integrating framework for enterprise-wide dynamic optimization. *Computers & Chemical Engineering*, 33(3):575–582, 2009.

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.

Massimo Canale, Lorenzo Fagiano, and Mario Milanese. Set membership approximation theory for fast implementation of model predictive control laws. *Automatica*, 45(1):45–54, 2009.

A Daniel Carnerero, Daniel R Ramirez, Daniel Limon, and Teodoro Alamo. Kernel-based state-space kriging for predictive control. *IEEE/CAA Journal of Automatica Sinica*, 10(5):1263–1275, 2023.

Agustin Castellano, Sohrab Rezaei, Jared Markowitz, and Enrique Mallada. Nonparametric policy improvement in continuous action spaces via expert demonstrations. In *Reinforcement Learning Conference*, 2025.

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.

Felix Fiedler, Benjamin Karg, Lukas Lüken, Dean Brandner, Moritz Heinlein, Felix Brabender, and Sergio Lucia. do-mpc: Towards fair nonlinear and robust model predictive control. *Control Engineering Practice*, 140:105676, 2023.

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.

Siddhartha Ganguly and Debasish Chatterjee. Explicit feedback synthesis driven by quasi-interpolation for nonlinear model predictive control. *IEEE Transactions on Automatic Control*, 70(7):4751–4758, 2025.

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.

David Haussler and Manfred Warmuth. The probably approximately correct (pac) and other learning models. *The Mathematics of Generalization*, pages 17–36, 2018.

Michael Hertneck, Johannes Köhler, Sebastian Trimpe, and Frank Allgöwer. Learning an approximate model predictive controller with guarantees. *IEEE Control Systems Letters*, 2(3):543–548, 2018.

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.

Linbin Huang, John Lygeros, and Florian Dörfler. Robust and kernelized data-enabled predictive control for nonlinear systems. *IEEE Transactions on Control Systems Technology*, 32(2):611–624, 2023.

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.

Craig Knuth, Glen Chou, Necmiye Ozay, and Dmitry Berenson. Planning with learned dynamics: Probabilistic guarantees on safety and reachability via lipschitz constants. *IEEE Robotics and Automation Letters*, 6(3):5129–5136, 2021.

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.

Yue Shen, Maxim Bichuch, and Enrique Mallada. Model-free learning of regions of attraction via recurrent sets. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 4714–4719. IEEE, 2022.

Hussein Sibai and Enrique Mallada. Recurrence of nonlinear control systems: Entropy, bit rates, and finite alphabet controllers. *Nonlinear Analysis: Hybrid Systems*, 59:101649, 2026.

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.

Van-Vuong Trinh, Mazen Alamir, Patrick Bonnay, and François Bonne. Explicit model predictive control via nonlinear piecewise approximations. *IFAC-PapersOnLine*, 49(18):259–264, 2016.

Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1): 25–57, 2006.

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

Appendix A.

In what follows we provide some useful inequalities that will aid the proofs of some results in the paper. First, by the Lipschitz assumption on $f(\mathbf{x}, \mathbf{u})$ and the fact that the origin is an equilibrium point, the following holds.:

For all $\mathbf{x} \in \mathbb{X}$ and for all $\mathbf{u} \in \mathbb{U}$ we have:

$$\|f(\mathbf{x}, \mathbf{u})\| \leq L_f \|\mathbf{x}\| + L_u \|\mathbf{u}\| \ . \tag{9}$$

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:

$$\left\|\mathbf{x}_t - \mathbf{x}'_t\right\| \leq L_f^k \cdot \left\|\mathbf{x}_0 - \mathbf{x}'_0\right\| + L_u \sum_{\ell=0}^{t-1} L_f^{t-1-\ell} \left\|\mathbf{u}_\ell - \mathbf{u}'_\ell\right\| \quad \forall t \geq 0 \ . \tag{10}$$

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

$$\left\|\mathbf{x}_1 - \mathbf{x}'_1\right\| \leq L_f \left\|\mathbf{x}_0 - \mathbf{x}'_0\right\| \ . \tag{11}$$

Appendix B. Proofs

B.1. Proposition 2

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:

$$R \triangleq \sup \{r \geq 0 : \mathbb{B}_{\mathbb{X}}(\mathbf{0}, r) \subseteq \mathbb{X}\} \ . \tag{12}$$

We will show the following two inequalities (the latter is Proposition 2).

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:

$$r(\mathbf{x}, \mathbf{u}) \triangleq \max \left\{0, \frac{R - L_u \|\mathbf{u}\|}{L_f} - \|\mathbf{x}\|\right\} \tag{13}$$

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:

$$r(\mathbf{x}') \triangleq \frac{R - \|\mathbf{x}'\|}{L_f} \leq \frac{\operatorname{dist}(\mathbf{x}', \partial\mathbb{X})}{L_f} \tag{14}$$

Proof

1. Let $\mathbf{x}' = f(\mathbf{x}, \mathbf{u})$. By virtue of (9):

$$\|\mathbf{x}'\| \leq L_f \|\mathbf{x}\| + L_u \|\mathbf{u}\| \tag{15}$$

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:

$$\|\mathbf{x}'_0\| - \|\mathbf{x}'\| \leq \|\mathbf{x}'_0 - \mathbf{x}'\| \leq L_f r \implies \tag{16}$$

$$\|\mathbf{x}'_0\| \leq \|\mathbf{x}'\| + L_f r \leq L_f \left(\|\mathbf{x}\| + r\right) + L_u \|\mathbf{u}\|, \tag{17}$$

where the first inequality in (16) follows from the reverse triangle inequality, and the second one from (10) (for two successor states under same control $\mathbf{u}$). In (17) we rearrange terms and use (15).

Imposing the right hand side of (17) be smaller than R:

$$L_f \left(\|\mathbf{x}\| + r\right) + L_u \|\mathbf{u}\| \leq R \implies r \leq \frac{R - L_u \|\mathbf{u}\|}{L_f} - \|\mathbf{x}\| \,, \tag{18}$$

as desired.

2. Re-using the first inequality in (17), and imposing that it be upper bounded by R:

$$\|\mathbf{x}'_0\| \leq \|\mathbf{x}'\| + L_f r \leq R \implies r \leq \frac{R - \|\mathbf{x}'\|}{L_f} \,. \tag{19}$$

For the remaining inequality, note:

$$\operatorname{dist}\left(\mathbf{x}', \partial\mathbb{X}\right) \geq R - \|\mathbf{x}'\| \quad \forall \mathbf{x}' \in \mathbb{X} \,. \tag{20}$$

■

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 $\mathcal{T}^\pi$:

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

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

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

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

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:

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

and the action-value function (Bertsekas et al., 2011) $Q : \mathbb{X} \times \mathbb{U} \to \mathbb{R} \cup \{+\infty\}$:

$$\begin{aligned} Q(\mathbf{x}_0, \mathbf{u}_0) \triangleq \min_{\mathbf{u}_{1:T-1}} & \sum_{t=0}^{T-1} \gamma^t c(\mathbf{x}_t, \mathbf{u}_t) + F(\mathbf{x}_T) \\ \text{subject to: } & \mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t), && t = 0, \dots, T-1 \\ & \mathbf{x}_t \in \mathbb{X}_{-\varepsilon}, && t = 1, \dots, T-1 \\ & \mathbf{u}_t \in \mathbb{U}, && t = 0, \dots, T-1 \end{aligned}$$

which is the optimal cost-to-go of the original problem fixing both $\mathbf{x}_0$ and $\mathbf{u}_0$. Note that $Q(\cdot, \cdot)$ satisfies the following Bellman equation:

$$Q(\mathbf{x}_0, \mathbf{u}_0) = c(\mathbf{x}_0, \mathbf{u}_0) + \gamma J(f(\mathbf{x}_0, \mathbf{u}_0)) \quad \forall (\mathbf{x}_0, \mathbf{u}_0) \in \mathbb{X} \times \mathbb{U}\,, \tag{21}$$

and furthermore:

$$Q(\mathbf{x}, \mathbf{u}) \geq J(\mathbf{x}) \quad \forall \mathbf{x} \in \mathbb{X}\,, \qquad \min_{\mathbf{u}\in\mathbb{U}} Q(\mathbf{x}, \mathbf{u}) = J(\mathbf{x}) \quad \forall \mathbf{x} \in \mathbb{X}\,.$$

In particular, with the definitions above, since $(\mathbf{x}_i, \mathbf{u}_i)$ comes as an optimal tuple in the dataset $\mathcal{D}$ we have:

$$Q(\mathbf{x}_i, \mathbf{u}_i) = J(\mathbf{x}_i)\,.$$

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

$$\begin{aligned}
\mathcal{T}^{\pi} J^{\lambda}_{\text{ub}}(\mathbf{x}) - J^{\lambda}_{\text{ub}}(\mathbf{x}) &= c(\mathbf{x}, \mathbf{u}_i) + \gamma J^{\lambda}_{\text{ub}}(\mathbf{x}') - J^{\lambda}_{\text{ub}}(\mathbf{x}) && (22)\\
&= Q(\mathbf{x}, \mathbf{u}_i) - \gamma J(\mathbf{x}') + \gamma J^{\lambda}_{\text{ub}}(\mathbf{x}') - J^{\lambda}_{\text{ub}}(\mathbf{x}) && (23)\\
&= Q(\mathbf{x}, \mathbf{u}_i) - \gamma J(\mathbf{x}') + \gamma J^{\lambda}_{\text{ub}}(\mathbf{x}') - \mathbf{J}_i - \lambda\|\mathbf{x} - \mathbf{x}_i\| && (24)\\
&\leq \mathbf{J}_i + L_J\|\mathbf{x} - \mathbf{x}_i\| + \gamma\Big(J^{\lambda}_{\text{ub}}(\mathbf{x}') - J(\mathbf{x}')\Big) - \mathbf{J}_i - \lambda\|\mathbf{x} - \mathbf{x}_i\| && (25)\\
&\leq (L_J - \lambda)\|\mathbf{x} - \mathbf{x}_i\| + \gamma\big(\mathbf{J}_j + \lambda\|\mathbf{x}' - \mathbf{x}_j\| - \mathbf{J}_j + L_J\|\mathbf{x}' - \mathbf{x}_j\|\big) && (26)\\
&= (L_J - \lambda)\|\mathbf{x} - \mathbf{x}_i\| + \gamma(\lambda + L_J)\|\mathbf{x}' - \mathbf{x}_j\| && (27)\\
&\leq (L_J - \lambda)\|\mathbf{x} - \mathbf{x}_i\| + \gamma(\lambda + L_J)L_f\|\mathbf{x} - \mathbf{x}_i\| && (28)\\
&\leq 0 \iff && (29)
\end{aligned}$$

$$\begin{aligned}
&\iff (L_J - \lambda) + \gamma(\lambda + L_J)L_f \leq 0 && (30)\\
&\iff L_J + \gamma L_J L_f \leq \lambda(1 - \gamma L_f) && (31)\\
&\iff \lambda \geq \frac{1 + \gamma L_f}{1 - \gamma L_f} L_J\ , && (32)
\end{aligned}$$

where the red identity follows from (21), the blue identity is the definition of $J^{\lambda}_{\text{ub}}(\cdot)$, the green inequality follows from an upper bound on $Q(\cdot,\cdot)$. Letting $\mathbf{x}_j = f(\mathbf{x}_i, \mathbf{u}_i)$, which belongs to $\mathcal{D}$ under the hypothesis that it contains trajectories, the second inequality follows from the definition of $J^{\lambda}_{\text{ub}}(\mathbf{x})$ and a lower bound on $J(\mathbf{x}')$, using $\mathbf{x}_j$ in the norms (a valid upper bound, even if index j does not minimize $J^{\lambda}_{\text{ub}}(\mathbf{x}')$). Then, the Lipschitz continuity of f:

$$\|\mathbf{x}' - \mathbf{x}_j\| = \|f(\mathbf{x}, \mathbf{u}_i) - f(\mathbf{x}_i, \mathbf{u}_i)\| \leq L_f\|\mathbf{x} - \mathbf{x}_i\|$$

gives the third inequality. ■

B.3. Theorem 2

Proof Note that the expression (8) we want to show is equivalent to:

$$(1 + \beta)\,(J(\mathbf{x}, \varepsilon) + \eta) \;\geq\; J^{\pi}(\mathbf{x}) + \eta. \quad \forall \mathbf{x} \in \mathbb{X}\,. \tag{33}$$

Recall as well (Def. 3) that $J^{\lambda}_{\text{lb}}(\cdot)$ is a lower bound of $J(\mathbf{x}, \varepsilon)$:

$$J(\mathbf{x}, \varepsilon) \geq J^{\lambda}_{\text{lb}}(\mathbf{x}) = \max_{1 \leq i \leq |\mathcal{D}|} \big\{\mathbf{J}_i + \eta - \lambda\|\mathbf{x} - \mathbf{x}_i\|\big\},$$

and that $J^{\lambda}_{\text{ub}}(\mathbf{x}) \geq J^{\pi}(\mathbf{x})$ since we are in the conditions of Theorem 1. Therefore, a sufficient condition for (33) is the inequality highlighted in red below.

$$(1 + \beta)\,(J(\mathbf{x}, \varepsilon) + \eta) \overset{\text{(Def. 3)}}{\geq} (1 + \beta)\left(J^{\lambda}_{\text{lb}}(\mathbf{x}) + \eta\right) \geq J^{\lambda}_{\text{ub}}(\mathbf{x}) + \eta \overset{\text{(Thm. 1)}}{\geq} J^{\pi}(\mathbf{x}) + \eta \tag{34}$$

Fix $\mathbf{x} \in \mathbb{X}$ and let $\mathbf{x}_i$ be the state in the dataset that maximizes $J^{\lambda}_{\text{lb}}(\mathbf{x})$. Note that we can further upper bound $J^{\lambda}_{\text{ub}}(\mathbf{x})$ using $\mathbf{x}_i$:

$$J^{\lambda}_{\text{ub}}(\mathbf{x}) = \min_{1 \le k \le |\mathcal{D}|} \{\mathbf{J}_k + \lambda \|\mathbf{x} - \mathbf{x}_k\|\} \le \mathbf{J}_i + \lambda \|\mathbf{x} - \mathbf{x}_i\| \, .$$

Our sufficient condition then becomes:

$$(1+\beta)\left(J^{\lambda}_{\text{lb}}(\mathbf{x}) + \eta\right) = (1+\beta)\left(\mathbf{J}_i + \eta - \lambda\|\mathbf{x} - \mathbf{x}_i\|\right) \ge \mathbf{J}_i + \lambda\|\mathbf{x} - \mathbf{x}_i\| + \eta \implies \tag{35}$$

$$\frac{\beta}{(2+\beta)\lambda}(\mathbf{J}_i + \eta) \ge \|\mathbf{x} - \mathbf{x}_i\| \, , \tag{36}$$

which is (8), completing the first half of our proof. To show (2), we have the following decomposition:

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

where (i) follows from the nonnegativity $\frac{J^{\pi}(\mathbf{x}) - J(\mathbf{x},\varepsilon)}{J(\mathbf{x},\varepsilon)+\eta} \ge 0$ and $\frac{J(\mathbf{x},\varepsilon)+\eta}{J(\mathbf{x})+\eta} \ge 1$. Using (8) yields (ii). For (iii), if ε is small enough such that $J(\mathbf{x},\varepsilon) + \eta - L\varepsilon > 0 \; \forall \mathbf{x} \in \mathbb{X}$, we invoke the first item of Assumption 4 to lower bound the denominator in $\frac{J(\mathbf{x},\varepsilon)+\eta}{J(\mathbf{x})+\eta}$. Rearranging terms, we finish the proof. ■

B.4. Proposition 4

Proof We split the proof in two parts.

1. A $2r-$cover of $\mathbb{X}$ yields a recursively feasible and $\beta-$optimal policy. The radius r in the proposition satisfies:

$$2r = \min\left\{\frac{\eta}{\lambda}\cdot\frac{\beta\eta - L\varepsilon(1+\beta)}{(2+\beta)\eta - L\varepsilon(1+\beta)}, \frac{\varepsilon}{L_f}\right\}.$$

First, note that if $\bigcup_{i=1}^{|\mathcal{D}|}\mathbb{B}(\mathbf{x}_i, 2r) \supseteq \mathbb{X}$, then policy $\pi_{\mathcal{D}}$ is recursively feasible: this is because $\forall\mathbf{x}, \exists\mathbf{x}_i : \|\mathbf{x}-\mathbf{x}_i\| \leq 2r \leq \frac{\varepsilon}{L_f}$ and we invoke the result of Proposition 3.

We now show that a $2r-$cover also implies that $\pi_{\mathcal{D}}$ is $\beta-$optimal. Recall the result of Theorem 2: if the dataset $\mathcal{D}$ has sufficient coverage, in the sense that for any $\mathbf{x}\in\mathbb{X}$ there exists $\mathbf{x}_i\in\mathcal{D}$ such that:

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

$$\implies \sup_{\mathbf{x}\in\mathbb{X}}\frac{J^\pi(\mathbf{x}) - J(\mathbf{x},\varepsilon)}{J(\mathbf{x},\varepsilon)+\eta} \leq \beta' \tag{38}$$

$$\implies \sup_{\mathbf{x}\in\mathbb{X}}\frac{J^\pi(\mathbf{x}) - J(\mathbf{x})}{J(\mathbf{x})+\eta} \leq \frac{\beta'\eta + L\varepsilon}{\eta - L\varepsilon} \tag{39}$$

where $\beta' > 0$ and $\eta > L\varepsilon$. We want to use the smallest β' such that the right-most term is upper-bounded by β, i.e.:

$$\frac{\beta'\eta + L\varepsilon}{\eta - L\varepsilon} \leq \beta \implies \beta' \leq \frac{\beta(\eta - L\varepsilon) - L\varepsilon}{\eta}. \tag{40}$$

We plug this value of β' back in (37) and upper bound the right hand side using the fact that $\mathbf{J}_i \geq 0$:

$$\|\mathbf{x}-\mathbf{x}_i\| \leq \frac{\beta'}{(2+\beta')\lambda}$$

that is to say:

$$\begin{aligned}\|\mathbf{x}-\mathbf{x}_i\| &\leq \frac{\frac{\beta(\eta-L\varepsilon)-L\varepsilon}{\eta}}{\lambda\left(2+\frac{\beta(\eta-L\varepsilon)-L\varepsilon}{\eta}\right)}\eta \\ &= \frac{(\beta(\eta-L\varepsilon)-L\varepsilon)\eta}{\lambda\left(2\eta+\beta(\eta-L\varepsilon)-L\varepsilon\right)}, &(41)\\ &= \frac{\eta}{\lambda}\cdot\frac{\beta\eta - L\varepsilon(1+\beta)}{(2+\beta)\eta - L\varepsilon(1+\beta)}. &(42)\end{aligned}$$

This finishes the first part of the proof.

2. Sample complexity of Algorithm 1 to achieve a 2r-cover of $\mathbb{X}$ In the prequel we showed that a $2r-$cover of $\mathbb{X}$ yields a recursively feasible and $\beta-$optimal policy. Now the question is: if we sample $\{\mathbf{x}_i\}_{i=1}^n$ uniformly from $\mathbb{X}$, what is the sample complexity of n to get a $2r-$cover?

Consider an arbitrary $r-$cover of $\mathbb{X}$, i.e. center points $\{\mathbf{y}_1, \dots, \mathbf{y}_m\}$ such that:

$$\bigcup_{k=1}^{m} \mathbb{B}(\mathbf{y}_k, r) \supseteq \mathbb{X} \,.$$

Such a cover exists because $\mathbb{X}$ is compact, and $m \approx \mathrm{N}_{\text{cover}}(\mathbb{X}; r)$. Let's call $B_k \triangleq \mathbb{B}(\mathbf{y}_k, r), k = 1, \dots m$. Observe that, if the following condition holds:

$$\forall B_k \; \exists \mathbf{x}_i \in \mathcal{D} : \mathbf{x}_i \in B_k \tag{43}$$

we can conclude

$$\|\mathbf{x} - \mathbf{x}_i\| \leq 2r \; \forall \mathbf{x} \in \mathbb{X},$$

that is to say: if there is at least one sample $\mathbf{x}_i$ *in every* ball B_k, then $\{\mathbf{x}_i\}_{i=1}^n$ form a $2r-$cover of $\mathbb{X}$. Indeed, by the triangle inequality:

$$\|\mathbf{x} - \mathbf{x}_i\| \leq \|\mathbf{x} - \mathbf{y}_k\| + \|\mathbf{x}_i - \mathbf{y}_k\| \leq 2r \tag{44}$$

Therefore, the sample complexity of Algorithm 1 is upper-bounded by the sample complexity of guaranteeing (43). Given samples $\mathbf{x}_i \overset{iid}{\sim} \mathrm{Unif}(\mathbb{X})$, observe that:

$$\mathbb{P}(\mathbf{x}_i \in B_k) \approx \frac{1}{m} = \frac{1}{\mathrm{N}_{\text{cover}}(\mathbb{X}; r)} \,.$$

Now, fix $k \in \{1, \dots, m\}$ and assume Algorithm 1 has been run for n rounds.

$$\mathbb{P}\left(\nexists i \in \{1, \dots, n\} : \mathbf{x}_i \in B_k\right) \lesssim \left(\frac{1}{\mathrm{N}_{\text{cover}}(\mathbb{X}; r)}\right)^n \leq \exp\left(-\frac{n}{\mathrm{N}_{\text{cover}}(\mathbb{X}; r)}\right) \tag{45}$$

We then apply a union bound over k and upper bound that probability by $\delta \in (0, 1)$:

$$\begin{aligned}
\mathbb{P}\left(\text{some ball } B_k \text{ not covered by any } \mathbf{x}_i\right) &= \mathbb{P}\left(\bigcup_{k=1}^{m} \{\nexists i \in \{1, \dots, n\} : \mathbf{x}_i \in B_k\}\right) && (46)\\
&\leq \sum_{i=1}^{m} \mathbb{P}\left(\nexists i \in \{1, \dots, n\} : \mathbf{x}_i \in B_k\right) && (47)\\
&\lesssim \mathrm{N}_{\text{cover}}(\mathbb{X}; r) \exp\left(-\frac{n}{\mathrm{N}_{\text{cover}}(\mathbb{X}; r)}\right) && (48)\\
&\leq \delta \implies && (49)
\end{aligned}$$

$$n \geq \mathrm{N}_{\text{cover}}(\mathbb{X}; r) \log\left(\mathrm{N}_{\text{cover}}(\mathbb{X}; r)\right) \log\left(\frac{1}{\delta}\right) \,, \tag{50}$$

as desired.

■

Appendix C. Considerations on Algorithm 1

Algorithm 1: Local Adaptive Data collector

Input: $\varepsilon > 0, T > 0$. 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

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 (Last equation in 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

    Run **Block 1** for $\lfloor \frac{N-K}{3^n} \rfloor$ cells

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

  end

end

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 1 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 1 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-optimal leaves 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.

Remark 3 (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 1. The Algorithm 1 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.*

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 $\mathcal{H}_{k,j}$.

Appendix D. Details on the experiments

D.1. Inverted Pendulum

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

$$\dot{\mathbf{x}} = \frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{g}{l}\sin x_1 \end{bmatrix} + \frac{1}{ml^2}\begin{bmatrix} 0 \\ 1 \end{bmatrix} u \ , \tag{51}$$

where $u \in \mathbb{R}$ is the scalar torque applied to the axis of the pendulum. We consider the discrete-time dynamics of (51) 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.

D.2. Minimum time with control regularization

$$\dot{\mathbf{x}} = \frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0.2 & 1 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \tag{52}$$

We consider the discrete-time version of (52) with sampling time $\delta t = 0.1s$, stage cost:

$$c(\mathbf{x}_t, \mathbf{u}_t) = 1 + 10 \cdot \|\mathbf{u}_t\| \ . \tag{53}$$

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 (52) 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, $\mathbb{U} = [-1, 1]$.

D.3. Verification on constrained LQR

$$A = \begin{bmatrix} 1 & 0.1 \\ 0 & 1 \end{bmatrix}; \qquad B = \begin{bmatrix} 0.15 \\ 1 \end{bmatrix}, \tag{54}$$

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.