Automatica 187 (2026) 112873

Contents lists available at ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

MPR: A novel randomized algorithm for multi-parametric programming☆

Alessandro Falsone [a,*], Federico Bianchi [b], Maria Prandini [a]

[a] Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Via Ponzio 34/5, 20133 Milano, Italy
[b] Dipartimento di Tecnologie di Generazione e Materiali, Ricerca sul Sistema Energetico (RSE) S.p.A., Via Rubattino 54, 20134 Milano, Italy

ARTICLE INFO

Article history:
Received 26 September 2024
Received in revised form 10 October 2025
Accepted 15 December 2025

Keywords:
Multi-parametric programming
Randomized algorithms
Explicit model predictive control

ABSTRACT

In this paper, we present a new paradigm to construct an optimal map of a multi-parametric quadratic or linear program, based on random sampling. Probabilistic guarantees for coverage of the region over which the map is constructed are provided in terms of user-defined coverage and confidence parameters. Extensive simulations show that the proposed Multi-Parametric Randomized algorithm (MPR) outperforms state-of-the-art competitors.

Notation. We denote with $\mathbb{R}$ the set of real numbers. Given a matrix $M \in \mathbb{R}^{n_r,n_c}$ with n_r rows and n_c columns we denote its transpose as $M^\top$. For a symmetric matrix $M = M^\top$, $M \succeq 0$ denotes that M is positive semi-definite. For a vector v, $\|v\|$ denotes its Euclidean norm. All inequalities between vectors or between a vector and a scalar are intended component-wise. Given a probability space $(\Sigma, \mathcal{F}, \mathbb{P})$, we denote with $\mathbb{P}(E)$ the probability measure of the event $E \in \mathcal{F}$. For a set $\mathbb{S} \subset \mathbb{R}^n$, vol($\mathbb{S}$) denotes its n-dimensional volume.

1. Introduction

We consider the following multi-parametric Quadratic Program (mpQP)

$$\min_z \quad \tfrac{1}{2} z^\top Q z + (F\vartheta + c)^\top z \qquad (\mathcal{P}_\vartheta)$$
$$\text{s.t:} \quad Gz \leq b + S\vartheta$$

where $z \in \mathbb{R}^{n_z}$ is the vector containing the decision variables, $\vartheta \in \mathbb{R}^{n_\vartheta}$ is the vector containing the parameters, $0 \preceq Q = Q^\top \in \mathbb{R}^{n_z,n_z}$, $F \in \mathbb{R}^{n_z,n_\vartheta}$, and $c \in \mathbb{R}^{n_z}$ define the cost function, and $G \in \mathbb{R}^{n_c,n_z}$, $b \in \mathbb{R}^{n_c}$, and $S \in \mathbb{R}^{n_c,n_\vartheta}$ define the n_c constraints. Note that $(\mathcal{P}_\vartheta)$ encompasses a multi-parametric Linear Program (mpLP) as a special case when $Q = 0$.

We will work under the standing assumption that the set $\Theta_f \subseteq \mathbb{R}^{n_\vartheta}$ of ϑ for which $(\mathcal{P}_\vartheta)$ is feasible is full dimensional, which guarantees that problem $(\mathcal{P}_\vartheta)$ admits an optimal map $z^\star : \Theta_f \to \mathbb{R}^{n_z}$ given by a PieceWise Affine (PWA) function defined over Θ_f and taking values in $\mathbb{R}^{n_z}$, i.e.,

$$z^\star(\vartheta) = K_i\vartheta + h_i, \qquad \vartheta \in \Theta_i, \qquad i = 1, \ldots, n_r, \qquad (1)$$

where the collection $\{\Theta_i\}_{i=1}^{n_r}$ is a polyhedral partition of Θ_f, which is also polyhedral, Jones and Morrari (2006). Each element Θ_i of the polyhedral partition is called *critical region* and is associated with a specific combination of active constraints (those satisfied with equality by $z^\star(\vartheta)$) that remains active for all $\vartheta \in \Theta_i$, Bemporad, Morari et al. (2002).

Problems of the form $(\mathcal{P}_\vartheta)$ arise in various contexts, including Model Predictive Control (MPC) for discrete-time linear systems. In MPC (see, e.g., Mayne (2014) and Morari and H. Lee (1999)), the control action to be applied at each time instant is computed by solving a finite-horizon optimal control problem with a cost function which is typically quadratic in the future state and control input and subject to linear constraints on the input and the state. By unrolling the system dynamics over the given horizon, one can express the state evolution as a function of the input sequence and the initial state. In this scenario, the input sequence is the decision vector, while the initial state can be regarded as a parameter, which is given by the system

☆ This work was supported by the PRIN 2022 project "The Scenario Approach for Control and Non-Convex Design" (project number D53D23001450006), by the PRIN PNRR project P2022NB77E "A data-driven cooperative framework for the management of distributed energy and water resources" (CUP: D53D23016100001), funded by the NextGeneration EU program (Mission 4, Component 2, Investment 1.1), and by the FAIR (Future Artificial Intelligence Research) project, funded by the NextGenerationEU program within the PNRR-PE-AI scheme (M4C2, Investment 1.3, Line on Artificial Intelligence). The material in this paper was not presented at any conference. This paper was recommended for publication in revised form by Associate Editor Martin Monnigmann under the direction of Editor Florian Dorfler.

* Corresponding author.
E-mail addresses: alessandro.falsone@polimi.it (A. Falsone), federico.bianchi@rse-web.it (F. Bianchi), maria.prandini@polimi.it (M. Prandini).

https://doi.org/10.1016/j.automatica.2026.112873

and influences cost and constraint of the finite-horizon optimal control problem. According to the receding horizon strategy, once an optimal input sequence is obtained, only the first input in the sequence is applied, then the finite-horizon optimal control problem is solved again starting from the new state to determine the next input, and the same procedure is repeated at every time step. This results in a state feedback control law, which can thus counteract modeling errors and disturbances, while accounting for constraints on both input and state variables.

Under this scheme, it would be very appealing to precompute the optimal map between any state and the corresponding optimal input to be applied at that state, to avoid having to solve an optimization problem at each time step (explicit MPC, Alessio and Bemporad (2009) and Bemporad, Borrelli et al. (2002)), which may be even infeasible in certain real-time applications or in case limited computation power is available. This is, in fact, possible since in standard MPC problems, the system's state (ϑ in $(\mathcal{P}_\vartheta)$) evolves within a full-dimensional region of the state space, despite input and state constraints. Thus the standing assumption on the feasibility region Θ_f is naturally satisfied, which ensures the existence of an optimal PWA map $z^\star$ in (1). Therefore, one only needs to measure the current state, determine its position within the polyhedral regions (*point location* problem), and evaluate the corresponding affine function.

Note that, although our driving motivation for studying problem $(\mathcal{P}_\vartheta)$ is the relationship between multi-parametric programming and explicit MPC, in this work we are *not* concerned with stability of the resulting MPC feedback scheme nor with any tweak that exploits the peculiarity of the MPC application. Our focus here is on the construction of an optimal map for a generic multi-parametric quadratic program, not necessarily associated to an MPC problem.

Several approaches have been proposed regarding efficient computation of an optimal map, see, e.g., Ahmadi-Moshkenani et al. (2018, 2016), Baotić (2002), Bemporad, Morari et al. (2002), Feller and Johansen (2013), Feller et al. (2013), Gupta et al. (2011), Herceg et al. (2015), Jones and Morrari (2006), Oberdieck et al. (2017), Spjøtvold et al. (2006) and Tøndel et al. (2003), and they can be divided into two main categories based on the way they explore the parameter space to construct an optimal map: geometric approaches and enumerative approaches.

Geometric approaches. The general idea behind a geometric approach is to perform the parameter space exploration starting from a suitably computed initial critical region, and then determine the remaining regions according to some exploration strategy.

This kind of approach has been originally proposed in Bemporad, Morari et al. (2002). Given a domain of exploration, the initial critical region is computed according to a two-steps procedure: in the first step, a feasible parameter is obtained by solving an LP and in the second step the active set of the optimal solution at the feasible parameter is retrieved by solving a QP. Exploration of the remaining parameter space proceeds recursively, taking as sub-domains of exploration the multiple polyhedral subsets composing the complement of the initial critical region. This approach works for a general mpQP but it has the drawback of introducing artificial cuts in the parameter space which do not necessarily match the polyhedral partition associated with the mpQP solution. Post-processing is then needed in order to merge those critical regions that are split into multiple sub-domains.

To overcome this issue, further methods were introduced to explore the parameter space more efficiently, Baotić (2002) and Tøndel et al. (2003). The idea is to obtain the active set of adjacent critical regions either by stepping outside a facet of the current region, Baotić (2002), or by directly inferring their active set from the kind of considered facets, Tøndel et al. (2003). Both methods rely however on the assumption that adjacent critical regions share a common facet, Tøndel et al. (2003). This property, called *facet-to-facet property*, does not hold for every mpQP partition, but whenever this property is violated, one can still resort to the method in Bemporad, Morari et al. (2002) in a reduced parameter space, Spjøtvold et al. (2006).

The method in Baotić (2002) is implemented in the Multi-Parametric Toolbox 2.0 (MPT2), Kvasnica et al. (2004), and a numerically more robust extended version based on reformulation of the mpQP as a parametric Linear Complementary Problem (pLCP) has been proposed in Jones and Morrari (2006) and implemented in the Multi-Parametric Toolbox 3.0 (MPT3), Herceg et al. (2013). The method in Tøndel et al. (2003), instead, has been implemented in the Hybrid Toolbox (HT), Bemporad (2004).

In general, as the dimensionality of the parameter space increases, geometric computations (e.g., computing the center of a lower-dimensional facet, Herceg et al. (2015)) become numerically sensitive, making these algorithms slower and less reliable.

Enumerative approaches. To overcome the need for a parameter space exploration step, the so called enumeration-based methods have been introduced, starting from the pioneer work of Gupta et al. (2011). These methods exploit the premise that corresponding to each parameter at which the quadratic program is feasible and some constraint qualification conditions hold, there exists a unique set of constraints which are active at the optimal solution. Thus, the enumeration of all possible sets of active constraints leads implicitly to the full exploration of the parameter space.

In Gupta et al. (2011), active sets are organized in a combinatorial tree, and each candidate is tested for optimality by solving an LP. This either results in a feasible solution, indicating a unique critical region where the active set is optimal, or an infeasible solution. In the case of infeasibility, this information can be used to prune (via another LP) other candidate sets, thus reducing the exploration. A computational analysis and comparison with the geometric method in Baotić (2002) is presented in Feller and Johansen (2013). The method in Gupta et al. (2011), proposed for strictly convex mpQPs only, has been extended to other problem classes. Specifically, Herceg et al. (2015) extends the approach to pLCPs, thus rendering it applicable to both linear and quadratic parametric programs. Also, various pruning and exploration strategies have been proposed to further enhance the method in Gupta et al. (2011). In Feller et al. (2013) the infeasibility of a candidate active set is tested by inspecting the vertices of a suitably defined polyhedron, which however grows exponentially with the number of constraints, parameters, and decision variables. In both Feller et al. (2013) and Gupta et al. (2011), feasible but non-optimal combinations of active constraints are never pruned. To address this, Ahmadi-Moshkenani et al. (2016) proposed exploring only those sets that are known to be potentially optimal. However, in case of degeneracy, still a geometric-based post-processing has to be invoked. Such a post-processing is instead not needed in Ahmadi-Moshkenani et al. (2018), where a method for handling degeneracy is introduced. This approach categorizes the different types of degeneracy that may occur on common facets and addresses each one specifically. However, it requires tracking all detected instances of degeneracy and, depending on the type, exploring all subsets or supersets of active constraints accordingly. In Gupta et al. (2011) and Mitze and Mönnigmann (2020) has been recently combined with dynamic programming techniques to reduce the number of candidate active sets that need to be evaluated for feasibility and optimality, but the approach is tailored to those mpQPs arising from finite-horizon optimal control problems.

The methods in Gupta et al. (2011) and Herceg et al. (2015) are both implemented in the Multi-Parametric Toolbox 3.0 (MPT3), Herceg et al. (2013).

A general drawback of enumerative approaches, even with effective pruning criteria, is that the number of possible combinations of active constraints grows exponentially with the number of constraints, so that exploring the tree becomes intensive.

It is also worth mentioning that, in Oberdieck et al. (2017), a connected graph approach for solving mpQPs combining geometric and combinatorial methods has been introduced. This approach, like Tøndel et al. (2003), identifies facet types of full-dimensional critical regions to infer active sets, which form nodes in the connected graph. The graph is then explored using the fathoming criterion from Gupta et al. (2011). However, identifying facet types remains challenging, particularly for mpQPs with many parameters.

Lastly, there are methods designed to reduce memory usage or simplify the online evaluation of the computed optimal piecewise map. A first category of methods minimizes complexity by reducing the number of polyhedral regions, for example eliminating those ones where the PWA function saturates (Kvasnica & Fikar, 2011). A second category focuses on efficient online evaluation of the precomputed PWA map, optimizing the search for the active region without explicitly storing the partition (Baotić et al., 2008). A third one leverages specialized PWA representations to approximate the optimal control law, including lattice PWAs (Xu et al., 2024), canonical PWAs (Bemporad et al., 2011), and convex lifting (Nguyen et al., 2017).

Contribution and outline. In this paper, we present a new exploration paradigm based on randomized sampling to construct an optimal map of a multi-parametric quadratic program, which does not rely on geometric properties of the partition nor enumerates all possible active sets. The proposed procedure comes with probabilistic guarantees of coverage of the region where the user wants to compute the solution, except for a user-selected fraction. If such a fraction is chosen small enough compared to the size of the critical regions, then, 100% coverage is achieved, otherwise we cannot guarantee full coverage. Differently from both geometric and enumerative approaches, the exploration process can be steered by the user and automatically prioritizes larger regions over smaller ones, resulting in broader coverage of the parameter space when there are constraints on runtime or on the maximum number of regions to be found. The proposed method compares favorably against state-of-the-art multi-parametric programming methods implemented in MPT3, both in computation time and parameter space coverage per number of regions.

The remainder of the paper is structured as follows. In Section 2, we present the main idea behind the proposed randomized approach and we discuss its differences with respect to other approaches in more detail. In Section 3 we introduce a variant of the procedure which is computationally more convenient and in Section 4 we apply the proposed approach to two optimal control problems and a benchmark library of mpQP problems. Finally, Section 5 concludes the paper.

2. Multi-parametric randomized algorithm

We propose a novel Multi-Parametric Randomized algorithm (MPR) to construct an optimal map $z^\star(\vartheta)$ on a polyhedral collection $\{\Theta_i\}_{i=1}^{n_r}$ over a user-provided set $\mathcal{D} \subseteq \mathbb{R}^{n_\vartheta}$. The set $\mathcal{D}$ should be chosen by the user based on the specific application and it should contain the portion of the feasibility set Θ_f over which the user is interested in computing an optimal map. The proposed randomized procedure is summarized in Algorithm 1 and described next.

The user has to provide a set $\mathcal{D}$ and a probability distribution $\mathbb{P}$ over the Borel space $(\mathcal{D}, \mathcal{B}(\mathcal{D}))$, jointly with a positive integer N. The procedure starts with an empty collection $\mathcal{C}$ (cf. Step 1) which will be populated by triplets, each containing a polyhedral region and the corresponding affine map coefficients of the optimal map $z^\star(\vartheta)$ over that region. At each iteration, a parameter vector $\bar\vartheta$ is sampled from the user-provided set $\mathcal{D}$ according to the user-provided probability distribution $\mathbb{P}$ (cf. Step 4). If $\mathcal{P}_{\bar\vartheta}$ is feasible (i.e., $\bar\vartheta \in \Theta_f$) and the sample is not contained in any of the regions inside $\mathcal{C}$ (cf. Step 5), then the (new) critical region Θ around $\bar\vartheta$ is computed along with the coefficients K and h of the optimal affine map[1] over Θ (cf. Steps 6–7), and the triplet (Θ, K, h) is added to $\mathcal{C}$ (cf. Step 8). The procedure stops whenever more than N *subsequent* samples are found to either make $(\mathcal{P}_\vartheta)$ infeasible or belong to some region within $\mathcal{C}$. This is achieved via the counter n_s which is initialized at zero (cf. Step 2), increased whenever a sample belongs to an already explored region in $\mathcal{C}$ or makes $(\mathcal{P}_\vartheta)$ infeasible (cf. Step 11), and reset to zero whenever a sample used to construct a new region is found (cf. Step 9).

MPR – Algorithm 1

Input: $(\mathcal{D}, \mathbb{P})$, N

1: $\mathcal{C} \leftarrow \emptyset$
2: $n_s \leftarrow 0$
3: **while** $n_s < N$ **do**
4: Sample a $\bar\vartheta$ from $\mathcal{D}$ according to $\mathbb{P}$
5: **if** $\bar\vartheta \in \Theta_f \wedge \bar\vartheta \notin \bigcup_{(\Theta,K,h)\in\mathcal{C}} \Theta$ **then**
6: Compute critical region Θ around $\bar\vartheta$
7: Compute affine map coefficients K and h
8: $\mathcal{C} \leftarrow \mathcal{C} \cup (\Theta, K, h)$
9: $n_s \leftarrow 0$
10: **else**
11: $n_s \leftarrow n_s + 1$
12: **end if**
13: **end while**

Output: $\mathcal{C}$

Intuitively, if N is large, then the probability of extracting a new sample from $\mathcal{D}$ (according to $\mathbb{P}$) for which $(\mathcal{P}_\vartheta)$ is feasible but does not belong to any region in $\mathcal{C}$ is expected to be low. However, this statement cannot be deterministic given the randomized nature of the algorithm. One can in fact be very unlucky and, for example, extract N samples that are within the same region in $\mathcal{C}$ thus halting the algorithm when yet the probability of the uncovered feasible area in $\mathcal{D}$ is high. However, this will happen with a low probability if N is large. The next theorem (whose proof is deferred to Appendix A) formalizes this intuition.

Theorem 1. *Select a coverage parameter $\varepsilon \in (0, 1)$ and a confidence parameter $\beta \in (0, 1)$. Let N in Algorithm 1 satisfy*

$$N \geq \frac{\log \beta - \log n_r}{\log(1-\varepsilon)}. \tag{2}$$

Then, with confidence at least $1 - \beta$, it holds that

$$\mathbb{P}(\{\vartheta \in \mathcal{D} : \ \vartheta \in \Theta_f \setminus \Theta_{\mathcal{C}}\}) \leq \varepsilon, \tag{3}$$

where $\Theta_{\mathcal{C}} = \bigcup_{(\Theta,K,h)\in\mathcal{C}} \Theta$ is the union of the regions within the collection $\mathcal{C}$ returned by Algorithm 1. □

Remark 1. It is worth noticing that one could alternatively fix N in Algorithm 1 and derive the probabilistic coverage guarantees associated to a given confidence level $1 - \beta$ by making the

[1] Computing the critical region and the map coefficients is standard, see, e.g., Pistikopoulos et al. (2020, Chapters 2 and 3).

inequality in (2) explicit with respect to ε, thus getting

$$\varepsilon \leq 1 - \sqrt[N]{\frac{\beta}{n_r}}. \tag{4}$$

Not surprisingly, as N grows to infinity, the coverage level $1-\varepsilon$ of $\Theta_f \cap \mathcal{D}$ tends to one. □

The next corollary is an immediate consequence of Theorem 1 and has an interesting practical implication.

Corollary 1. *Select $\varepsilon \in (0,1)$, $\beta \in (0,1)$ and let N be chosen according to Theorem 1. Then, if $\mathbb{P}$ is the uniform probability distribution over $\mathcal{D}$, with confidence at least $1-\beta$, it holds that*

$$\frac{\text{vol}((\Theta_f \setminus \Theta_{\mathcal{C}}) \cap \mathcal{D})}{\text{vol}(\mathcal{D})} \leq \varepsilon, \tag{5}$$

where $\Theta_{\mathcal{C}} = \bigcup_{(\Theta,K,h)\in\mathcal{C}} \Theta$ is the union of the regions within the collection $\mathcal{C}$ returned by Algorithm 1. □

Corollary 1 implies that if we draw many samples from $\mathcal{D}$ uniformly at random and each of them either renders $(\mathcal{P}_\vartheta)$ infeasible or falls within one of the explored regions in $\mathcal{C}$, then we have covered (almost) every part of $\mathcal{D}$. Furthermore, if $\mathcal{D}$ is a box (i.e., $\mathcal{D} = \{\vartheta \in \mathbb{R}^{n_\vartheta} : \vartheta_\ell \leq \vartheta \leq \vartheta_u\}$), then its volume is very easy to compute and the user can easily tune via ε the volume of the region $\Theta_f \setminus \Theta_{\mathcal{C}}$ within $\mathcal{D}$ left unexplored. If N is chosen large enough so that the corresponding ε is smaller than the volume of the smallest region relative to $\mathcal{D}$ (or if ε is directly selected this way), then, with probability $1-\beta$ the entire $\Theta_f \cap \mathcal{D}$ will be explored. Given the nice interpretation provided by Corollary 1, we advise to choose $\mathbb{P}$ as the uniform probability distribution over $\mathcal{D}$, whenever possible. Note also that if $\mathcal{D}$ is a box and $\mathbb{P}$ is uniform, then, drawing an n_ϑ dimensional sample from $\mathcal{D}$ is simple since it reduces to drawing n_ϑ scalar samples from the edges of the box, according to a uniform distribution. If the uniform distribution choice is not possible, e.g., because the set $\mathcal{D}$ is unbounded, then we advise to choose a $\mathbb{P}$ with more probability mass in those parts of the parameter space where the user wants a better coverage, to steer the exploration towards those parts.

The number $N \geq \frac{\log\beta - \log n_r}{\log(1-\varepsilon)}$ of subsequent samples that enters the stopping criterion in Step 3 is a function of the coverage parameter ε, the confidence parameter β, and the number n_r of critical regions partitioning Θ_f. As for the dependence on ε, coverage is costly, since N grows with the inverse of ε given that for small ε $\log(1-\varepsilon) \simeq -\varepsilon$. Confidence is instead cheap since β appears under the logarithm and, hence, can be chosen very low, e.g., $\beta = 10^{-6}$, without growing too much N while making the statement hold basically deterministically. In particular, if ε is (directly or as a result of selecting N) smaller than the measure according to $\mathbb{P}$ of any critical region in $\Theta_f \cap \mathcal{D}$ and β is very low, then the entire $\Theta_f \cap \mathcal{D}$ will be explored with practical certainty. As for n_r, the reader should note that, according to Bemporad, Morari et al. (2002, Section 4.4), the number of regions of the solution of an mpQP is given by the number of optimal combinations of active constraints, which depends on the number of optimization variables, the number of constraints, and the dimension of the parameter space. If the problem is non-degenerate,[2] then any optimal combination of active constraints is composed of at most n_z constraints. Moreover, the number of optimal combinations of at most n_z active constraints is clearly bounded by the number of (not necessarily optimal) combinations of at most n_z active constraints. Therefore, the number n_r of critical region partitioning Θ_f must satisfy

$$n_r \leq \sum_{i=0}^{n_z} \binom{n_c}{i} \leq 2^{n_c},$$

where the first inequality is true when the problem is non-degenerate and $n_z \leq n_c$, while the second inequality is always true and gives a worst-case value for n_r, which can always be used in (2) if no information on n_r is available. If, instead, the user has some prior knowledge (i.e., a bound) on n_r, this can be directly used in (2) without affecting the guarantees.

[2] Problem $(\mathcal{P}_\vartheta)$ is said to be (primal) degenerate if it has more than n_z active constraint at the optimal solution. For an extensive discussion on degeneracy see Pistikopoulos et al. (2020, Section 2 of Chapter 2).

MPR – Algorithm 2

Input: $(\mathcal{D}, \mathbb{P})$, N_s

1: $\mathcal{C} \leftarrow \emptyset$
2: **for** $j = 1, 2, \ldots, N_s$ **do**
3: Sample a $\bar{\vartheta}$ from $\mathcal{D}$ according to $\mathbb{P}$
4: **if** $\bar{\vartheta} \in \Theta_f \wedge \bar{\vartheta} \notin \bigcup_{(\Theta,K,h)\in\mathcal{C}} \Theta$ **then**
5: Compute critical region Θ around $\bar{\vartheta}$
6: Compute affine map coefficients K and h
7: $\mathcal{C} \leftarrow \mathcal{C} \cup (\Theta, K, h)$
8: **end if**
9: **end for**

Output: $\mathcal{C}$

2.1. MPR variant with a fixed number of samples

Suppose that we have some constraint on the total number of extractions from the set $\mathcal{D}$, e.g., because we have a limited amount of time to compute the solution. We can then run the variant of MPR summarized in Algorithm 2, where the while loop starting at Step 3 of Algorithm 1 is replaced with a for loop cycling through the N_s available samples (cf. Step 2 of Algorithm 2), and the counter n_s is discarded.

Proposition 1 (whose proof is presented in Appendix B) provides guarantees on the optimal map obtained by using Algorithm 2 in terms of coverage-confidence pair (ε, β). As expected, also in this case, if N_s tends to infinity, then the coverage level of $\Theta_f \cap \mathcal{D}$ tends to one.

Proposition 1. *Select a confidence parameter $\beta \in (0,1)$. Set $\bar{N} = \left\lfloor \frac{N_s}{n_r+1} \right\rfloor$, where N_s is the number of samples processed in Algorithm 2. Then, with confidence at least $1-\beta$, it holds that*

$$\mathbb{P}(\{\vartheta \in \mathcal{D} : \vartheta \in \Theta_f \setminus \Theta_{\mathcal{C}}\}) \leq \varepsilon \leq 1 - \sqrt[\bar{N}]{\frac{\beta}{n_r}}, \tag{6}$$

where $\Theta_{\mathcal{C}} = \bigcup_{(\Theta,K,h)\in\mathcal{C}} \Theta$ is the union of the regions within the collection $\mathcal{C}$ returned by Algorithm 2. □

2.2. Comparison with other approaches

The proposed approach differs from the alternative ones in the literature with respect to the following three aspects: exploration strategy, covered region, and coverage guarantees.

Exploration strategy: In the proposed approach, the exploration of the critical regions is driven by the sampling process and, hence, depends on the selected probability distribution $\mathbb{P}$ over the Borel space on $\mathcal{D}$. A region with a large probability is much more likely to be discovered at an earlier stage of the algorithm, so that

if the algorithm is prematurely stopped, the returned collection $\mathcal{C}$ will likely contain that region. The user can thus guide the exploration of $\mathcal{D}$ leveraging $\mathbb{P}$. If $\mathbb{P}$ is the uniform distribution, then the probability measure of a region is directly proportional to its volume and the algorithm will favor the exploration of larger critical regions at earlier stages instead of smaller ones. If one wants a better coverage in certain regions, then, the probability mass should be concentrated on them. In contrast, existing algorithms are bound to an exploration strategy driven by some notion of vicinity: spatial closeness for geometric approaches and active set similarity for enumerative approaches, which require some computational effort.

Covered region: An advantage of the proposed approach is that the explored regions within $\mathcal{C}$ can extend also outside the user-provided set $\mathcal{D}$. This is unlike the geometric approaches, which clips all regions to $\mathcal{D}$ and use this information to determine when the algorithm can be terminated. However, in MPC, there are situations where $\mathcal{D}$ is heuristically selected by the user but the state of the dynamical system (ϑ in $(\mathcal{P}_\vartheta)$) can accidentally evolve outside $\mathcal{D}$ while still being in Θ_f. In these cases, having an optimal map that extends outside $\mathcal{D}$ enables us to apply the optimal control input, whereas the optimal map returned by other approaches would be simply undefined outside $\mathcal{D}$. This limitation is, of course, not critical and can be circumvented by suitably choosing a larger $\mathcal{D}$, but in our case this is done automatically as a byproduct of the exploration strategy.

Coverage guarantees: As a downside, our approach does not deterministically guarantee that the collection of regions in $\mathcal{C}$ will cover all of $\Theta_f \cap \mathcal{D}$, whereas other approaches do. This drawback can be partly overcome by assuming a uniform distribution and setting the coverage parameter ε and the confidence parameter β small enough so as to guarantee full coverage with practical certainty, or by selecting a very large N (or N_s) so that Algorithm 1 (or 2, respectively) would extract enough samples to practically cover the entire $\Theta_f \cap \mathcal{D}$.

3. Checking infeasibility

Note that Step 5 in Algorithm 1 and Step 4 in Algorithm 2 involve checking whether $\bar{\vartheta} \in \Theta_f$. This check is fairly easy if an hyperplane representation of the polyhedral set Θ_f is available, and it can even be skipped if $\mathcal{D}$ is known to be contained in Θ_f. In all other cases, it amounts to verify if problem $(\mathcal{P}_\vartheta)$ is feasible for $\vartheta = \bar{\vartheta}$, which can be computationally expensive, thus ultimately slowing down the entire procedure if the probability of drawing a sample outside Θ_f is high.

Motivated by this observation, we next propose in Algorithm 3 a variant of Algorithm 1. The same can be done for Algorithm 2, but to ease the presentation we discuss the variant of Algorithm 1 first and then comment on how to obtain the corresponding variant of Algorithm 2 at the end of this section.

Algorithm 3 incorporates some additional steps with respect to Algorithm 1 to outer-approximate Θ_f with a polyhedral set $\hat{\Theta}_f$. The approximation gets tighter and tighter as the algorithm progresses and provides a sufficient condition for $\bar{\vartheta} \notin \Theta_f$. The equivalence between Algorithm 1 and its variant in Algorithm 3 is discussed next.

At the beginning, $\hat{\Theta}_f$ is set equal to $\mathbb{R}^{n_\vartheta}$ (cf. Step 2), which clearly contains Θ_f. Then, at each iteration, whenever a point $\bar{\vartheta}$ is extracted from $\hat{\Theta}_f \setminus \Theta_f$ the else clause is executed: its projection $\hat{\vartheta}$ onto the set Θ_f is first computed (cf. Step 14) and then used to refine the approximation $\hat{\Theta}_f$ via a separating hyperplane between $\bar{\vartheta}$ and Θ_f (cf. Step 15). Computing $\hat{\vartheta}$ amounts to solving the optimization program

$$\hat{\vartheta} = \Pi_{\Theta_f}(\bar{\vartheta}) = \arg\min_{\vartheta \in \Theta_f} \tfrac{1}{2}\|\vartheta - \bar{\vartheta}\|^2, \tag{7}$$

MPR – Algorithm 3

Input: $(\mathcal{D}, \mathbb{P})$, N

1: $\mathcal{C} \leftarrow \emptyset$
2: $\hat{\Theta}_f \leftarrow \mathbb{R}^{n_\vartheta}$
3: $n_s \leftarrow 0$
4: **while** $n_s < N$ **do**
5: Sample a $\bar{\vartheta}$ from $\mathcal{D}$ according to $\mathbb{P}$
6: **if** $\bar{\vartheta} \notin \hat{\Theta}_f \vee \bar{\vartheta} \in \bigcup_{(\Theta,K,h)\in\mathcal{C}} \Theta$ **then**
7: $n_s \leftarrow n_s + 1$
8: **else if** $\mathcal{P}_{\bar{\vartheta}}$ is feasible **then**
9: Compute critical region Θ around $\bar{\vartheta}$
10: Compute affine map coefficients K and h
11: $\mathcal{C} \leftarrow \mathcal{C} \cup (\Theta, K, h)$
12: $n_s \leftarrow 0$
13: **else**
14: $\hat{\vartheta} = \Pi_{\Theta_f}(\bar{\vartheta})$
15: $\hat{\Theta}_f \leftarrow \hat{\Theta}_f \cap \{\vartheta \in \mathbb{R}^{n_\vartheta} : (\bar{\vartheta} - \hat{\vartheta})^\top(\vartheta - \hat{\vartheta}) \le 0\}$
16: $n_s \leftarrow n_s + 1$
17: **end if**
18: **end while**

Output: $\mathcal{C}$, $\hat{\Theta}_f$

where we can use the fact that

$$\Theta_f = \{\vartheta \in \mathbb{R}^{n_\vartheta} : \exists z \in \mathbb{R}^{n_z} : Gz \le b + S\vartheta\} \tag{8}$$

to equivalently implement Step 14 as

$$(\hat{\vartheta}, \hat{z}) \in \arg\min_{\vartheta, z} \ \tfrac{1}{2}\|\vartheta - \bar{\vartheta}\|^2 \quad \text{s.t:} \ \ Gz \le b + S\vartheta, \tag{9}$$

where $\hat{z}$ is an instrumental variable and can be discarded.[3]

To see that Steps 14–15 result in an outer-approximation it is sufficient to recall that, by the projection theorem (see, e.g., Bertsekas (2015, p. 471)), any triplet $(\bar{\vartheta}, \hat{\vartheta}, \vartheta)$ such that $\bar{\vartheta} \notin \Theta_f$, $\hat{\vartheta} = \Pi_{\Theta_f}(\bar{\vartheta})$, and $\vartheta \in \Theta_f$, satisfy

$$(\bar{\vartheta} - \hat{\vartheta})^\top(\vartheta - \hat{\vartheta}) \le 0, \tag{10}$$

that is precisely the inequality added to $\hat{\Theta}_f$ and is satisfied by any $\vartheta \in \Theta_f$, implying $\Theta_f \subseteq \hat{\Theta}_f$ at each step. Moreover, since $\bar{\vartheta} \notin \Theta_f$ and $\hat{\vartheta} \in \Theta_f$, we have $\hat{\vartheta} \neq \bar{\vartheta}$, so that $(\bar{\vartheta} - \hat{\vartheta})^\top(\bar{\vartheta} - \hat{\vartheta}) = \|\bar{\vartheta} - \hat{\vartheta}\|^2 > 0$, which shows that $\bar{\vartheta}$ violates constraint (10), and, hence, Step 15 strictly reduces $\hat{\Theta}_f$, thus tightening the outer-approximation of Θ_f every time it is executed.

Now that we have established that $\Theta_f \subseteq \hat{\Theta}_f$, it is also easy to show by induction that Algorithms 1 and 3 returns the same collection $\mathcal{C}$, which motivates why they have the same name. To ease the exposition let us recall the definition of $\Theta_\mathcal{C} = \bigcup_{(\Theta,K,h)\in\mathcal{C}} \Theta$ to denote the union of the regions inside $\mathcal{C}$. Algorithms 1 and 3 are clearly initialized with the same values of $\mathcal{C}$ and n_s. Now let us assume that they have the same $\mathcal{C}$ and n_s at the beginning of one iteration and assume also that they extract the same parameter value $\bar{\vartheta}$. Since $\Theta_\mathcal{C} \subseteq \Theta_f \subseteq \hat{\Theta}_f$, we have four cases:

[3] Note that the inclusion is used in (9) because $\hat{z}$ is not necessarily unique, whereas $\hat{\vartheta}$ is always unique due to strict convexity of the objective function with respect to ϑ.

(i) if $\bar{\vartheta} \in \Theta_{\mathcal{C}}$, then the check in Step 5 in Algorithm 1 fails, the check in Step 6 in Algorithm 3 passes, and both algorithms increase n_s by one;
(ii) if $\bar{\vartheta} \in \Theta_f \setminus \Theta_{\mathcal{C}}$, then the check in Step 5 in Algorithm 1 passes, the check in Step 6 in Algorithm 3 fails but the check in Step 8 in Algorithm 3 passes, and both algorithms compute the same new critical region, update $\mathcal{C}$ in the same way and both reset n_s to zero;
(iii) if $\bar{\vartheta} \in \hat{\Theta}_f \setminus \Theta_f$, then the check in Step 5 in Algorithm 1 fails, the checks in Step 6 and Step 8 of Algorithm 3 both fails, and both algorithms increase n_s by one (Algorithm 3 updating $\hat{\Theta}_f$ is irrelevant);
(iv) if $\bar{\vartheta} \notin \hat{\Theta}_f$, then the check in Step 5 in Algorithm 1 fails, the check in Step 6 in Algorithm 3 passes, and both algorithms increase n_s by one.

This means that at the beginning of the next iteration the two algorithms will have the same values of $\mathcal{C}$ and n_s, which proves the equivalence in terms of collection $\mathcal{C}$ returned, if the sequence of parameter values extracted are the same. All comments made for Algorithm 1 thus remains valid also for Algorithm 3 and the previous discussion readily proves the following result.

Corollary 2. *Select a coverage parameter $\varepsilon \in (0, 1)$ and a confidence parameter $\beta \in (0, 1)$. If*

$$N \geq \frac{\log \beta - \log n_r}{\log(1-\varepsilon)}, \tag{11}$$

then, with confidence at least $1-\beta$,

$$\mathbb{P}(\{\vartheta \in \mathcal{D} : \ \vartheta \in \Theta_f \setminus \Theta_{\mathcal{C}}\}) \leq \varepsilon, \tag{12}$$

and, if we further assume that $\mathbb{P}$ is the uniform probability distribution over $\mathcal{D}$, then

$$\frac{\mathrm{vol}((\Theta_f \setminus \Theta_{\mathcal{C}}) \cap \mathcal{D})}{\mathrm{vol}(\mathcal{D})} \leq \varepsilon, \tag{13}$$

where $\Theta_{\mathcal{C}} = \bigcup_{(\Theta,K,h)\in\mathcal{C}} \Theta$ is the union of the regions within the collection $\mathcal{C}$ returned by Algorithm 3. □

As an additional comment we shall stress that while in those iterations in which $\bar{\vartheta} \in \hat{\Theta}_f \setminus \Theta_f$, Algorithm 3 solves two optimization problems because it tries to solve $\mathcal{P}_{\bar{\vartheta}}$ in Step 8 first (which is however infeasible) and then it needs to solve (9) for the projection in Step 14, the computational benefits of being able to quickly discard samples outside $\hat{\Theta}_f$ at a later stage vastly outweigh the computational cost of the (fewer) additional optimization problems.

The same approximation scheme can also be introduced in Algorithm 2, resulting in a variant of Algorithm 3 with a fixed number of samples. To obtain this new variant, it is sufficient to replace the condition of the while loop starting at step 4 of Algorithm 3 with a for loop cycling over the samples available for the computation of the optimal map, and to discard the counter n_s together with the related steps.

Finally, we shall admit that the proposed approach requires performing a point location step for the parameter sample $\bar{\vartheta}$ at each iteration (cf. Step 5 of Algorithm 1 and Step 6 of Algorithm 3), which becomes increasingly demanding as the number of regions in $\mathcal{C}$ grows. While this is partly alleviated in Algorithm 3, where samples outside $\hat{\Theta}_f$ are quickly discarded, avoiding the point location step represents an interesting aspect for future improvement. We will see in the numerical example section that, despite the point location step, Algorithm 3 is almost always faster than the competitors.

4. Numerical tests

In this section we explore the performance of MPR on different examples. We first consider two mpQPs, resulting from finite-horizon optimal control problems, the first with a simpler and the second with a more complex partition induced by their optimal state-input map, and, then, a benchmark with 100 mpQP problems.

In all tests, we ran the MPR Algorithm 3 in MATLAB R2019a and compared it against the pLCP method first introduced in Jones and Morrari (2006) and implemented as default solver in MPT3 (Herceg et al., 2013) for MATLAB. Both methods use CPLEX 12.9 as a solver for the QPs. Simulations are carried out on a laptop with 16 GB of RAM and an i7-8565U CPU. MPT3 also implements the enumerative approaches in Gupta et al. (2011) and Herceg et al. (2015), but they exhibit inferior performance with respect to the default pLCP method used by MPT3 on the two finite-horizon optimal control problems we studied. Moreover, the MPT3 implementation of Herceg et al. (2015) does not provide an option to stop the algorithm after constructing a certain number of regions, which is instead crucial in our benchmark analysis. We therefore decided to present their results on the two control-based examples but not to compare against (Gupta et al., 2011; Herceg et al., 2015) in the benchmark test.

As for the MPR algorithm, we adopted a uniform probability distribution $\mathbb{P}$ over $\mathcal{D}$ in light of its ease of interpretation, we set $\varepsilon = 10^{-3}$, $\beta = 10^{-6}$, so as to cover at least $1-\varepsilon = 99.9\%$ of the volume of $\mathcal{D}$ with probability at least $1-\beta = 99.9999\%$, and we used 2^{n_c} as upper bound on n_r, so as to have a worst-case performance reference. Accordingly, we then set

$$N = \left\lceil \frac{\log \beta - n_c \log 2}{\log(1-\varepsilon)} \right\rceil.$$

As for the MPT3 implementations of Gupta et al. (2011), Herceg et al. (2015) and Jones and Morrari (2006), we used the default parameters. From now on, we will refer to the MPT3 implementation of Jones and Morrari (2006) simply as MPT3 and the MPT3 implementation of Gupta et al. (2011) and Herceg et al. (2015) simply as Gupta et al. (2011) and Herceg et al. (2015), respectively.

All volume measurements reported are approximated using 10^5 samples drawn uniformly over $\mathcal{D}$. We do not specify $\mathcal{D}$ here because it is different for the different examples.

4.1. MPC control of a linear dynamical system – 1

Our first example is the MPC control of a single-input single-output discrete-time double integrator dynamical system subject to input constraints. The example is taken from Falsone et al. (2023) and is a slightly modified version of Bemporad, Morari et al. (2002, Section 7.3), which we briefly recall here for completeness.

Let us consider the following discrete-time dynamical system

$$x_{k+1} = \underbrace{\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}}_{A} x_k + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{B} u_k, \tag{14}$$

with state $x_k \in \mathbb{R}^2$ and input $u_k \in \mathbb{R}$, k being the discrete-time index. The control objective is to regulate the system to the origin of the state space subject to constraints on the actuation capability $u_k \in [-1, 1]$ and constraints on the second state component $x_k^{[2]}$, which should lie within the interval $[-1, 1]$. To this end we

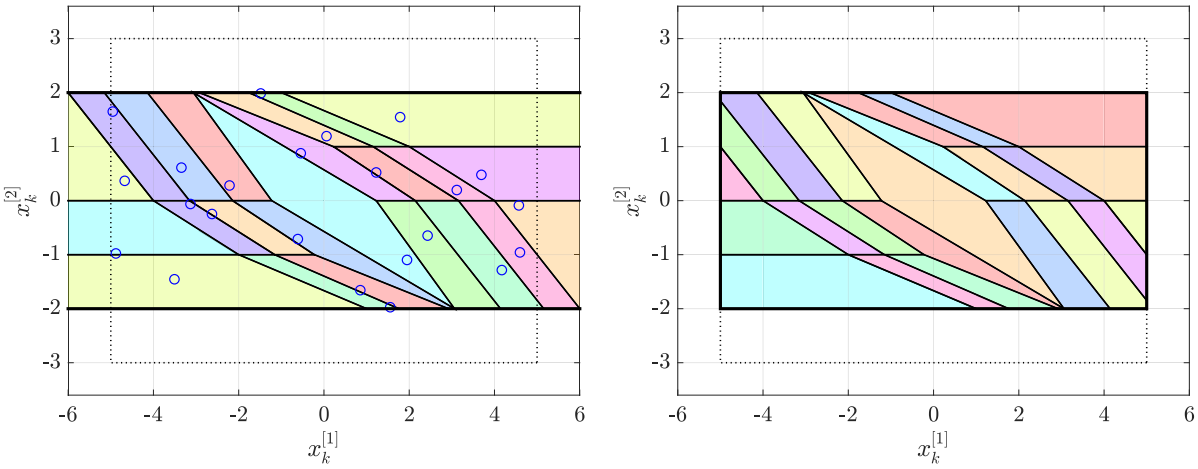

Fig. 1. Partition induced by the optimal map $z^\star$ associated with mpQP (15). MPR (left) and MPT3 (right). Colored areas represent different regions of the partition. Thick black lines denote the boundary of $\hat{\Theta}_f$ (for MPR) or $\Theta_f \cap \mathcal{D}$ (for MPT3). The dotted rectangle represents $\mathcal{D}$. The blue circles represents the samples of ϑ used by MPR to construct the regions. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

formulate the following finite-horizon optimal control problem parametric in x_k

$$\min_{\{u_t, x_t\}_t} \sum_{t=k}^{k+T-1} {x_{t+1}}^\top H x_{t+1} + {u_t}^\top R u_t \tag{15}$$

$$\begin{aligned} \text{s.t:} \quad & x_{t+1} = Ax_t + Bu_t && t = k, \dots, k+T-1 \\ & -1 \le u_t \le 1 && t = k, \dots, k+T-1 \\ & -1 \le x_t^{[2]} \le 1 && t = k+1, \dots, k+T, \end{aligned}$$

with $H = \text{diag}([1\ \ 0])$, $R = 0.1$, and $T = 5$ (set according to Bemporad, Morari et al. (2002)).

The problem can be easily reduced to the structure of $(\mathcal{P}_\vartheta)$ using the dynamic equation (14) to express each x_t as a function of x_k and u_s, $s = k, \dots, t-1$, for each $t = k, \dots, k+T$. In particular $z = [u_k \ \cdots \ u_{k+T-1}]^\top \in \mathbb{R}^T$, $\vartheta = x_k \in \mathbb{R}^2$, and G, S, and b model the actuation constraints $u_t \in [-1, 1]$, $t = k, \dots, k+T-1$ as well as the state constraints $x_t^{[2]} \in [-1, 1]$, $t = k+1, \dots, k+T$. We computed the optimal map of mpQP (15) over the box $\mathcal{D} = [-5, 5] \times [-3, 3]$ both with MPR and with MPT3. The results are reported in Fig. 1. In this example, both MPR and MPT3 returned a (complete) collection of 23 regions, covering all of $\Theta_f \cap \mathcal{D}$. MPR took 0.54 s while MPT3 took 1.66 s. As can be seen from Fig. 1 the regions returned by MPR (left) extend beyond the boundaries of $\mathcal{D}$ (dotted rectangle), whereas those returned by MPT3 (right) are clipped to $\mathcal{D}$.

Since MPR is a randomized algorithm, its performance may vary depending of the specific sequence of parameter values extracted. We thus also performed 100 runs of MPR varying the sequence of parameter values extracted while leaving all the rest unchanged. In all 100 runs MPR returned all 23 regions and took an average of 0.51 s with a standard deviation of 0.029 s.

As for the enumerative approaches, both Gupta et al. (2011) and Herceg et al. (2015) returned a collection of 29 regions (more than 23 because 6 of them are overlapping) after 22.5 and 13 s, respectively. Apart from the fact that they returned more regions than necessary, their running times are well above those of MPR and MPT3.

4.2. MPC control of a linear dynamical system – 2

The second control-oriented example is taken from Chen et al. (2018) and is presented next.

Let us consider the following discrete-time dynamical system

$$x_{k+1} = \underbrace{\begin{bmatrix} 1 & 0 \\ \alpha & 1 \end{bmatrix}}_{A} x_k + \underbrace{\begin{bmatrix} \alpha \\ \frac{1}{2}\alpha^2 \end{bmatrix}}_{B} u_k, \tag{16}$$

with state $x_k \in \mathbb{R}^2$ and input $u_k \in \mathbb{R}$, k being the discrete-time index, and $\alpha = 0.1$. The control objective is again to drive the system to the origin of the state space while satisfying constraints on the input $u_k \in [-2, 2]$ and constraints on the state $x_k \in [-1, 1] \times [-6, 6]$. To this end we formulate the following finite-horizon optimal control problem parametric in x_k

$$\min_{\{u_t, x_t\}_t} \sum_{t=k}^{k+T-1} {x_{t+1}}^\top H x_{t+1} + {u_t}^\top R u_t \tag{17}$$

$$\begin{aligned} \text{s.t:} \quad & x_{t+1} = Ax_t + Bu_t && t = k, \dots, k+T-1 \\ & -2 \le u_t \le 2 && t = k, \dots, k+T-1 \\ & -1 \le x_t^{[1]} \le 1 && t = k+1, \dots, k+T, \\ & -6 \le x_t^{[2]} \le 6 && t = k+1, \dots, k+T, \end{aligned}$$

with $H = \text{diag}([1\ \ 1])$, $R = 1$, and $T = 10$. Similarly to the previous example, one can easily show that (17) fits the structure of $(\mathcal{P}_\vartheta)$. We computed the optimal map of mpQP (17) over the box $\mathcal{D} = [-1, 1] \times [-6, 6]$ both with MPR and with MPT3.

In this example, over 100 runs, MPR returned 82 ± 2.64 regions after 2.38 ± 0.19 s (mean $\pm$ standard deviation) and MPT3 returned 109 regions after 6.31 s. Even if in this case MPR returned (on average) only 75% of the regions with respect to MPT3, the fraction of the volume of $\mathcal{D}$ left unexplored by MPR is $\text{vol}((\Theta_f \setminus \Theta_C) \cap \mathcal{D})/\text{vol}(\mathcal{D}) \approx 2.65 \cdot 10^{-5} \pm 2.43 \cdot 10^{-5}$, with a maximum of $1.1 \cdot 10^{-4}$ over all runs, which is (way) less then the selected coverage parameter $\varepsilon = 10^{-3}$, as guaranteed by Corollary 2. MPR thus covered almost all $\Theta_f \cap \mathcal{D}$ in less than half the time required by MPT3.

As for the enumerative approaches, Gupta et al. (2011) took 1 min to process the first 15 depth levels of the 64-level tree storing the active sets to be explored, while Herceg et al. (2015) took 1 min and 50 s to process the first 3 depth levels of the 10-level tree storing the active sets to be explored. We stopped both Gupta et al. (2011) and Herceg et al. (2015) prematurely as their timings were already well above those of MPR and MPT3 also for this example.

To show the different exploration strategy pursued by MPR and MPT3, we also ran the two algorithms, again for problem (17),

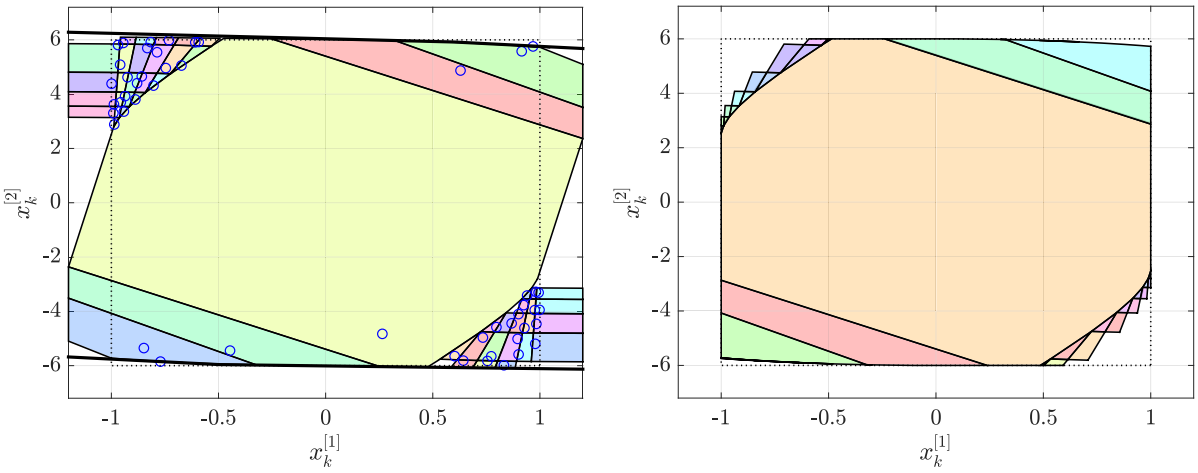

Fig. 2. Partition induced by the optimal map $z^\star$ associated with mpQP (17). MPR (left) and MPT3 (right). Colored areas represent different regions of the partition. Thick black lines denote the boundary of $\hat{\Theta}_f$ (for MPR). The dotted rectangle represents $\mathcal{D}$. The blue circles represents the samples of ϑ used by MPR to construct the regions. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

but with a limit of 50 as the maximum number of regions. Over 100 runs MPR always returned 50 regions after 1.2 ± 0.06 s (mean $\pm$ standard deviation) and MPT3 returned 50 regions after 4.07 s. The results of the first run are reported in Fig. 2. Clearly, in this case, MPR and MPT3 do not cover $\Theta_f \cap \mathcal{D}$ as they are both stopped prematurely. However, as can be seen from Fig. 2, the regions returned by MPR (left) extend beyond the boundaries of $\mathcal{D}$ and cover almost all $\Theta_f \cap \mathcal{D}$ with just a few gaps in the bottom-right corner, whereas the regions returned by MPT3 (right) leave the upper-left and bottom-right corners almost completely uncovered. This is because there are many small regions in the bottom-left corner (not visible in the picture) that MPT3 is forced to explore while MPR skips in favor of bigger regions. This statement is also supported by the fraction of volume of $\mathcal{D}$ left unexplored:

$$\textbf{MPR}: \ \frac{\text{vol}((\Theta_f \setminus \Theta_{\mathcal{C}}) \cap \mathcal{D})}{\text{vol}(\mathcal{D})} \approx 4.1 \cdot 10^{-3} \pm 0.94 \cdot 10^{-3},$$

$$\textbf{MPT3}: \ \frac{\text{vol}((\Theta_f \setminus \Theta_{\text{MPT3}}) \cap \mathcal{D})}{\text{vol}(\mathcal{D})} \approx 39.8 \cdot 10^{-3},$$

where Θ_{MPT3} denotes the union of the regions returned by MPT3 and we report mean $\pm$ standard deviation over the 100 runs for MPR, with MPR outperforming MPT3 by almost one order of magnitude in terms of coverage. In this case, the fraction of volume of $\mathcal{D}$ left unexplored by MPR is bigger than ε because it was stopped prematurely.

As for the enumerative approaches, Gupta et al. (2011) took 1.5 s but returned only 4 regions despite the maximum number of regions was set to 50, while we could not enforce any limit on the number of regions for Herceg et al. (2015).

4.3. Benchmark

Lastly, we tested the proposed approach on the POP-mpQP1 test set available at Oberdieck et al. (2016). It is a library of 100 mpQP problems with varying number of decision variables $n_z \in [1, 8]$, constraints $n_c \in [8, 70]$, and parameters $n_\vartheta \in [1, 8]$. Each problem in the library also contains its own bounding box for ϑ, which we used as $\mathcal{D}$. To minimize the chance to run into numerical issues, for each problem in the library, we normalize the cost function coefficients with respect to the maximum entry among Q, F, and c and we normalize constraints coefficients with respect to the maximum entry among G and S.

Since the timings of MPR in the previous examples exhibited a low dispersion, we performed a single run of MPR for each problem in the library. Moreover, given the low performance achieved by Gupta et al. (2011) and Herceg et al. (2015) in the previous examples, we here compare MPR with MPT3 only.

We fixed a maximum of 150 regions both for MPR and MPT3. This entails that the probabilistic statement in Corollary 2 holds only when MPR ends with less than 150 regions. The maximum value of 150 was selected to divide the dataset in half and gather statistics both when the algorithms exited normally and when they stopped prematurely.

As for MPT3, in 51 tests it exited because it reached the maximum allowed number of regions,[4] in 48 tests MPT3 successfully exited returning the complete optimal map, and in 1 test (Test #7) MPT3 deemed the problem infeasible and returned nothing.

In Fig. 3 we report, on a logarithmic scale, the execution times (in seconds) for the POP-mpQP1 test set. Each circle corresponds to one problem in the library with its horizontal coordinate reporting the time needed by MPT3 to run on that problem and its vertical coordinate reporting the time needed by MPR. As can be seen from the picture, almost all circles are below the diagonal, meaning that MPR is faster than MPT3, with only 3 exceptions: Test #3 and Test #60 for which MPR took slightly longer than MPT3 (1.06 s vs. 1.03 s and 21.3 s vs. 19.8 s, respectively) and Test #7 for which MPR took 10.06 s while MPT3 took 1.01 s, but MPT3 deemed the problem infeasible and readily exited. For most tests MPR is within one order of magnitude faster than MPT3 and for some instances its more than two order of magnitudes faster.

It is also interesting to look at the number of regions returned by the two algorithms. Fig. 4 reports, on a logarithmic scale, the number of regions found by the two algorithms for the POP-mpQP1 test set. Each circle corresponds to one problem in the library with its horizontal coordinate reporting the number of regions returned by MPT3 and its vertical coordinate reporting the number of regions returned by MPR. As we can see from the picture, all points fall onto or below the diagonal. This is expected because when MPT3 exits normally, it covers the entire $\Theta_f \cap \mathcal{D}$, it thus find the true number of regions n_r, and MPR cannot discover more regions, while when MPT3 stops prematurely because it reaches the maximum number of regions MPR cannot do better

[4] Actually, in 4 tests MPT3 exited saying that it reached the maximum number of regions but returned 149 (Tests #47, #51, and #76) or 148 (Test #91) regions. We believe that one or two regions have been merged by MPT3 after a post-processing phase and we treated those tests as if MPT3 had returned 150 regions in the following analysis.

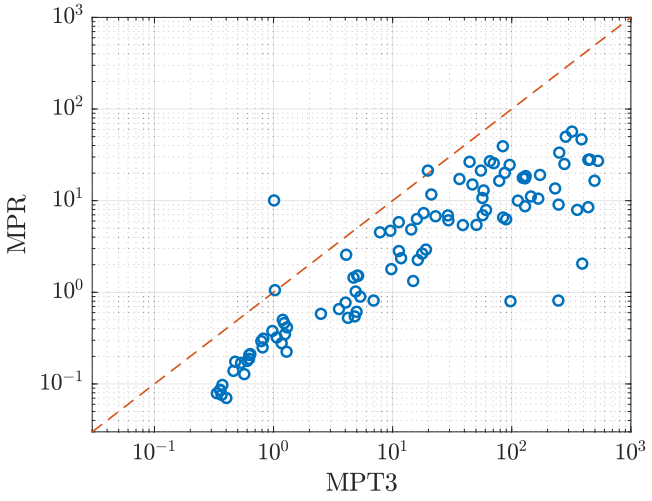

Fig. 3. Execution times (in seconds) of MPR (vertical axis) and MPT3 (horizontal axis) for the POP-mpQP1 test set. Each circle corresponds to one problem. The red dashed line highlights the diagonal of the plot.

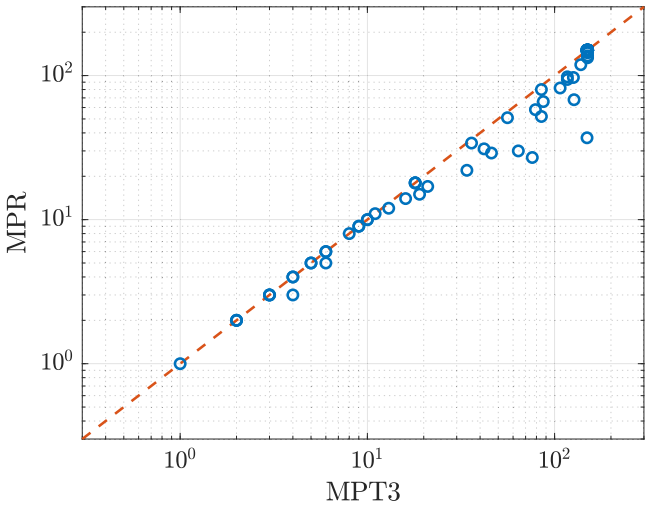

Fig. 4. Number of regions found by MPR (vertical axis) and MPT3 (horizontal axis) for the POP-mpQP1 test set. Each circle corresponds to one problem. The red dashed line highlights the diagonal of the plot.

because it has the same limit on the number of regions it can discover. From the picture we also see that when the true number of regions is relatively low (below 30) the two algorithms tend to discover the same number of regions with MPR discovering slightly fewer regions than MPT3, whereas when the true number of regions is higher (above 30) MPR discovers consistently less regions than MPT3. Even when MPT3 maxed out at 150 regions, in some tests MPR returned less regions. Finally, it is worth discussing some cases which are not visible from Fig. 4: in 44 tests both algorithms maxed out at 150 regions (these are all overlapping circles), in 1 test (Test #7) MPR found 150 regions and MPT3 none because it deemed the problem infeasible (this point does not appear because of the logarithmic scale), and in 5 tests (Tests #54, #69, #73, #86, and #90) MPR returned zero regions (also these points do not appear because of the logarithmic scale) but this is in full agreement with Corollary 2 as for those tests $\Theta_{\mathcal{C}} = \emptyset$ and $\text{vol}(\hat{\Theta}_f \cap \mathcal{D})/\text{vol}(\mathcal{D}) < 2 \cdot 10^{-4}$, which implies $\text{vol}(\Theta_f \cap \mathcal{D})/\text{vol}(\mathcal{D}) < 2 \cdot 10^{-4}$, way less than the selected coverage parameter $\varepsilon = 10^{-3}$.

We next show that, despite returning no-more regions than MPT3, MPR consistently covers a bigger portion of the parameter space than MPT3. In Fig. 5 we report, on a linear scale, the percentage of $\Theta_f \cap \mathcal{D}$ covered in volume by the two algorithms for the POP-mpQP1 test set. Each point (circle or asterisk) corresponds to one problem in the library with its horizontal coordinate reporting the value of $\text{vol}(\Theta_{\text{MPT3}} \cap \mathcal{D})/\text{vol}(\Theta_f \cap \mathcal{D}) \cdot 100$ (Θ_{MPT3} denoting the union of the regions returned by MPT3) and its vertical coordinate reporting the value of $\text{vol}(\Theta_{\mathcal{C}} \cap \mathcal{D})/\text{vol}(\Theta_f \cap \mathcal{D}) \cdot 100$ ($\Theta_{\mathcal{C}}$ denoting the union of the regions returned by MPR). Differently from the right hand side of (5) and (13), these indices measures the coverage of $\Theta_f \cap \mathcal{D}$ instead of $\mathcal{D}$. We choose to normalize with respect to $\text{vol}(\Theta_f \cap \mathcal{D})$ as for this library $\mathcal{D}$ is given and for some tests $\text{vol}(\Theta_f \cap \mathcal{D})$ is very small while for other tests $\text{vol}(\Theta_f \cap \mathcal{D}) \approx \text{vol}(\mathcal{D})$, and, for control purposes, one would be more interested in the $\Theta_f \cap \mathcal{D}$ region rather than the whole $\mathcal{D}$. As can be seen from the picture, for those tests where MPT3 maxed out at 150 regions (51/100 tests, blue circles) MPR consistently outperforms MPT3 in terms of volume coverage since all blue circles are above the diagonal, except for 4 cases: in 1 test (Test #87) they both covered more than 99.99% of $\Theta_f \cap \mathcal{D}$ and in 3 tests (Tests #36, #86, and #90) they both covered less than 0.001% of $\Theta_f \cap \mathcal{D}$ but for Tests #86 and #90 $\text{vol}(\hat{\Theta}_f \cap \mathcal{D})/\text{vol}(\mathcal{D}) < 2 \cdot 10^{-4}$, meaning that $\Theta_f \cap \mathcal{D}$ is very small when compared to $\mathcal{D}$. As for those tests

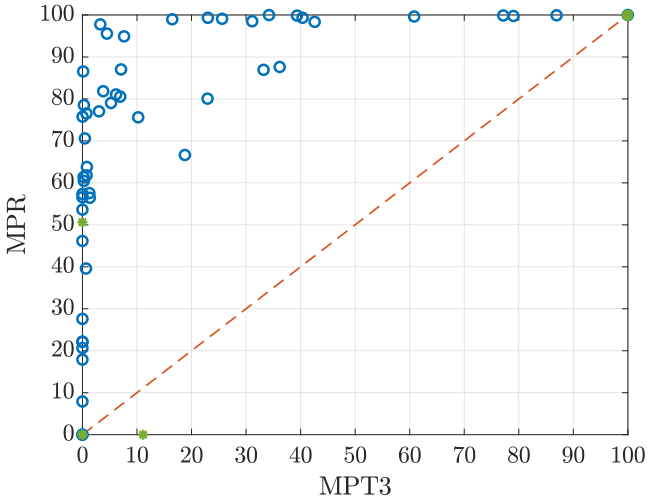

Fig. 5. Percentage of $\Theta_f \cap \mathcal{D}$ covered by MPR (vertical axis) and MPT3 (horizontal axis) for the POP-mpQP1 test set. Each point (circle or asterisk) corresponds to one problem. Blue circles corresponds to tests where MPT3 maxed out at 150 regions while green asterisks corresponds to tests where MPT3 exited normally. The red dashed line highlights the diagonal of the plot.

where MPT3 exited normally (49/100, green asterisks) we would expect both algorithm to cover almost 100% of $\Theta_f \cap \mathcal{D}$. Indeed, this is the case for 45 tests, each one covering more than 99.7% of $\Theta_f \cap \mathcal{D}$ (these are all overlapping asterisks in the top-right corner), but there are 4 peculiar tests (asterisks in the bottom-left part): for 2 tests (Tests #54, and #73) both algorithms covered less than 0.001% of $\Theta_f \cap \mathcal{D}$ but for these tests $\text{vol}(\hat{\Theta}_f \cap \mathcal{D})/\text{vol}(\mathcal{D}) < 2 \cdot 10^{-4}$ meaning that $\Theta_f \cap \mathcal{D}$ is very small when compared to $\mathcal{D}$, for Test #69 MPT3 covered 11.1% of $\Theta_f \cap \mathcal{D}$ while MPR covered 0% of $\Theta_f \cap \mathcal{D}$ (it returned 0 regions) but this is in agreement with Corollary 2 since $\text{vol}(\Theta_f \cap \mathcal{D})/\text{vol}(\mathcal{D}) < 2 \cdot 10^{-4} \ll \varepsilon$, and, finally, for Test #7 MPT3 covered 0% of $\Theta_f \cap \mathcal{D}$ (deemed the problem infeasible) while MPR covered 50.7% of $\Theta_f \cap \mathcal{D}$ and stopped because it maxed out at 150 regions.

5. Conclusions

To conclude, in this paper, we presented a new randomized algorithm for constructing an optimal solution map of a multi-parametric quadratic program via sampling of the parameter space. The proposed procedure is probabilistically guaranteed to cover the region where the user wants to compute the solution, except for a user-selected fraction. The exploration process can be guided by the user through the selection of the probability distribution for sampling the parameter space and automatically prioritizes larger regions over smaller ones when the uniform distribution is selected, resulting in broader coverage of the parameter space when there are constraints on runtime or on the maximum number of regions to be found. We compared the proposed method with the state-of-the-art multi-parametric programming toolbox MPT3, and the proposed approach was shown to outperform MPT3 in both computation time and parameter space coverage, despite typically returning a smaller number of regions.

An interesting direction for future research would be to find a way to detect and "close" the gaps that MPR may leave in the partition or to complement them with a suboptimal (but feasible) solution. Some inspiring ideas can be found in Chen et al. (2018), where the problem is recovering feasibility when the MPC control law is coded by a neural network. Another intriguing problem would be to adapt the probability distribution based on the extracted samples to better scale the exploration. Another direction for future research would be to extend our sampling strategy to a broader class of parametric problems, like those addressed according to an enumerative resolution approach in Mate et al. (2023).

Appendix A. Proof of Theorem 1

We start by noticing that the outcome of Algorithm 1 would be the same if at each iteration of the while loop a batch of N samples were independently extracted from $(\mathcal{D}, \mathcal{B}(\mathcal{D}), \mathbb{P})$ and the resulting $\bar{\vartheta}^{(i)}$, $i = 1, \dots, N$, values in the multi-sample $\omega = (\bar{\vartheta}^{(1)}, \dots, \bar{\vartheta}^{(N)})$ were examined starting from $i = 1$ and increasing i progressively till either a sample $\bar{\vartheta}^{(i)}$ from a new critical region were discovered and the while loop were run once more, or the algorithm were halted because all N samples belong to either the already identified critical regions or the infeasibility region.

Note that the maximal number of iterations of this equivalent algorithm is $M = n_{r,b} + 1$, where $n_{r,b} \leq n_r$ denotes the number of critical regions intersecting $\mathcal{D}$. We can then refer to the product space $(\mathcal{D}^{NM}, \mathcal{B}(\mathcal{D})^{NM}, \mathbb{P}^{NM})$ to determine the measure of the set $\mathcal{S}$ of multi-samples $(\omega_1, \dots, \omega_M)$ extracted from $\mathcal{D}^{NM}$ such that the algorithm stops but the probability $\mathbb{P}$ of extracting from $\mathcal{D}$ a parameter ϑ that belongs either to one of the identified critical regions or to the infeasibility set is smaller than $1 - \varepsilon$. N should be chosen so that this measure is smaller than β.

To this purpose, we can partition $\mathcal{S}$ into $M - 1$ subsets $\mathcal{S}_j$, $j = 0, 1, \dots, M - 2$, such that $\mathcal{S}_j$ is the set of multi-samples $(\omega_1, \dots, \omega_M)$ where the first j batches of N samples allow to identify a set $\Theta_{\to j} = \Theta_{\to j}((\omega_1, \dots, \omega_j))$ composed of j critical regions (with $\Theta_{\to 0} = \emptyset$), the $(j + 1)$th batch has all samples belonging to

$$\mathcal{R}_j = \Theta_{\to j} \cup (\mathcal{D} \setminus \Theta_f),$$

and the probability of $\mathcal{R}_j$ is smaller than $1 - \varepsilon$.

Set $\mathcal{S}_j$, $j = 0. \dots, M - 2$, can be characterized as follows:

$$\mathcal{S}_j = \{(\omega_1, \dots, \omega_{j+1}, \dots, \omega_M) : \ (\omega_1, \dots, \omega_{j+1}) \in \Omega_{j+1}\}$$

where

$$\Omega_{j+1} = \Big\{(\omega_1, \dots, \omega_{j+1}) : \omega_k \notin \mathcal{R}_{k-1}^N, \ k = 1, \dots, j, \ \omega_{j+1} \in \mathcal{R}_j^N, \ \mathbb{P}(\mathcal{R}_j) < 1 - \varepsilon\Big\}.$$

Let us now define $\bar{\mathbb{P}}^j = \mathbb{P}^{jN}$ as a shorthand for $\mathbb{P}^{jN}$, for any $j = 1, \dots, M$. Given that $\mathcal{S} = \cup_{j=0}^{M-2} \mathcal{S}_j$ and the sets $\mathcal{S}_j, j = 0, \dots, M-2$, are disjoint, the probability measure of $\mathcal{S}$ can then be computed as

$$\bar{\mathbb{P}}^M(\mathcal{S}) = \sum_{j=0}^{M-2} \bar{\mathbb{P}}^M(\mathcal{S}_j) = \sum_{j=0}^{M-2} \bar{\mathbb{P}}^{(j+1)}(\Omega_{j+1}). \tag{A.1}$$

If we set

$$\tilde{\Omega}_j = \Big\{(\omega_1, \dots, \omega_j) : \omega_k \notin \mathcal{R}_{k-1}^N, \ k = 1, \dots, j, \ \mathbb{P}(\mathcal{R}_j) < 1 - \varepsilon\Big\}$$

we can compute $\bar{\mathbb{P}}^{(j+1)}(\Omega_{j+1})$ as follows

$$\begin{aligned}\bar{\mathbb{P}}^{(j+1)}(\Omega_{j+1}) &= \int_{\tilde{\Omega}_j} \left(\int_{\mathcal{R}_j^N} \mathrm{d}\bar{\mathbb{P}}(\omega_{j+1}) \right) \mathrm{d}\bar{\mathbb{P}}^j((\omega_1, \dots, \omega_j)) \\ &= \int_{\tilde{\Omega}_j} \mathbb{P}(\mathcal{R}_j)^N \mathrm{d}\bar{\mathbb{P}}^j((\omega_1, \dots, \omega_j))\end{aligned} \tag{A.2}$$

where we used the fact that

$$\int_{\mathcal{R}_j^N} \mathrm{d}\bar{\mathbb{P}}(\omega_{j+1}) = \int_{\mathcal{R}_j^N} \mathrm{d}\mathbb{P}^N(\omega_{j+1}) = \mathbb{P}(\mathcal{R}_j)^N.$$

Now since $\mathbb{P}(\mathcal{R}_j) \leq 1 - \varepsilon$ over $\tilde{\Omega}_j$, from (A.2) we obtain

$$\bar{\mathbb{P}}^{(j+1)}(\Omega_{j+1}) \leq (1 - \varepsilon)^N \bar{\mathbb{P}}^j(\tilde{\Omega}_j) \leq (1 - \varepsilon)^N. \tag{A.3}$$

By plugging (A.3) into (A.1), we finally get

$$\bar{\mathbb{P}}^M(S) = \sum_{j=0}^{M-2} \bar{\mathbb{P}}^{j+1}(\Omega_{j+1}) \leq (M - 1)(1 - \varepsilon)^N$$

We shall then choose N such that

$$(M - 1)(1 - \varepsilon)^N \leq \beta$$

thus getting

$$(1 - \varepsilon)^N \leq \frac{\beta}{n_{r,b}}.$$

which is equivalent to

$$N \geq \frac{\log \beta - \log n_{r,b}}{\log(1 - \varepsilon)}.$$

Since $n_{r,b} \leq n_r$, we can then require

$$N \geq \frac{\log \beta - \log n_r}{\log(1 - \varepsilon)}$$

to enforce $\bar{\mathbb{P}}^M(\mathcal{S}) \leq \beta$, and this concludes the proof. □

Appendix B. Proof of Proposition 1

Consider a realization of N_s samples extracted independently from $\mathcal{D}$ according to $\mathbb{P}$.

If we run Algorithm 1 with $N = \bar{N} = \left\lfloor \frac{N_s}{n_r + 1} \right\rfloor$ on this multi-sample realization by processing the samples sequentially, then Algorithm 1 will either stop before all N_s samples are processed or process them all providing a full coverage of $\Theta_f \cap \mathcal{D}$ (see the discussion at the beginning of the proof of Theorem 1).

If we run Algorithm 2 on the same multi-sample realization, all N_s samples will be processed thus providing a coverage that cannot be worse than that obtained with Algorithm 1, since additional critical regions intersecting $\mathcal{D}$ will be possibly discovered, thus resulting in a lower ε coverage parameter value.

Recall now that, from Theorem 1 and Remark 1, it follows that, when Algorithm 1 is applied, all except a β probability set of multi-sample realizations are guaranteed to provide a coverage $1 - \varepsilon$ where ε satisfies (4) with $\bar{N}$ in place of N. This, together with the previous observation, entails that the same guarantees hold for Algorithm 2, which concludes the proof. □

References

Ahmadi-Moshkenani, P., Johansen, T. A., & Olaru, S. (2018). Combinatorial approach toward multiparametric quadratic programming based on characterizing Adjacent Critical Regions. *IEEE Transactions on Automatic Control*, *63*(10), 3221–3231.

Ahmadi-Moshkenani, P., Olaru, S., & Johansen, T. A. (2016). Further results on the exploration of combinatorial tree in multi-parametric quadratic programming. In *2016 European control conference* (pp. 116–122).

Alessio, A., & Bemporad, A. (2009). A survey on explicit model predictive control. In *Nonlinear model predictive control: towards new challenging applications* (pp. 345–369). Springer.

Baotić, M. (2002). *An efficient algorithm for multi-parametric quadratic programming*: *Technical report*, Physikstrasse 3, CH-8092, Switzerland: ETH Zürich, Institut für Automatik.

Baotić, M., Borrelli, F., Bemporad, A., & Morari, M. (2008). Efficient on-line computation of constrained optimal control. *SIAM Journal on Control and Optimization*, *47*(5), 2470–2489.

Bemporad, A. (2004). Hybrid toolbox - user's guide. http://cse.lab.imtlucca.it/~bemporad/hybrid/toolbox.

Bemporad, A., Borrelli, F., & Morari, M. (2002). Model predictive control based on linear programming — The explicit solution. *IEEE Transactions on Automatic Control*, *47*(12), 1974–1985.

Bemporad, A., Morari, M., Dua, V., & Pistikopoulos, E. N. (2002). The explicit linear quadratic regulator for constrained systems. *Automatica*, *38*(1), 3–20.

Bemporad, A., Oliveri, A., Poggi, T., & Storace, M. (2011). Ultra-fast stabilizing model predictive control via canonical piecewise affine approximations. *IEEE Transactions on Automatic Control*, *56*(12), 2883–2897.

Bertsekas, D. (2015). *Convex optimization algorithms*. Athena Scientific.

Chen, S., Saulnier, K., Atanasov, N., Lee, D. D., Kumar, V., Pappas, G. J., & Morari, M. (2018). Approximating explicit model predictive control using constrained neural networks. In *2018 annual American control conference* (pp. 1520–1527).

Falsone, A., Bianchi, F., & Prandini, M. (2023). Dealing with infeasibility in multi-parametric programming for application to explicit model predictive control. *Automatica*, *157*, Article 111279.

Feller, C., & Johansen, T. A. (2013). Explicit MPC of higher-order linear processes via combinatorial multi-parametric quadratic programming. In *2013 European control conference* (pp. 536–541).

Feller, C., Johansen, T. A., & Olaru, S. (2013). An improved algorithm for combinatorial multi-parametric quadratic programming. *Automatica*, *49*(5), 1370–1376.

Gupta, A., Bhartiya, S., & Nataraj, P. S. V. (2011). A novel approach to multiparametric quadratic programming. *Automatica*, *47*(9), 2112–2117.

Herceg, M., Jones, C. N., Kvasnica, M., & Morari, M. (2015). Enumeration-based approach to solving parametric linear complementarity problems. *Automatica*, *62*, 243–248.

Herceg, M., Kvasnica, M., Jones, C. N., & Morari, M. (2013). Multi-parametric toolbox 3.0. In *2013 European control conference* (pp. 502–510).

Jones, C. N., & Morrari, M. (2006). Multiparametric linear complementarity problems. In *Proceedings of the 45th IEEE conference on decision and control* (pp. 5687–5692).

Kvasnica, M., & Fikar, M. (2011). Clipping-based complexity reduction in explicit MPC. *IEEE Transactions on Automatic Control*, *57*(7), 1878–1883.

Kvasnica, M., Grieder, P., Baotić, M., & Morari, M. (2004). Multi-parametric toolbox (MPT). In *Hybrid systems: computation and control* (pp. 448–462).

Mate, S., Jaju, P., Bhartiya, S., & Nataraj, P. (2023). Semi-explicit model predictive control of quasi linear parameter varying systems. *European Journal of Control*, *69*, Article 100750.

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

Mitze, R., & Mönnigmann, M. (2020). A dynamic programming approach to solving constrained linear–quadratic optimal control problems. *Automatica*, *120*, Article 109132.

Morari, M., & H. Lee, J. (1999). Model predictive control: past, present and future. *Computers & Chemical Engineering*, *23*(4), 667–682.

Nguyen, N. A., Gulan, M., Olaru, S., & Rodriguez-Ayerbe, P. (2017). Convex lifting: Theory and control applications. *IEEE Transactions on Automatic Control*, *63*(5), 1243–1258.

Oberdieck, R., Diangelakis, N. A., Papathanasiou, M. M., Nascu, I., & Pistikopoulos, E. N. (2016). POP - The Parametric Optimization Toolbox. https://parametric.tamu.edu/POP/. (Accessed 24 January 2024).

Oberdieck, R., Diangelakis, N. A., & Pistikopoulos, E. N. (2017). Explicit model predictive control: A connected-graph approach. *Automatica*, *76*, 103–112.

Pistikopoulos, E. N., Diangelakis, N. A., & Oberdieck, R. (2020). *Wiley series in operations research and management science*, *Multi-parametric optimization and control*. Wiley.

Spjøtvold, J., Kerrigan, E. C., Jones, C. N., Tøndel, P., & Johansen, T. A. (2006). On the facet-to-facet property of solutions to convex parametric quadratic programs. *Automatica*, *42*(12), 2209–2214.

Tøndel, P., Johansen, T. A., & Bemporad, A. (2003). An algorithm for multi-parametric quadratic programming and explicit MPC solutions. *Automatica*, *39*(3), 489–497.

Xu, J., Lou, Y., De Schutter, B., & Xiong, Z. (2024). Error-free approximation of explicit linear MPC through lattice piecewise affine expression. *IEEE Transactions on Automatic Control*.

Alessandro Falsone received his Master degree in Automation and Control Engineering in 2013, and his Ph.D. degree in Information Technology, in 2018, both cum laude and from Politecnico di Milano. During his Ph.D. studies he spent three months as a visiting researcher at the University of Oxford. In 2018 he joined the Dipartimento di Elettronica, Informazione e Bioingegneria at Politecnico di Milano, where he is now Associate Professor. His current research interests include distributed optimization and control, passivity theory, data-based optimal control of stochastic systems, and multi-parametric programming. In 2018 he was the recipient of the Dimitris N. Chorafas Prize for his Ph.D. thesis. In 2019 he received the IEEE CSS Italy Chapter Best Young Author Journal Paper Award.

Federico Bianchi received the M.Sc. degree in Computer Science and Engineering and the Ph.D. in Information Technology, both from Politecnico di Milano, Department of Electronics, Information and Bioengineering (DEIB). After two years as a Postdoctoral Researcher in the Systems and Control group at DEIB, he is currently a tenured researcher at Ricerca sul Sistema Energetico (RSE) S.p.A., Milan, Italy, within the Automation and Control of Energy Networks Research Group, Generation Technologies and Materials Department. His research focuses on nonlinear and hybrid system identification, modeling and control of energy systems, energy flexibility aggregation, and vehicle-to-grid applications. He serves as an Associate Editor for the IEEE Control Systems Society Technical Conference Editorial Board and represents Italy in CIGRE Working Group C1.52 on Virtual Power Plants. +39 324 9226699

Maria Prandini received the Ph.D. degree in Information Technology from the University of Brescia, Italy, in 1998. She was a postdoctoral researcher at the University of California at Berkeley from 1998 to 2000. She also held visiting positions at Delft University of Technology (1998), Cambridge University (2000), UC Berkeley (2005), ETH Zurich (2006), and University of Oxford (2022). In 2002, she joined Politecnico di Milano, where she is currently full professor. She was elected Fellow of the IEEE in 2020 and received the IEEE Control Systems Society Distinguished Member award in 2018. In 2017, she was August-Wilhelm Scheer Visiting Professor and Honorary fellow of the TUM Institute for Advanced Studied. She was nominated Visiting Professor in Engineering at the University of Oxford for the triennium 2022–2025, renewed for 2025–2028. She has been contributing to the IEEE Control Systems Society (CSS), the International Federation of Automatic Control (IFAC), and the Association for Computing Machinery (ACM), contributing to their activities in different roles. She is IFAC President-elect for the triennium 2023-26. Previously, she was Vice-President for conference activities for IFAC (2020–23) and IEEE CSS (2016 and 2017), and a member of SIGBED Board of Directors (2019–21). Her research interests include stochastic hybrid systems, randomized algorithms, distributed and data-driven optimization, multi-agent systems, and the application of control theory to transportation and energy systems.