Control analysis and design via randomised coordinate polynomial minimisation

tion problems with quadratic objective functions, see Nesterov (1998), Alon and Naor (2006).

The mentioned convex relaxations enjoy important theoretical properties: it is shown for instance in Lasserre (2001) that the minimum of a polynomial over a compact set can be approximated as closely as desired by solving a finite sequence of semidefinite programs. SDP relaxations thus provide *guaranteed* lower bounds on the global optimum and, if finite convergence occurs at some point in the approximation hierarchy, they can provide a certified global minimiser. In practice, however, SDP relaxation methods may become numerically very demanding already for medium-scale problems, essentially due to linear matrix inequality (LMI) representations with a matrix size that scales with the number of monomials of degree d in n variables as $\binom{n+d}{n}$, see, e.g.Parrilo and Sturmfels (2003). The difficulty in providing solutions in reasonable time already in medium-sized problems make SDP relaxation methods hardly usable in practical control applications, especially in those contexts where control actions need be computed in real time, such as in nonlinear model predictive control (NMPC).

There hence appears to exist a need for reliable numerical methods that can tackle constrained polynomial optimisation problems of realistic size and provide: (a) optimal or sub-optimal solutions in reasonable time, and (b) some form of theoretical guarantee of convergence to the global optimum. A class of numerical methods that gained popularity in recent years, especially in the context of machine learning, is that of *coordinate descent methods*, see Wright (2015) for a recent survey. Coordinate descent algorithms solve optimisation problems by successively performing exact or approximate minimisations along coordinate directions.

An example of these techniques is the *Gauss–Seidel* (GS) method, which is based on updating the current solution vector with the solution of the scalar problem obtained by allowing the variation of just one coordinate at a time. In its classical implementation, such a method updates the entries of the estimate cyclically, starting from an initial point in the feasible set. Convergence results of the classical GS method have been given for both the constrained and unconstrained cases under suitable (pseudo)convexity assumptions, see, e.g. Zadeh (1970), Bertsekas and Tsitsiklis (1989), Grippo and Sciandrone (2000), even in certain non-differentiable cases (Tseng, 2001). However, for nonconvex problems, the GS method need not converge to a critical point, see, e.g.the counter-examples given in Powell (1973). Nevertheless, algorithms based on the GS method have been proved useful in practice for solving certain control problems, see, e.g. Bloemen and Verbruggen (2004).

Stochastic versions of the GS method, usually referred to as *random coordinate descent* methods, exist in which the coordinate to be updated is chosen each time at random; see, e.g. Nesterov (2012), Necoara and Clipici (2013), Richtárik and Takáč (2014). These methods were proved successful in obtaining an η-accurate solution with probability at least $1-\sigma$, with η and σ being arbitrary positive real numbers, provided that the function to be minimised is the sum of a smooth convex and a convex nonsmooth but block-separable function. Further, such methods are capable of solving huge-scale problems in a reasonable amount of time, see Nesterov (2014). Coordinate descent methods, however, generally have no guarantee of converging to the global optimum, for nonconvex problems.

A family of methods for dealing with general nonconvex problems is the *random search algorithms* which, in their most basic form, are based on iterating of the following procedure: pick a random sample from a suitable neighbourhood of the current estimate of the solution, and update such an estimate with the picked random sample if the value of the objective function decreases. Such simple scheme can be proved to converge with probability one to a neighbourhood of the global optimum, see, e.g.Solis and Wets (1981), in which also a linear relation between the mean number of function evaluations and the dimension is found experimentally, see also Schumer and Steiglitz (1968) and Rastrigin (1963). After some popularity for control applications in the 70s, random search techniques fell out of fashion for a while, until they were reemployed in different flavours in the early 2000s in the context of Monte Carlo and randomised algorithms for robust control, see, e.g. Spall (2005) and Tempo et al. (2012).

One goal of the present work is to propose a novel type of computational method for control that may synergise the speed and large-scale efficiency of coordinate-descent methods with the probabilistic global optimality properties of random search methods. In this direction, we propose a general method for solving nonconvex constrained polynomial optimisation problems based on a nonuniform random coordinate minimisation algorithm, in which transverse directions (i.e. different from the coordinate axes) are taken with nonzero probability. The proposed algorithm can be envisioned as a nonuniform random coordinate descent method, coupled with a hit-and-run type of random search (see, e.g. Zabinsky, 2008) in which, however, rather than moving to a random point along the randomly chosen direction, we move to the global minimum along that direction.

In particular, at each iteration the proposed algorithm picks, with probability p, a random coordinate direction (possibly, with nonuniform probability), or, with probability $1-p$, a direction from the unit sphere uniformly at random. The current estimate of the solution to the nonlinear program is then updated with the solution to the univariate optimisation problem obtained restricting the original problem to the line passing through the current estimate and having the randomly selected direction. The main advantage of allowing transverse directions is that it enables guaranteed convergence in probability to the global solution, while preserving the effectiveness of coordinate-descent methods. Further, since the proposed method is a descent method, it always returns a feasible suboptimal solution whenever it is stopped, even before convergence, and this is a key enabling feature in real-time control applications.

A preliminary version of this method was presented in Calafiore and Possieri (2018), where it was assumed that the feasible set is convex (such an hypothesis is removed here) and that the distribution governing the selection of the coordinates is uniform, whereas we here allow such probability mass function to be generic.

Further, in the present work we provide a detailed iteration complexity analysis in Section 4, present a specific result for

finding an initial feasible point in Theorem 4.1, and we develop a full-fledged NMPC example in Section 6, through which the proposed method is compared with state-of-the-art techniques to solve polynomial optimisation problems in practical applications.

The algorithm proposed here has been developed into a `Matlab` package ,[1] which was used for all the examples reported in this paper, including an NMPC example on an aerospace system discussed in Section 6. These examples suggest that the proposed technique performs well compared to other existing methods, and that it can provide practical solutions for many cases in which other tools are unsuccessful.

2. Problem statement and motivating examples

Let $\mathbb{Z}$, $\mathbb{N}$, $\mathbb{R}$, $\mathbb{R}_{\geqslant 0}$, and $\mathbb{R}_{>0}$ denote the sets of integer, natural, real, nonnegative real, and positive real numbers, respectively. The symbols $\mathbb{B}^n$, $\mathbb{B}^n_o$ and $\mathbb{S}^n$ denote the closed and open unit balls and the unit sphere in $\mathbb{R}^n$, respectively. Let $\mathbf{e}_i$ denote the ith standard unit vector. The symbol $\operatorname{int}\mathcal{A}$ denotes the *interior* of the set $\mathcal{A} \subset \mathbb{R}^n$.

Given a compact set $\mathcal{A} \subset \mathbb{R}^n$, $\|\mathbf{x}\|_{\mathcal{A}} \doteq \inf_{\mathbf{y}\in\mathcal{A}} \|\mathbf{x}-\mathbf{y}\|_2$ denotes the ℓ_2 distance between $\mathbf{x} \in \mathbb{R}^n$ and $\mathcal{A}$. A function $\varrho : \mathbb{R}^n \to \mathbb{R}_{\geqslant 0}$ is positive semidefinite with respect to $\mathcal{A}$, denoted $\varrho \in \mathcal{PD}(\mathcal{A})$, if $\varrho(\mathbf{x}) = 0 \iff \mathbf{x} \in \mathcal{A}$. Given $\mathcal{T} \subset \mathbb{R}^n$, let $\mathbb{I}_{\mathcal{T}}(\cdot)$ be the *indicator function of* $\mathcal{T}$. Let $\delta(\cdot)$ denote the *Kronecker delta*. A continuous function $\alpha : \mathbb{R}_{\geqslant 0} \to \mathbb{R}_{\geqslant 0}$ is of class $\mathcal{K}_\infty$, if it is strictly increasing, $\alpha(0) = 0$, and $\lim_{r\to+\infty} \alpha(r) = +\infty$. A function $f : \mathbb{R}^n \to \mathbb{R}$ is *radially unbounded* on $\Omega \subset \mathbb{R}^n$, denoted $f \in \mathrm{ru}(\Omega)$, if for every $\{\mathbf{x}^k\}_{k\in\mathbb{N}}$ such that $\mathbf{x}^k \in \Omega$ for all $k \in \mathbb{N}$ and $\lim_{k\to+\infty} \|\mathbf{x}^k\|_2 = +\infty$, one has $\lim_{k\to\infty} f(\mathbf{x}^k) = +\infty$.

The symbol $\mathrm{Uni}(\cdot)$ denotes the uniform distribution over its set argument, and the symbol $\sim$ reads as 'has the distribution'.

A set-valued mapping $G : \mathbb{R}^n \times \mathbb{S}^n \rightrightarrows \mathbb{R}^n$ is said to be *with closed values* if $\mathbf{x} \mapsto G(\mathbf{x}, \mathbf{s})$ is outer-semicontinuous for all $\mathbf{s} \in \mathbb{S}^n$; see Definition 5.4 and Theorem 5.7(a) of Rockafellar and Wets (2009).

Let $\mathbf{x} = [x_1 \dots x_n]^\top$, with $n \in \mathbb{N}$, be a vector of variables. Given $\boldsymbol{\alpha}$, let $|\boldsymbol{\alpha}| \doteq \sum_{i=1}^n \alpha_i$ and $\mathbf{x}^{\boldsymbol{\alpha}} = x_1^{\alpha_1} \dots x_n^{\alpha_n}$. A *polynomial* p in x is a finite, $\mathbb{R}$-linear combination of monomials, $p = \sum_{\boldsymbol{\alpha}\in\mathcal{E}} c_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where $\mathcal{E} \subset \mathbb{N}^n$ is a finite set and $c_{\boldsymbol{\alpha}} \in \mathbb{R}$, $\forall \boldsymbol{\alpha} \in \mathcal{E}$; the *total degree of* p is $\deg(p) \doteq \max\{|\boldsymbol{\alpha}|, \boldsymbol{\alpha} \in \mathcal{E}\}$. The *ring* of all the polynomials in $\mathbf{x}$ with coefficients in $\mathbb{R}$ is $\mathbb{R}[\mathbf{x}]$, whereas $\mathbb{R}^m[\mathbf{x}]$ denotes the set of all the m–dimensional vectors whose entries are in $\mathbb{R}[\mathbf{x}]$.

2.1 Problem statement

Let $f \in \mathbb{R}[\mathbf{x}]$ and $[h_1 \dots h_m]^\top \in \mathbb{R}^m[\mathbf{x}]$ be given. Let

$$\Omega \doteq \{\mathbf{x} \in \mathbb{R}^n : h_1(\mathbf{x}) \leqslant 0, \dots, h_m(\mathbf{x}) \leqslant 0\},$$

and assume that Ω is full-dimensional, i.e. $\int_\Omega 1\,\mathrm{d}\mathbf{x} \neq 0$. We define the *polynomial minimisation problem* (PMP)

$$\left|\begin{array}{ll} \min & f(\mathbf{x}), \\ \text{s.t.} & \mathbf{x} \in \Omega. \end{array}\right. \tag{1}$$

The main goal of this paper is to design a procedure for solving the PMP (1), that is, for finding $f^\star \in \mathbb{R}$ and $\mathbf{x}^\star \in \Omega$ such that

$$f^\star = f(\mathbf{x}^\star) = \min_{\mathbf{x}\in\Omega} f(\mathbf{x}).$$

2.2 Motivating control examples

We next propose a brief motivating selection of some specific control problems that can be reduced to the solution a PMP in the form (1). A broader perspective on polynomial optimisation problems in control is given in the Introduction, and further practical examples can be found in, e.g. Henrion and Lasserre (2004).

2.2.1 LQ differential games

Consider the scenario in which N agents attempt at selfishly optimizing individual and potentially conflicting objectives in a non-cooperative environment. The system is assumed to be completely characterised by the state $\boldsymbol{\xi}(t) \in \mathbb{R}^\nu$, whose dynamics are

$$\dot{\boldsymbol{\xi}}(t) = \mathbf{A}\,\boldsymbol{\xi}(t) + \sum_{i=1}^N \mathbf{B}_i \mathbf{u}_i(t), \tag{2a}$$

where $\mathbf{A} \in \mathbb{R}^{\nu\times\nu}$, $\mathbf{B}_i \in \mathbb{R}^{\nu\times\mu_i}$, $\mathbf{u}_i(t) \in \mathbb{R}^{\mu_i}$ is the control action of the ith agent, $i = 1, \dots, N$. The objective of the ith agent is to minimise

$$J_i = \int_0^\infty (\boldsymbol{\xi}^\top(t)\,\mathbf{Q}_i\,\boldsymbol{\xi}(t) + \mathbf{u}_i^\top(t)\,\mathbf{R}_i\,\mathbf{u}_i(t))\,\mathrm{d}t, \tag{2b}$$

where $\mathbf{Q}_i$ and $\mathbf{R}_i$ are positive definite matrices. In Engwerda (2005), Possieri and Sassano (2016), it has been shown that there exists a Nash equilibrium for the dynamical game (2) if there exist symmetric solutions $\mathbf{X}_i = \mathbf{X}_i^\top \in \mathbb{R}^{\nu\times\nu}$, $i = 1, \dots, N$, to the following equations:

$$\begin{aligned}\mathbf{E}_i(\mathbf{x}) \doteq \left(\mathbf{A} - \sum_{j\neq i}^N \mathbf{S}_j\mathbf{X}_j\right)^\top \mathbf{X}_i + \mathbf{X}_i\left(\mathbf{A} - \sum_{j\neq i}^N \mathbf{S}_j\mathbf{X}_j\right) \\ + \mathbf{Q}_i - \mathbf{X}_i\mathbf{S}_i\mathbf{X}_i = 0,\end{aligned} \tag{3}$$

where $\mathbf{S}_i = \mathbf{B}_i\mathbf{R}_i^{-1}\mathbf{B}_i^\top$, such that the matrix

$$\mathbf{A}_{\mathrm{cl}} \doteq \mathbf{A} - \sum_{i=1}^N \mathbf{S}_i\mathbf{X}_i$$

is Hurwitz. Therefore, letting $\wp_1, \dots, \wp_n$ be the polynomial entries of first column of the Routh table of the characteristic polynomial of $\mathbf{A}_{\mathrm{cl}}$, and considering that $\mathbf{A}_{\mathrm{cl}}$ is Hurwitz if and only if $\wp_i > 0$, $i = 1, \dots, \nu$, determining a Nash equilibrium of the game (2) corresponds to solving the following PMP in the entries $\mathbf{x}$ of the matrices $\mathbf{X}_1, \dots, \mathbf{X}_N$,

$$\left|\begin{array}{ll} \min & \|\mathrm{vec}(\mathbf{E}_i((\mathbf{x})))\|_2^2 + \dots + \|\mathrm{vec}(\mathbf{E}_N(\mathbf{x})))\|_2^2, \\ \text{s.t.} & \wp_i(\mathbf{x}) \geqslant \varepsilon, \quad i = 1, \dots, n, \end{array}\right. \tag{4}$$

where the symbol $\mathrm{vec}(\cdot)$ denotes the *vec* operator and $\varepsilon \in \mathbb{R}_{>0}$ is a sufficiently small parameter.

2.2.2 ℓ_2 distance between a point and a polynomial surface

Determining the ℓ_2 distance between a point and a surface is an important problem in both robust and nonlinear control, see, e.g.Chesi et al. (2001). Given a surface described as $\mathcal{V} = \{\mathbf{x} \in \mathbb{R}^n : h(\mathbf{x}) = 0\}$, where h is a polynomial, and a point $\mathbf{y} \in \mathbb{R}^n$, assume, without loss of generality, that $h(\mathbf{y}) > 0$ (otherwise, $h \leftarrow -h$). Computing $\|\mathbf{y}\|_{\mathcal{V}}$ corresponds to solving in $\mathbf{x}$,

$$\left| \begin{array}{ll} \min & \|\mathbf{x}-\mathbf{y}\|_2^2, \\ \text{s.t.} & h(\mathbf{x}) = 0. \end{array} \right. \tag{5}$$

Actually, it can be proved that for this problem we can substitute the equality constraint with an inequality one (which will be active at optimum), and thus solve the PMP

$$\left| \begin{array}{ll} \min & \|\mathbf{x}-\mathbf{y}\|_2^2, \\ \text{s.t.} & h(\mathbf{x}) \leqslant 0. \end{array} \right. \tag{6}$$

2.2.3 Nonlinear model predictive control

NMPC is one of the most effective and flexible frameworks for designing control inputs for nonlinear systems, taking into account input/state/output constraints and managing systematically the trade-off between performance and control effort, see, e.g.Mayne et al. (2000), Grüne and Pannek (2011). Consider a discrete-time system of the form

$$\boldsymbol{\xi}(k+1) = \boldsymbol{\varsigma}(\boldsymbol{\xi}(k), \mathbf{u}(k)), \tag{7}$$

where $\boldsymbol{\varsigma} = [\varsigma_1 \ \dots \ \varsigma_\nu]^\top \in \mathbb{R}^\nu[\boldsymbol{\xi}, \mathbf{u}]$, and $\mathbf{u}(k) \in \mathbb{R}^\mu$ denotes the control input. The set of admissible inputs is specified as

$$\mathcal{U} \doteq \{\mathbf{u} \in \mathbb{R}^\mu : h_1(\mathbf{u}) \leqslant 0, \dots, h_{m_1}(\mathbf{u}) \leqslant 0\},$$

where $h_1, \dots, h_{m_1} \in \mathbb{R}[\mathbf{u}]$ are given. The set of admissible states is specified as

$$\mathcal{X} \doteq \{\boldsymbol{\xi} \in \mathbb{R}^\nu : y_1(\boldsymbol{\xi}) \leqslant 0, \dots, y_{m_2}(\boldsymbol{\xi}) \leqslant 0\},$$

for given $y_1, \dots, y_{m_2} \in \mathbb{R}[\boldsymbol{\xi}]$. The system performance is expressed via a given function $q \in \mathbb{R}[k, \boldsymbol{\xi}(k), \dots, \boldsymbol{\xi}(k+\kappa), \mathbf{u}(k), \dots, \mathbf{u}(k+\kappa)]$. Assuming that the state $\boldsymbol{\xi}(k)$ can be measured, NMPC essentially consists in determining, for each $k \in \mathbb{N}$, a sequence of control inputs $\{\mathbf{u}^\star(k+\kappa)\}_{\kappa=0,\dots,N}$ that solves the PMP

$$\left| \begin{array}{ll} \min & \displaystyle\sum_{\kappa=0}^{N} q(\kappa, \boldsymbol{\xi}(k+\kappa), \mathbf{u}(k+\kappa)), \\ \text{s.t.} & \boldsymbol{\xi}(\kappa+1) = \boldsymbol{\varsigma}(\boldsymbol{\xi}(\kappa), \mathbf{u}(\kappa)), \quad \kappa = k, \dots, k+N-1, \\ & h_i(\mathbf{u}(\kappa)) \leqslant 0, \quad \kappa = 0, \dots, N, \ i = 0, \dots, m_1, \\ & y_i(\boldsymbol{\xi}(\kappa)) \leqslant 1, \quad \kappa = 0, \dots, N, \ i = 0, \dots, m_2. \end{array} \right. \tag{8}$$

Once a solution to the PMP (8) has been determined, the control input $\mathbf{u}^\star(k)$ is given as input to system (7), and then the process is repeated at the next step, in a rolling horizon fashion. Assuming that $\boldsymbol{\varsigma}$, h_i, y_i and q are polynomials, by substituting the expression for $\boldsymbol{\xi}(\kappa+1)$ in the objective function and in the constraints, the PMP (8) has the form (1).

The hypothesis of polynomial functions is not restrictive, in theory. Indeed, any square-integrable function can be approximated with arbitrary precision on a compact set by an orthogonal polynomial superposition. Also from a practical standpoint, a polynomial approximation of non-polynomial functions can be obtained numerically, in a data-driven framework.

In the next section, we describe our proposed numerical method to tackle such problems.

3. Random coordinate descent with transverse directions

The algorithm we propose to solve the PMP (1) is described in words as follows: for any current solution estimate $\mathbf{x}^k \in \Omega$ at iteration k, with probability $p \in [0, 1)$ we pick a random coordinate direction $\mathbf{v} \in \{\mathbf{e}_1, \dots, \mathbf{e}_n\}$ with $\mathbb{P}(\mathbf{v} = \mathbf{e}_i) = \varpi(i)$, $i = 1, \dots, n$, where $\varpi : \{1, \dots, n\} \to [0, 1]$ is a given probability mass function, $\sum_{i=1}^n \varpi(i) = 1$. Otherwise (i.e. with probability $1-p$), we pick a random transverse direction $\mathbf{v}$ uniformly on $\mathbb{S}^n$. We then set $\mathbf{s}^k$ equal to $\mathbf{v}$ and we update the solution estimate according to the rule

$$\mathbf{x}^{k+1} \in \mathbf{x}^k + \lambda^k \mathbf{s}^k, \tag{9a}$$

where

$$\lambda^k \doteq \arg\min_{\lambda \in \mathcal{I}_k} f(\mathbf{x}^k + \lambda \mathbf{s}^k), \tag{9b}$$

and $\mathcal{I}_k$ is the (possibly unbounded) set

$$\mathcal{I}_k \doteq \{\lambda \in \mathbb{R} : \mathbf{x}^k + \lambda \mathbf{s}^k \in \Omega\}. \tag{9c}$$

Equation (9) are the core of the proposed minimisation algorithm; the complete algorithm will be presented in Section 4. It can be noted that our idea is to solve the multivariable optimisation problem (1) by iteratively solving the single-variable optimisation problem (9b), which minimises the objective function along a chosen direction $\mathbf{s}^k$. With probability p, $\mathbf{s}^k$ is a coordinate direction, with probability $1-p$ it is a transverse direction. We also observe that the method is non-local in nature, since the univariate minimisation in (9b) is solved for the global minimum at each iteration.

For $p = 1$, the above method is a (possibly non-uniform) random coordinate minimisation method, while for $p = 0$ it becomes a random direction search with exact line search.

The algorithm's dynamics given by (9) can be rewritten in the form of a stochastic difference inclusion

$$\mathbf{x}^{k+1} \in G(\mathbf{x}^k, \mathbf{s}^k), \tag{10}$$

where $G : \Omega \times \mathbb{S}^n \rightrightarrows \Omega$, $G(\mathbf{x}, \mathbf{s}) \doteq \{\mathbf{y} \in \Omega : \exists \lambda^\star \in \mathcal{I}$ such that $\mathbf{y} = \mathbf{x} + \lambda^\star \mathbf{s}$ and $f(\mathbf{x} + \lambda^\star \mathbf{s}) \leqslant f(\mathbf{x} + \lambda\, \mathbf{s})$, for all $\lambda \in \mathbb{R}$ such that $\mathbf{x} + \lambda\, \mathbf{s} \in \Omega\}$, and $\{\mathbf{s}^k\}_{k \in \mathbb{N}}$ is a sequence of independent, identically distributed (i.i.d.) random variables defined from the probability space $(\Psi, \mathcal{F}, \mathbb{P})$. Namely, for each $k \in \mathbb{N}$, the random variable $\mathbf{s}^k : \Psi \to \mathbb{S}^n$ is such that the probability measure $\mu(F) = \mathbb{P}(\psi \in \Psi : \mathbf{s}^k(\psi) \in F)$ is well defined for each F in the Borel σ-field on $\mathbb{S}^n$. In particular, for each $k \in \mathbb{N}$, $\mathbf{s}^k \sim (1-p)\mathrm{Uni}(\mathbb{S}^n) + p \sum_{i=1}^n \varpi(i)\delta(\mathbf{e}_i)$. In the following, the symbol $\mathcal{R}(\mathbf{x}^0)$ is used to denote the set of all the maximal solutions (i.e. those that cannot be extended) to the stochastic difference inclusion (10) starting at $\mathbf{x}^0$; see Subbaraman and Teel (2013) for formal definitions.

The following lemma establishes that, under mild assumptions about the problem, the proposed method admits well-behaved solutions. In particular, it guarantees that maximal

random solutions from Ω of the stochastic inclusion (10) exist and are complete (i.e. if $\mathbf{x}^0 \in \Omega$, then the length of each sequence $\{\mathbf{x}^k\}_{k=0}^{J_\mathbf{x}} \in \mathcal{R}(x_0)$ is $J_\mathbf{x} = +\infty$), and that the map $G : \Omega \times \mathbb{S}^n \to \Omega$ satisfies suitable regularity assumptions.

Lemma 3.1: *Define the set*

$$\mathcal{A} \doteq \{x^\star \in \Omega : f(x^\star) \leqslant f(x),\ \forall x \in \Omega\}$$

that is constituted by all the optimal solutions of the PMP (1). *Assume that* $\mathcal{A} \neq \emptyset$, *and that either* Ω *is compact or* $f \in \mathrm{ru}(\Omega)$. *Then,* $G : \Omega \times \mathbb{S}^n \to \Omega$ *is locally bounded and* $\mathbf{s} \mapsto \mathrm{graph}(G(\cdot, \mathbf{s})) \doteq \{(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^n \times \mathbb{R}^n : \mathbf{y} \in G(\mathbf{x}, \mathbf{s})\}$ *is measurable with closed values. Moreover, for all* $\mathbf{x}^0 \in \Omega$, *maximal random solutions in* $\mathcal{R}(\mathbf{x}^0)$ *exist and are complete.*

A proof of Lemma 3.1 is given in Appendix A.1. Consider now the following definition, see also Bhatia and Szegö (2002), Thygesen (1997), Teel (2013).

Definition 3.1: A compact set $\mathcal{A} \subset \mathbb{R}^n$ is *stable in probability for* (9) if for each $\varepsilon, \sigma \in \mathbb{R}_{>0}$ there exists $\delta \in \mathbb{R}_{>0}$ such that

$$\mathbf{x}^0 \in (\mathcal{A} + \delta\,\mathbb{B}^n) \cap \Omega, \{\mathbf{x}^k\}_{k\in\mathbb{N}} \in \mathcal{R}(\mathbf{x}^0) \implies \mathbb{P}(\mathbf{x}^k(\psi) \in \mathcal{A} + \varepsilon\,\mathbb{B}_o^n,\ k \in \mathbb{N}) \geqslant 1 - \sigma.$$

The set $\mathcal{A} \subset \mathbb{R}^n$ is *attractive in probability for* (9) if for each $\varepsilon \in \mathbb{R}_{>0}$, $\sigma \in \mathbb{R}_{>0}$ and $\Delta \in \mathbb{R}_{>0}$, there is $K \in \mathbb{R}_{>0}$ such that

$$\mathbf{x}^0 \in (\mathcal{A} + \Delta\,\mathbb{B}^n) \cap \Omega, \{\mathbf{x}^k\}_{k\in\mathbb{N}} \in \mathcal{R}(\mathbf{x}^0) \implies \mathbb{P}(\mathbf{x}^k(\psi) \in \mathcal{A} + \varepsilon\,\mathbb{B}_o^n,\ k \geqslant K) \geqslant 1 - \sigma.$$

Hence, the set $\mathcal{A} \subset \mathbb{R}^n$ is *asymptotically stable in probability for* (9) if it is both uniformly attractive and stable in probability for (9). If the set $\mathcal{A}$ is asymptotically stable in probability for (9), then we say that (9) *converges in probability to* $\mathcal{A}$.

Clearly, asymptotic stability in probability of the set of all solutions of the PMP (1) is a desirable property for a minimisation algorithm since it implies that, if one already has a solution that is close to the optimum, then the iterations of the algorithm do not perturb such an optimality, and that the algorithm converges in probability to the solution of the PMP, even when the initial condition of the algorithm is far from optimality. In the following theorem, we show that the set of all the solutions to the PMP (1) is indeed asymptotically stable in probability for the stochastic inclusion (9).

Theorem 3.1: *Let the assumptions of Lemma* 3.1 *hold. If, additionally,* $p < 1$ *and there exists* $\nu^\star \in \mathbb{R}_{>0}$ *such that the set* $(\mathcal{A} + \nu\mathbb{B}^n) \cap \Omega$ *has nonzero measure for all* $\nu \in (0, \nu^\star)$, *then the set* $\mathcal{A}$ *is asymptotically stable in probability from* Ω.

A proof of Theorem 3.1 is given in Appendix A.2.

Remark 3.1 (About the assumptions in Theorem 3.1): It may be worth to observe that Theorem 3.1 requires no convexity assumption for guaranteeing global probabilistic convergence. Also observe that the technical hypothesis that $(\mathcal{A} + \nu\mathbb{B}^n) \cap \Omega$ has nonzero measure holds in several standard situations; for instance, if the set Ω is a *regular closed set* (i.e. the closure of its interior coincides with the set itself) then the hypothesis holds. In words, if Ω is everywhere full dimensional then $(\mathcal{A} + \nu\mathbb{B}^n) \cap \Omega$ has positive measure. One situation in which this hypothesis may fail is when the constraints include linear equality constraints among the variables, which in practice reduce the dimensionality of the feasible set. This situation, however, is easily circumvented via the standard trick of eliminating the equality constraints and rewriting the problem in a reduced set of free variables. Further, the assumptions of Lemma 3.1 are satisfied whenever the feasible set Ω is compact. In this regard, it may be observed that, since in most engineering problem the variables always have physical limits, a 'compactifying constraint' of the form $\|\mathbf{x}\|_2^2 \leqslant M$, for some large $M > 0$, can typically be added to the constraints of the problem without altering its optimal set. In such case, and assuming that the optimal set is nonempty, the hypotheses of Lemma 3.1 remain always satisfied.

Under the above mild assumptions, the technique we proposer allows us to determine an arbitrarily good approximation of the solution to the PMP (1), with probability arbitrarily close to 1, in a finite number of iterations of Equation (9). Moreover, the proof of Theorem 3.1 establishes also robustness of the proposed method. In fact, by Grammatico et al. (2013), the existence of a smooth Lyapunov function establishes (semiglobal, practical) robustness with respect to small constant perturbations, such as those arising from numerical errors.

Remark 3.2 (On the generality of the method): We observe that the polynomial nature of the objective function f and of $h_1, \ldots, h_m$ has not been exploited in the proof of Theorem 3.1. Indeed, this theorem holds even when the proposed minimisation technique is applied to problems in which f and $h_1, \ldots, h_m$ are general continuous functions. In practice, however, application of the algorithm in the general setting would require to have at one's disposal an 'oracle' that returns, at each step, a global optimal solution to the univariate minimisation problem (9a). Solving such a problem may be non trivial for non-polynomial instances, while it is solvable quite efficiently in the polynomial case, see Section 4.3. Therefore, the theoretical properties of Theorem 3.1 remain valid also for general continuous functions, but the practical numerical efficiency of the method may be considerably worse in the case of non-polynomial problem instances.

3.1 Theoretical probabilistic convergence bounds

In this section, we derive a theoretical result that provides probabilistic bounds on the number of iterations required by the proposed algorithm for bringing the value of the objective function f below a given threshold.

Let the assumptions of Theorem 3.1 hold and let $G : \mathbb{R}^n \times \mathbb{S}^n \rightrightarrows \mathbb{R}^n$ be the set-valued mapping given in (10). Then, given $p \in \mathbb{R}_{\geqslant 0}$, $p < 1$, and an open set $\mathcal{O} \subset \mathbb{R}^n$, for each $\mathbf{x} \in \Omega$, define $\ell_{\subset\mathcal{O}}(0, \mathbf{x}) \doteq 1$, $\ell_{\cap\mathcal{O}}(0, \mathbf{x}) \doteq 0$, and, for each $(\mathbf{x}, k) \in \Omega \times \mathbb{N}$,

define $\ell_{\subset\mathcal{O}}(k+1,\mathbf{x})$ and $\ell_{\cap\mathcal{O}}(k+1,\mathbf{x})$ as

$$\ell_{\subset\mathcal{O}}(k+1,\mathbf{x}) \doteq \int_{\mathbb{S}^n} \min_{\mathbf{g}\in G(\mathbf{x},\mathbf{s})} \mathbb{I}_{\mathcal{O}}(\mathbf{g})\ell_{\subset\mathcal{O}}(k,\mathbf{x})\,\mu(\mathrm{d}\mathbf{s}), \quad (11a)$$

$$\ell_{\cap\mathcal{O}}(k+1,\mathbf{x}) \doteq \int_{\mathbb{S}^n} \min_{\mathbf{g}\in G(\mathbf{x},\mathbf{s})} \mathbb{J}_{\mathcal{O}}(k,\mathbf{g})\,\mu(\mathrm{d}\mathbf{s}), \quad (11b)$$

where $\mathbb{J}_{\mathcal{O}}(k,\mathbf{g}) \doteq [\mathbb{I}_{\mathcal{O}}(\mathbf{g}), \mathbb{I}_{\mathbb{R}^n\setminus\mathcal{O}}(\mathbf{g})\ell_{\cap\mathcal{O}}(k,\mathbf{g})]$. The functions $\ell_{\subset\mathcal{O}}$ and $\ell_{\cap\mathcal{O}}$ bound the probabilities that the set $\mathcal{O}$ is strongly invariant and strongly recurrent for (9), respectively; see Possieri and Teel (2017) for the formal definition of these two properties. The next theorem shows how these functions can be used to provide probabilistic bounds on the convergence of the given method.

Theorem 3.2: *Let the assumptions of Theorem* 3.1 *hold and let* $\eta \in \mathbb{R}_{>0}$ *be given. Define the sets*

$$\mathcal{O}_1 = \mathbb{R}^n \setminus \{\mathbf{x}\in\Omega : f(\mathbf{x}) \leqslant \eta\},$$
$$\mathcal{O}_2 = \operatorname{int}\{\mathbf{x}\in\Omega : f(\mathbf{x}) \leqslant \eta\},$$

and assume that they are nonempty. Then, for all $K \in \mathbb{N}$, *for all* $\mathbf{x}^0 \in \Omega$, *and for all solutions* $\{\mathbf{x}^k\}_{k\in\mathbb{N}} \in \mathcal{R}(\mathbf{x}^0)$, *one has*

$$\ell_{\cap\mathcal{O}_2}(K,\mathbf{x}^0) \leqslant \mathbb{P}(f(\mathbf{x}^i) \leqslant \eta,\ \forall i \geqslant K) \leqslant 1 - \ell_{\subset\mathcal{O}_1}(K,\mathbf{x}^0).$$

A proof of Theorem 3.2 is given in Appendix A.3. The relation given in Theorem 3.2 can be used, in principle, to determine the expected number of iterations K needed to diminish the objective function f to values lower than or equal to η. Note that, differently from the result given in Theorem 2 of Calafiore and Possieri (2018), the bounds provided by Theorem 3.2 can be computed without requiring prior knowledge of the set $\mathcal{A}$, which was a rather unrealistic requirement of Calafiore and Possieri (2018). Furthermore, by Lemmas 1 and 2 of Possieri and Teel (2017), the maps $k \mapsto \ell_{\cap\mathcal{O}_2}(k,\mathbf{x}^0)$ and $k \mapsto 1-\ell_{\subset\mathcal{O}_1}(k,\mathbf{x}^0)$ are monotonically increasing and, under the assumptions of Theorem 3.2, we have that $\lim_{k\to\infty}\ell_{\cap\mathcal{O}_2}(k,\mathbf{x}^0) = \lim_{k\to\infty} 1-\ell_{\subset\mathcal{O}_1}(k,\mathbf{x}^0) = 1$ for all $\mathbf{x}^0 \in \Omega$, by Theorem 3.1. Therefore, in theory, given $\eta > f^\star$ and $\mathbf{x}^0$, such functions could be used to compute lower and upper bounds on the number of iterations required to lower the objective function below η with probability arbitrarily close to 1. Notice, however, that the bounds in Theorem 3.2 have mainly a theoretical, rather than practical, interest, since they are generally hard to compute numerically. In practical implementations of the algorithm, as described in Section 4, we shall instead rely on a stopping criterion for the iterations (9) in order to exit the loop.

4. Implementation and complexity analysis

In this section, we present the complete polynomial minimisation algorithm and its implementation for the solution of the PMP (1). In view of Lemma 3.1, the method given in (9) essentially consists in applying the following Algorithm 1.

By Theorem 3.1, under some mild assumptions, the proposed algorithm, if run indefinitely (i.e. with $L = +\infty$), would converge with probability one to the solution of the PMP (1).

Algorithm 1 Random coordinate descent with transverse directions

Input: the PMP (1), a scalar $p \in [0,1)$, the probability mass function $\varpi : \{1,\dots,n\} \to [0,1]$, $\mathbf{x}^0 \in \Omega$, a numerical tolerance $\varepsilon \in \mathbb{R}_{>0}$, and a positive integer $L \in \mathbb{N}$

Output: estimates $\hat{\mathbf{x}}^\star$ and $\hat{f}^\star$ of $\mathbf{x}^\star$ and $f^\star$, respectively

1: $k = 0$, $\varkappa = 0$
2: **while** $\varkappa \leqslant L$ **do**
3: $\quad k \leftarrow k+1$
4: $\quad$ pick a random number r in Uni(0, 1)
5: $\quad$ **if** $r \leqslant p$ **then**
6: $\quad\quad$ pick $\mathbf{s} \sim \sum_{i=1}^n \varpi(i)\delta(\mathbf{e}_i)$
7: $\quad$ **else**
8: $\quad\quad$ pick $\mathbf{s} \sim \mathrm{Uni}(\mathbb{S}^n)$
9: $\quad$ define $\tilde{f}(\lambda) \doteq f(\mathbf{x}+\lambda\,\mathbf{s})$ and $\tilde{\mathbf{h}}(\lambda) \doteq \mathbf{h}(\mathbf{x}+\lambda\,\mathbf{s})$
10: $\quad$ let $\lambda^\star$ be a solution to the univariate PMP (A2)
11: $\quad$ let $\mathbf{x}^k = \mathbf{x}^{k-1} + \lambda^\star\,\mathbf{s}$
12: $\quad$ **if** $|f(\mathbf{x}^k) - f(\mathbf{x}^{k-1})| \leqslant \varepsilon\,(1+|f(\mathbf{x}^k)|)$ **then**
13: $\quad\quad \varkappa \leftarrow \varkappa + 1$
14: $\quad$ **else**
15: $\quad\quad \varkappa \leftarrow 0$
16: **return** $\hat{\mathbf{x}}^\star = \mathbf{x}^k$ and $\hat{f}^\star = f(\mathbf{x}^k)$

In practice, $\mathbf{x}^k$ gets arbitrarily close to $\mathcal{A}$ with probability arbitrarily close to 1 if a sufficient number of iterations are carried out, but there need not exist $K \in \mathbb{N}$ such that $\mathbf{x}^K \in \mathcal{A}$ exactly. Therefore, a stopping criterion is required to interrupt the loop in Algorithm 1 when the iterations get sufficiently close to the optimal value. In particular, Algorithm 1 is interrupted as soon as the relative decrement $\frac{|f(\mathbf{x}^k)-f(\mathbf{x}^{k-1})|}{1+|f(\mathbf{x}^k)|}$ is smaller than a given tolerance ε for at least L consecutive iterations. Such a stopping criterion is commonly used in numerical practice when dealing with asymptotic techniques, see, e.g.Kearfott and Walster (2000).

In order to execute Algorithm 1, one has to perform the following three operations:

- determine $\mathbf{x}^0 \in \Omega$ for initializing the sequence $\mathbf{x}^k$;
- compute $\tilde{f}$ and $\tilde{\mathbf{h}}$ at Step 9;
- solve an univariate PMP at Step 10.

The main objective of the remainder of this section is to show how such operations can be carried out in practice, also providing a detailed complexity analysis of each operation. In particular, in Section 4.1, we show how Algorithm 1 itself can be used, if needed, to determine an initial feasible point $\mathbf{x}^0 \in \Omega$. In Section 4.2, we show how to practically compute the polynomials $\tilde{f}$ and $\tilde{\mathbf{h}}$, and, in Section 4.3, we discuss a polynomial-time algorithm to solve an univariate PMP.

4.1 Determining an initial feasible point

In order to use Algorithm 1 for solving the PMP (1), an initial feasible point $\mathbf{x}^0 \in \Omega$ must be available. To this end, there are three possibilities: (*i*) if $\Omega = \mathbb{R}^n$, then the initial feasible point

can be chosen simply as $\mathbf{x}^0 = \mathbf{0} \in \mathbb{R}^n$; (*ii*) if Ω is a 'simple' subset of $\mathbb{R}^n$, such as a norm ball, a parallelotope, a polyhedron or a spectrahedron, then the initial feasible point can be determined by direct geometric considerations, or by solving a preliminary convex problem, such as a linear program in the case of a polyhedron, or an SDP in the case of a spectrahedron; (*iii*) if Ω is a generic intersection of the feasibility sets of polynomial inequalities, then finding $\mathbf{x}^0 \in \Omega$ corresponds to determining a solution for such a system of inequalities, which can be performed by solving an auxiliary PMP of the form

$$\left|\begin{array}{ll} \min & \epsilon, \\ \text{s.t.} & h_i(\mathbf{x}) - \epsilon \leqslant 0, \quad i = 1, \dots, m, \\ & \|\mathbf{x}\|_2^2 \leqslant M, \\ & |\epsilon| \leqslant M. \end{array}\right. \tag{12}$$

where $M \in \mathbb{R}_{>0}$ is a sufficiently large constant, and the additional constraint $\|\mathbf{x}\|_2^2 \leqslant M$ imposes compactness of the feasible set. Adding such constraint is not restrictive in practical engineering problems, as discussed in Remark 3.1.

While it is worth to observe that, by and large, the most common situations in control applications fall in the 'easy' cases (*i*) and (*ii*), we show next that in all other cases an initial point $\mathbf{x}^0$ can be found by applying Algorithm 1 to problem (12). To this end, let $\mathbf{x}_e \doteq [\mathbf{x}^\top \ \epsilon]^\top$, and let

$$\begin{aligned} \Omega_e \doteq \{&\mathbf{x}_e \in \mathbb{R}^{n+1} : h_i(\mathbf{x}) - \epsilon \\ &\leqslant 0, \ i = 1, \dots, m, \|\mathbf{x}\|_2^2 \leqslant M, |\epsilon| \leqslant M\} \end{aligned} \tag{13}$$

be the feasible set of the (12). We observe that $\hat{\mathbf{x}}_e^0 \doteq [\mathbf{0}^\top \ \max\{h_1(\mathbf{0}), \dots, h_m(\mathbf{0})\}]^\top$ is in Ω_e for each $\mathbf{h} \in \mathbb{R}^m[\mathbf{x}]$. Therefore, an initial feasible point for the PMP (12) is always known. Moreover, the set Ω_e is compact for any $\mathbf{h} \in \mathbb{R}^m[\mathbf{x}]$ and each $M \in \mathbb{R}_{>0}$. The next theorem shows that, under some mild assumptions, one can determine an initial feasible point for the PMP (1) by applying Algorithm 1 to (12).

Theorem 4.1: *For each (sufficiently small) $\tau \in \mathbb{R}_{\geqslant 0}$, define*

$$\breve{\Omega}_{e,\tau} \doteq \Omega_e \cap \{\mathbf{x}_e \in \mathbb{R}^{n+1} : \epsilon \leqslant -\tau\}.$$

Assume that there exists $\tau^\star \in \mathbb{R}_{>0}$ such that the set $\breve{\Omega}_{e,\tau^\star}$ has nonzero measure. For each $\mathbf{x}_e^0 \in \Omega_e$ and each $\sigma \in \mathbb{R}_{>0}$, letting $\{\mathbf{x}_e^k\}_{k\in\mathbb{N}}$, $\mathbf{x}_e^k = [(\mathbf{x}^k)^\top \ \epsilon]^\top$, be the sequence obtained by applying Algorithm 1 to the PMP (12) with initial guess $\mathbf{x}_e^0$, there exists a finite integer K such that

$$\mathbb{P}(h_i(\mathbf{x}^k) \leqslant 0, i = 1, \dots, m, \ k \geqslant K) \geqslant 1 - \sigma.$$

A proof of Theorem 4.1 is given in Appendix A.4. In view of such a theorem, it can be easily derived that if there exists a strictly feasible point $\mathbf{x}^* \in \Omega$ (i.e. such that $h_i(\mathbf{x}^*) < 0$, $i = 1, \dots, m$), then Algorithm 1 applied to the PMP (12) with $\hat{\mathbf{x}}_e^0 \in \Omega_e$ as initial guess is capable of determining a feasible point in finite time with probability arbitrarily close to 1. Furthermore, the iteration of Steps 3–11 of such an algorithm can be interrupted as soon as an $\mathbf{x}_e^* = [(\mathbf{x}^*)^\top \ \epsilon^*]^\top$ such that $\epsilon^* \leqslant 0$ has been found.

4.2 Step 9: polynomial substitution

We next provide some details on how Step 9 of Algorithm 1 can be carried out. If the vector $\mathbf{s}$ to be used for computing $\tilde{f}$ and $\tilde{g}$ is one of the coordinate directions $\mathbf{e}_1, \dots, \mathbf{e}_n$, then those polynomials can be computed in a quite efficient way; the case when $\mathbf{s}$ is not a coordinate direction instead typically requires more computational effort; for this reason we treat next the cases $\mathbf{s} = \mathbf{e}_i$ and $\mathbf{s} \neq \mathbf{e}_i$ separately.

4.2.1 The case $\mathbf{s} = \mathbf{e}_i$

First, note that given a monomial $\mathbf{x}^{\boldsymbol{\alpha}} \in \mathbb{R}[\mathbf{x}]$, it can be equivalently rewritten as $\mathbf{x}^{\boldsymbol{\alpha}} = \mathbf{x}_{-i}^{\boldsymbol{\alpha}_{-i}} x_i^{\alpha_i}$, for each $i \in \{1, \dots, n\}$. Similarly, given $f \in \mathbb{R}[\mathbf{x}]$, the ith *coordinate-wise polynomial* of f at $\mathbf{x}_{-i}$ is the univariate polynomial in x_i with coefficients in $\mathbb{R}[\mathbf{x}_{-i}]$ that is obtained by considering all values in $\mathbf{x}_{-i}$ being fixed, and only the ith variate x_i as variable, i.e.

$$f_i(x_i) = \sum_{\boldsymbol{\alpha}\in\mathcal{E}} c_\alpha \mathbf{x}^{\boldsymbol{\alpha}} = \sum_{\boldsymbol{\alpha}\in\mathcal{E}} c_\alpha \mathbf{x}_{-i}^{\boldsymbol{\alpha}_{-i}} x_i^{\alpha_i} = \sum_{\boldsymbol{\alpha}\in\mathcal{E}} \tilde{c}_\alpha(\mathbf{x}_{-i}) x_i^{\alpha_i}$$

where $\tilde{c}_\alpha(\mathbf{x}_{-i}) \doteq c_\alpha \mathbf{x}_{-i}^{\boldsymbol{\alpha}_{-i}} \in \mathbb{R}[\mathbf{x}_{-i}]$. The ith coordinate-wise polynomial f_i is useful when we update the estimate of the solution to the PMP (1) with $\mathbf{s} = \mathbf{e}_i$: determining $\tilde{f}(\lambda)$ and $\tilde{\mathbf{h}}(\lambda)$ essentially consists in computing $m+1$ coordinate-wise polynomials. The next proposition, whose proof is given in Appendix A.5, provides an upper bound on the number of elementary operations (sum and products) that have to be carried out to determine such polynomials.

Proposition 4.1: *Let $\mathbf{x} = [x_1 \ \dots \ x_n]^\top$, let $f \in \mathbb{R}[\mathbf{x}]$ and $\mathbf{h} = [h_1 \ \dots \ h_m] \in \mathbb{R}^m[\mathbf{x}]$ be given, and let $d \doteq \max(\deg(f), \deg(h_1), \dots \deg(h_m))$. If $\mathbf{s}^k = \mathbf{e}_i$, then the computational complexity of Step 9 of Algorithm 1 is upper bounded by $\mathfrak{c}_{\mathbf{e}_i} = d\,(d+1)\,n\,(m+1)\binom{d+n}{d+1}$.*

4.2.2 The case $\mathbf{s} \neq \mathbf{e}_i$

In order to obtain an explicit polynomial representation of $\tilde{f}(\lambda)$, we consider the jth monomial $\xi_{\boldsymbol{\alpha}}(\lambda) \doteq (\mathbf{x}^k + \lambda \mathbf{s}^k)^{\boldsymbol{\alpha}} = \prod_{i=1}^n (x_i^k + \lambda s_i^k)^{\alpha_i}$. Each term in the product above can be written as $(x_i^k + \lambda s_i^k)^{\alpha_i} = \sum_{j=0}^{\alpha_i} \binom{\alpha_i}{j} (x_i^k)^{\alpha_i - j} (s_i^k)^j \cdot \lambda^j = \sum_{j=0}^{\alpha_i} \zeta_{i,j}^{\alpha_i} (x_i^k, s_i^k) \lambda^j$, where $\zeta_{i,j}^{\alpha_i}(x_i^k, s_i^k) \doteq \binom{\alpha_i}{j} (x_i^k)^{\alpha_i - j} (s_i^k)^j$. Therefore, by defining the row vector $\boldsymbol{\zeta}_i^{\alpha_i}(x_i^k, s_i^k) \doteq [\zeta_{i,0}^{\alpha_i}(x_i^k, s_i^k) \ \dots \ \zeta_{i,\alpha_i}^{\alpha_i}(x_i^k, s_i^k)]$, the product in the expression for $\xi_{\boldsymbol{\alpha}}(\lambda)$ can be readily obtained by taking the convolution of the $\boldsymbol{\zeta}_i^{\alpha_i}$'s, i.e. letting $\boldsymbol{\omega}_{\boldsymbol{\alpha}}(x_i^k, s_i^k) = [\omega_{\boldsymbol{\alpha},0}(x_i^k, s_i^k) \ \dots \ \omega_{\boldsymbol{\alpha},|\boldsymbol{\alpha}|}(x_i^k, s_i^k)]$, $\omega_{\boldsymbol{\alpha}}(x_i^k, s_i^k) \doteq \boldsymbol{\zeta}_1^{\alpha_1}(x_i^k, s_i^k) * \boldsymbol{\zeta}_2^{\alpha_2}(x_i^k, s_i^k) * \cdots * \boldsymbol{\zeta}_n^{\alpha_n}(x_i^k, s_i^k)$, where $*$ denotes the convolution operator, we have that $\xi_{\boldsymbol{\alpha}}(\lambda) = \sum_{i=0}^{|\boldsymbol{\alpha}|} \omega_{\boldsymbol{\alpha},i}(x_i^k, s_i^k)\lambda^i$, and hence

$$\tilde{f}(\lambda) = \sum_{\boldsymbol{\alpha}\in\mathcal{E}} \sum_{i=0}^{|\boldsymbol{\alpha}|} c_{\boldsymbol{\alpha}} \omega_{\boldsymbol{\alpha},i}(x_i^k, s_i^k)\lambda^i. \tag{14}$$

This construction for $\tilde{f}$ can be used also to compute each entry of the vector $\tilde{\mathbf{h}}$. The next proposition, whose proof is given in Appendix A.6, provides an upper bound on the computation complexity of Step 9 of Algorithm 1 in the case $\mathbf{s} \neq \mathbf{e}_i$.

Proposition 4.2: *Let* $\mathbf{x} = [x_1 \ \dots \ x_n]^\top$, *let* $f \in \mathbb{R}[\mathbf{x}]$ *and* $\mathbf{h} = [h_1 \ \dots \ h_m] \in \mathbb{R}^m[\mathbf{x}]$ *be given, and let* $d \doteq \max(\deg(f), \deg(h_1), \dots \deg(h_m))$. *If* $\mathbf{s}^k \neq \mathbf{e}_i$, *then the computational complexity of Step* 9 *of Algorithm* 1 *is upper bounded by* $\mathfrak{c}_\mathbf{s} = \frac{1}{2}(d+1)^2(4d^3 + 5d\,n + 5d + 10)\,(m+1)\binom{d+n}{d+1}$.

Remark 4.1 (On complexity of coordinate and transverse updates): By computing the ratio between the computational complexities $\mathfrak{c}_{\mathbf{e}_i}$ and $\mathfrak{c}_\mathbf{s}$, one obtains that

$$\frac{\mathfrak{c}_{\mathbf{e}_i}}{\mathfrak{c}_\mathbf{s}} = \frac{2\,d\,(d+1)\,n\,(m+1)}{(d+1)^2(4d^3 + 5d\,n + 5d + 10)\,(m+1)} = \frac{\mathcal{O}(d^2\,n\,m)}{\mathcal{O}(d^3 n\,m + d^5 m)}.$$

This shows that, when dealing with PMP involving polynomials with high degree d, it is more computationally efficient to select a coordinate direction rather than a transverse one.

4.3 Step 10: solving an univariate PMP

In order to execute Step 10 of Algorithm 1, an univariate PMP has to be solved. The following proposition guarantees that Algorithm 2 allows to determine a solution, if any, to the PMP (15) and characterises its computational complexity.

Algorithm 2 Solution to an univariate PMP

Input: $f \in \mathbb{R}[\lambda]$ and $\mathbf{h} = [h_1 \ \dots \ h_m]^\top \in \mathbb{R}^m[\lambda]$
Output: if any, a solution to the univariate PMP

$$\left| \begin{array}{ll} \min & f(\lambda), \\ \text{s.t.} & h_i(\lambda) \leqslant 0, \quad i = 1, \dots, m. \end{array} \right. \tag{4}$$

1: let $f'(\lambda) = \frac{d}{d\lambda} f(\lambda)$
2: compute the real roots $r_1, \dots, r_\theta$ of $f', h_1, \dots, h_m$
3: compute $f_i \doteq f(r_i)$, $i = 1, \dots, \theta$
4: sort $f_1, \dots, f_\theta$ in ascending order
5: sort $r_1, \dots, r_\theta$ according to the sorted $f_1, \dots, f_\theta$
6: **if** there exists $\bar{r} \in \mathbb{R}$ such that $h_1(\bar{r}) \leqslant 0, \dots, h_m(\bar{r}) \leqslant 0$ and $f(\bar{r}) = f_1 - 2|f_1|$ **then**
7: **return** the PMP (15) is unbounded
8: **else**
9: **for** $j = 1$ **to** θ **do**
10: **if** $h_1(r_j) \leqslant 0, \dots, h_m(r_j) \leqslant 0$ **then**
11: **return** r_j and f_j
12: **return** the PMP (15) is unfeasible

Proposition 4.3: *Let* $f \in \mathbb{R}[\lambda]$ *and* $\mathbf{h} = [h_1 \ \dots \ h_m]^\top \in \mathbb{R}^m[\lambda]$ *be given and let* $d \doteq \max\{\deg(f), \deg(h_1), \dots, \deg(h_m)\}$. *Algorithm* 2 *returns a solution, if any, to the PMP* (1). *Furthermore, its computational complexity is upper bounded by* $d^6 m + 2d^6 - 6d^5 + 15d^4 - 20d^3 + 2d^2m^2 + 6d^2 m + 17d^2 + 2dm^2 + 4dm - 5d - 2m - 1$.

Proof: The effectiveness of Algorithm 2 follows directly from Menini et al. (2018a, Lem. 5) and from the fact that the considered problem is scalar. Note that the polynomial f' can be determined by carrying out d elementary operations and hence the roots $r_1, \dots, r_\theta$ can be computed by carrying out $(d-1)^6 + m\,d^6$ elementary operations, see Collins and Akritas (1976). Note that, since θ is the total number of real roots of f' and $h_1, \dots, h_m$, we have that $\theta \leqslant m\,d + d - 1$. Thus the total number of operations required to compute $f_1, \dots, f_\theta$ is upper bounded by $2(d+1)(m\,d + d - 1)$, see Borwein and Erdélyi (2012). In order to carry out Step 6, one has to find all the real roots of the polynomial $f - f_1 - 2|f_1|$ and has to evaluate the polynomials $h_1, \dots, h_m$ on such roots. The computational complexity of these operations is upper bounded by d^6 and $2\,d\,(d+1)\,m$, respectively. Similarly, the computational complexity of Steps 9–11 is upper bounded by $2\,(d+1)\,m(m\,d + d - 1)$, thus concluding the proof. ■

The complexity of each iteration of Algorithm 1 has been characterised by means of Propositions 4.1, 4.2, and 4.3. The overall convergence, in terms of number of iterations needed for bringing of the objective function f below a given threshold, depends in a complex way on the problem structure and on the probability p, as shown theoretically in Theorem 3.2. The next two sections provide numerical evidence of the practical performance of the proposed methodology.

5. Numerical tests

In this section, we demonstrate the effectiveness of the given minimisation procedure through several numerical tests.

Randomly generated experiments have been carried out to compare the performances of our `polMin` toolbox (which implements the procedure given in (9) through the tools outlined in Section 4) with the nonlinear optimisation package `IPOPT` (Wächter & Biegler, 2006), the `Matlab` function `fmincon` (Byrd et al., 2000), the toolbox `GloptiPoly` (Henrion & Lasserre, 2003), the toolbox `SOSTOOLS` (Papachristodoulou et al., 2013) and the toolbox `Yalmip` (Löfberg, 2009). In particular, in each experiment, random matrices $\mathbf{A} \in \mathbb{R}^{\nu \times \nu}$ and $\mathbf{B} \in \mathbb{R}^{\nu \times \nu}$ with entries uniformly distributed in $[-100, 100]$ have been generated and it has been assumed that $\mathbf{X} = \mathrm{diag}(x_1, \dots, x_\nu)$. Then, the matrix $\mathbf{A}_{\mathrm{cl}}$ has been defined as $\mathbf{A}_{\mathrm{cl}} \doteq \mathbf{A} + \mathbf{B}\,\mathbf{X}$ and the corresponding Routh polynomials $\wp_1, \dots, \wp_\nu \in \mathbb{R}[\mathbf{x}]$ have been computed. Finally, the following PMP has been considered

$$\left| \begin{array}{ll} \min & x_1^2 + \dots + x_\nu^2, \\ \text{s.t.} & \wp_i(\mathbf{x}) \geqslant 0.1, \quad i = 1, \dots, \nu, \end{array} \right. \tag{16}$$

which corresponds to a decentralised static feedback problem. It is worth noticing that although the objective function of the PMP (16) is rather simple, the polynomials $\wp_i(\mathbf{x})$ may be very complex (e.g.for $\nu = 5$, one of these polynomials is given by the sum of 1623 monomials with random coefficients), thus making the problem of computing the solution to the PMP (16) very complex; see Table 1.

All the experiments have been carried out on a laptop with an Intel i5 CPU (2.4 GHz) and 8 GB, 1600 MHz, DDR3 RAM. In each test, the same set of 100 PMPs, randomly generated as detailed above, has been used as input to either:

Table 1. Average execution times (s).

	ν				
Method	1	2	3	4	5
(a1)	0.825	1.258	1.570	6.102	454.8
(a2)	**0.707**	**0.981**	1.318	5.470	195.17
(a3)	0.747	1.002	**1.281**	4.395	187.65
(a4)	0.765	1.092	1.325	**4.197**	185.48
(a5)	0.775	1.093	1.326	4.258	**184.22**
(b)	0.0115	0.0215	0.0428	0.197	4.067
(c)	0.031	0.043	0.11	0.36	3.11
(d)	0.264	0.256	0.3005	0.292	–
(e)	0.226	0.544	0.871	–	–
(f)	0.724	2.128	10.405	–	–

The selection of p that led to the best execution time has been highlighted in bold.

(a) the `MATLAB` toolbox `polMin`, by using the functions `findFeas` and `polMin` (which implement Algorithms 1 and 2), with $\varpi(i) = \frac{1}{\nu}$, $i = 1, \ldots, \nu$, $\varepsilon = 10^{-4}$, $L = 100$, (a1)$p = 0.99$; (a2)$p = 0.97$; (a3)$p = 0.95$; (a4)$p = 0.93$; (a5)$p = 0.91$;
(b) the nonlinear optimisation package `IPOPT` with gradient and Hessian information;
(c) the function `fmincon` with gradient information;
(d) the `MATLAB` toolbox `GloptiPoly`, interfaced with the solver `MOSEK` (MOSEK ApS, 2017), using the functions `msdp` and `msol`;
(e) the `MATLAB` toolbox `Yalmip`, interfaced with the solvers `BMIBNB` and `MOSEK`, using the function `optimize`;
(f) the `MATLAB` toolbox `SOSTOOLS`, interfaced with the solver `SDTP3` (Tütüncü et al., 2003), using the function `findbound`.

A total of 5 tests (corresponding to 5000 numerical experiments) have been considered. Each column of Table 1 corresponds to the same $\nu \in \mathbb{N}$ and reports the average execution time of the methods (a)–(f). Just the cases in which the method returned a feasible solution have been considered in the computation of the execution times.

Table 2 reports the success rate of each method (an experimental test has been considered successful when a feasible solution to the PMP (16) is returned).

Finally, let $(f^\star_{(a1)}, \mathbf{x}^\star_{(a1)})$, $(f^\star_{(a2)}, \mathbf{x}^\star_{(a2)})$, $(f^\star_{(a3)}, \mathbf{x}^\star_{(a3)})$, $(f^\star_{(a4)}, \mathbf{x}^\star_{(a4)})$, $(f^\star_{(a5)}, \mathbf{x}^\star_{(a5)})$, $(f^\star_{(b)}, \mathbf{x}^\star_{(b)})$, $(f^\star_{(c)}, \mathbf{x}^\star_{(c)})$, $(f^\star_{(d)}, \mathbf{x}^\star_{(d)})$, $(f^\star_{(e)}, \mathbf{x}^\star_{(e)})$, and $(f^\star_{(f)}, \mathbf{x}^\star_{(f)})$ be the solution of the PMP (16) obtained by using the methods (a1), (a2), (a3), (a4), (a5), (b), (c), (d), (e), and (f), respectively. Table 3 shows the percentage of cases in which $\mathbf{x}^\star_i \in \Omega$ and $f^\star_i \leqslant \min\{f^\star_{(a1)}, f^\star_{(a2)}, f^\star_{(a3)}, f^\star_{(a4)}, f^\star_{(a5)}, f^\star_{(b)}, f^\star_{(c)}, f^\star_{(d)}, f^\star_{(e)}, f^\star_{(f)}\}$, $i \in \{(a1), (a2), (a3), (a4), (a5), (b), (c), (d), (e), (f)\}$, i.e. the percentage of cases in which each method has obtained the best feasible solution with respect to the others (columns do not sum to one since $\arg\min\{f^\star_i, i \in \{(a1), (a2), (a3), (a4), (a5), (b), (c), (d), (e), (f)\}\}$ need not be a singleton).

As shown in Tables 1 and 2, the proposed technique, while being slower, succeeded to return a feasible solution to the PMP (16) in all the considered cases. In regard of execution times, it is fair to mention that our implementation of the proposed algorithm is an academic prototype, which was not finely tuned or optimised for speed, and that currently even relies on symbolic computations in some steps. The timing comparison is here made with respect to state-of-the-art commercial solvers, and it is therefore reasonable to believe that the execution time gap shown in Table 1 might be considerably reduced by a more refined implementation and coding of the proposed algorithm. More importantly, however, as shown in Table 3, the proposed tool was able to determine a good approximate solution to the PMP (16). It is worth noticing that, in all the tests, letting

$$f^\star = \min\{f^\star_{(a1)}, f^\star_{(a2)}, f^\star_{(a3)}, f^\star_{(a4)}, f^\star_{(a5)}, f^\star_{(b)}, f^\star_{(c)}, f^\star_{(d)}, f^\star_{(e)}, f^\star_{(f)}\},$$

we obtained that

$$\frac{|\max\{f^\star_{(a1)}, f^\star_{(a2)}, f^\star_{(a3)}, f^\star_{(a4)}, f^\star_{(a5)}\} - f^\star|}{1 + |f^\star|} \leqslant 0.0939,$$

i.e. in all the considered tests, the proposed minimisation method provides a solution that, when sub-optimal, is close to the optimal one (obtained by using another minimisation method with certified global optimality). Furthermore, the results reported in Tables 1 and 3 highlight the effects of the selection of the parameter p (which has been selected close to 1 in order to limit the number of steps in which a transverse direction is chosen) on the convergence of the algorithm, as detailed in the following remark, which provides suggestions on how to select such a parameter.

Remark 5.1: By analysing the results reported in Tables 1 and 3, the convergence time of the proposed algorithm reduces if p is decreased with the degree d of the involved polynomials. This is due to the fact that, since the PMP (1) is generically nonconvex, as d increases, it may be convenient to take more often transverse directions, which are selected with probability

Table 2. Success rate of each toolbox (in %).

	ν				
Method	1	2	3	4	5
(a1)	100	100	100	100	100
(a2)	100	100	100	100	100
(a3)	100	100	100	100	100
(a4)	100	100	100	100	100
(a5)	100	100	100	100	100
(b)	100	99	95	79	56
(c)	100	91	50	32	17
(d)	100	83	13	9	0
(e)	100	49	17	0	0
(f)	100	81	12	0	0

Table 3. Percentage of cases in which each method had the best performance. The selection of p that led to the best performance has been highlighted in bold.

	ν				
Method	1	2	3	4	5
(a1)	**100**	75	71	83	96
(a2)	**100**	**78**	71	86	94
(a3)	**100**	73	**83**	**87**	95
(a4)	**100**	75	69	82	**98**
(a5)	**100**	72	68	83	96
(b)	94	87	83	78	34
(c)	100	89	50	32	7
(d)	100	81	13	9	0
(e)	100	21	13	0	0
(f)	100	79	12	0	0

$1-p$. Indeed, in view of (A5), smaller values of p correspond to a larger expected decrease of the objective function $f(\mathbf{x})$; for further details, see Appendix A.2. However, selecting smaller values of p corresponds to an increased expected per-iteration effort due to the fact that transverse directions are selected more often than coordinate ones; for further details, see Remark 4.1. Therefore, since decreasing the parameter p implies at the same time an improved convergence rate and an increased computational complexity, it has to be selected via empirical tests as a trade-off between the per-iteration complexity and the overall convergence rate.

6. Control application

In this section, we use the proposed technique to solve an NMPC problem arising from an aerospace application.

A space rendezvous is a manoeuvre in which a spacecraft (called the chaser) must approach another spacecraft (called the target) to a very close distance. These manoeuvres are important ingredients of many present and future space missions, see, e.g. Weiss et al. (2015), Q. Li et al. (2017). Traditionally, rendezvous, docking and similar kinds of manoeuvres are carried out 'manually', by means of ad-hoc corrections finalised to compensate possible positioning errors. In this simulated example, space rendezvous is carried out automatically, by means of NMPC.

Consider a chaser spacecraft flying in a neighbourhood of a target. Suppose this latter is travelling in a circular orbit around a planet with angular speed ω. The dynamics of the chaser is described by the well known Hill–Clohessy–Wiltshire equations (Q. Li et al., 2017):

$$\begin{aligned} \ddot{z}_1 &= u_1 + 3\omega^2 z_1 + 2\omega \dot{z}_2, \\ \ddot{z}_2 &= u_2 - 2\omega \dot{z}_1, \\ \ddot{z}_3 &= u_3 - \omega^2 z_3, \end{aligned} \tag{17}$$

where z_i are the chaser coordinates in a target-centred reference frame and u_i are the accelerations given by the chaser thrusters. The value $\omega = 0.0011$ rad/s is assumed for the angular speed, corresponding to a low Earth orbit with a period of about 95 min. The control problem is to move the chaser to a point located near to a specific side of the target, to allow a subsequent docking manoeuvre. A state constraint must be imposed to avoid collisions between the two spacecraft. Input constraints are present as well, accounting for thruster saturation. Note that (17) is an (unstable) linear time-invariant system. However, as shown in the following, the control problem that is solved hereafter is nonlinear, since the state constraint is nonlinear (and nonconvex).

In order to solve the control problem, the system (17) has been discretised, using the zero-order hold method with sampling time equal to 5 s (the input is assumed to be constant between two sampling times). Then, the discretised equations have been written in the form (7), upon definition of the discretised state $\boldsymbol{\xi}(k) \doteq [z_1(k)\ \dot{z}_1(k)\ z_2(k)\ \dot{z}_2(k)\ z_3(k)\ \dot{z}_3(k)]^\top \in \mathbb{R}^6$ and input $\mathbf{u}(k) \doteq [u_1(k)\ u_2(k)\ u_3(k)]^\top \in \mathbb{R}^3$ vectors. Next, the constant reference state $\boldsymbol{\xi}_r \doteq [z_{r,1}\ 0\ z_{r,2}\ 0\ z_{r,3}\ 0]^\top$ has been defined, where $\mathbf{z}_r \doteq [z_{r,1}\ z_{r,2}\ z_{r,3}]^\top$ is the desired chaser arrival point. The objective function (8) has been considered, with

$$q(k, \boldsymbol{\xi}(k), \mathbf{u}(k)) \doteq \begin{cases} \mathbf{u}(k)^\top \mathbf{R}\, \mathbf{u}(k), & \text{if } k < N, \\ \tilde{\boldsymbol{\xi}}(k)^\top \mathbf{Q}\, \tilde{\boldsymbol{\xi}}(k), & \text{if } k = N, \end{cases} \tag{18}$$

where $\tilde{\boldsymbol{\xi}}(k) \doteq \boldsymbol{\xi}_r - \boldsymbol{\xi}(k)$ is the tracking error, while $\mathbf{Q} \in \mathbb{R}^{6\times 6}$ and $\mathbf{R} \in \mathbb{R}^{3\times 3}$ are positive semidefinite weight matrices. Note that, by construction, for $k = N$, the function q equals a weighted norm of the tracking error, whereas for $k < N$ it equals a weighted norm of the command input. This latter term is a measure of the command activity, which in turn provides a quantification of the propellant consumption. Minimizing the objective function in (8), with q given by (18), allows us to minimise the tracking error and, at the same time, limit the propellant consumption. The matrices $\mathbf{Q}$ and $\mathbf{R}$ can be used to trade-off between tracking performance and consumption.

Then, the set of admissible inputs has been defined as

$$\mathcal{U} \doteq \{\mathbf{u} \in \mathbb{R}^3 : g(\mathbf{u}) \doteq \|\mathbf{u}\|_\infty^2 \leqslant 1\}.$$

This set accounts for thruster saturation. The set of admissible states has been defined as

$$\mathcal{X} \doteq \{\boldsymbol{\xi} \in \mathbb{R}^6 : y(\boldsymbol{\xi}) \doteq z_{\min} - \|\mathbf{z}\|_2^2 \leqslant 0\},$$

where $\mathbf{z} \doteq [z_1\, z_2\, z_3]^\top$, and $z_{\min}$ is a minimum distance, imposed to avoid collisions between chaser and target.

A numerical simulation has been carried out to test the effectiveness of the proposed tool in solving the optimisation problem (8), by choosing the following parameters/matrices for the MPC algorithm:

$$\begin{aligned} &\boldsymbol{\xi}_r = [-15\ 0\ 0\ 0\ 0\ 0]^\top, \quad z_{\min} = 10, \quad N = 20, \\ &\boldsymbol{\xi}_0 = [11\ 0\ 0\ 0\ 0\ 0]^\top, \quad \mathbf{Q} = 10\, I, \quad \mathbf{R} = I, \end{aligned}$$

where all the distances are expressed in metres. Note that the PMP to be solved at each iteration involves 60 minimisation variables (namely, the three control inputs for 20 time steps) and 60 nonlinear and nonconvex polynomial constraints arising from the admissible sets $\mathcal{X}$ and $\mathcal{U}$, and hence it is hardly tractable with conventional methods.

Algorithm 1 has been used to solve the PMP (8) at each time step, thus determining the control input that has to be applied to the chaser. In order to carry out the simulation in a realistic scenario, we constrained the execution time of such an algorithm (including the time required to determine the initial feasible point) to 4 s, i.e. the algorithm has been interrupted after 4 s of execution (i.e. we forced an execution time smaller than the sampling time of the zero order holder) and the last estimate of the optimal solution to the PMP (8) has been used to design the control input. Furthermore, at all the time steps, a random noise uniformly distributed in $[-0.001,\ 0.001]$ has been added to the control input u (determined as detailed above) in order to take into account possible disturbances acting on the craft. Figure 1 depicts the results of such a simulation.

As shown in Figure 1, despite the limited amount of time that has been allotted to perform the optimisation, the complexity of the problem to be solved at each step, and the presence of the disturbance, the proposed technique has successfully solved

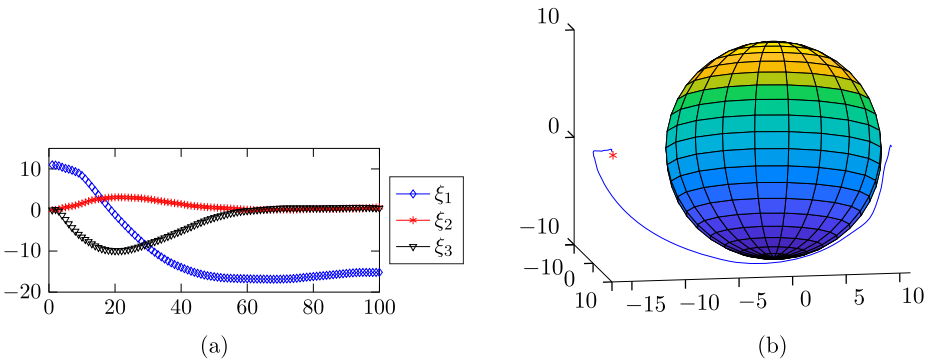

Figure 1. Application of the MPC algorithm to system (17): (a) discrete-time state and (b) continuous-time state.

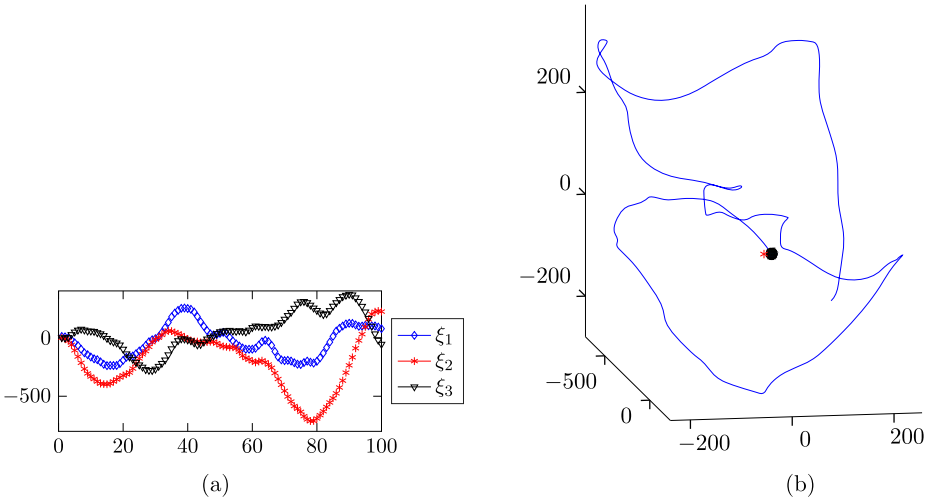

Figure 2. Application of the MPC algorithm to system (17) using `ga` to solve the PMP (8): (a) discrete-time state and (b) continuous-time state.

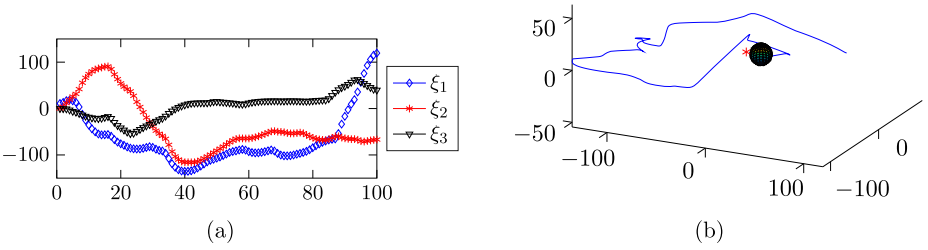

Figure 3. Application of the MPC algorithm to system (17) using `fmincon` to solve the PMP (8): (a) discrete-time state and (b) continuous-time state.

the rendezvous control problem by steering the craft to the desired position while avoiding collisions. For comparison, the same MPC algorithm has been applied using either the `Matlab` generic algorithm `ga` or the `Matlab` function `fmincon` to solve the PMP (8). As for the proposed minimisation technique, both the genetic algorithm `ga` and the function `fmincon` have been interrupted after 4 s of execution and the last estimate of the optimal solution to the PMP (8) has been used to design the control input. Figures 2 and 3 depict the results of these numerical simulations.

As shown in Figures 1–3, the algorithm `polMin` outperforms both `ga` and `fmincon` for the considered rendezvous control problem.

7. Conclusion

A random coordinate minimisation method with occasional transverse steps has been proposed for solving constrained polynomial minimisation problems arising in control. From

a methodological point of view, we proved convergence in probability to the global optimum for the proposed algorithm, and we established theoretical bounds on the rate of convergence in probability. From the practical point of view, the proposed algorithm performed satisfactorily in several numerical tests, also in comparison with existing techniques for polynomial optimisation. The algorithm is a descent algorithm, hence it can be interrupted at any time while providing a sub-optimal feasible solution and a corresponding upper bound on the optimal value. This feature is specially appealing when solving problems arising from NMPC, where availability of a sub-optimal solution in the allotted solution time is an essential requirement in order to ensure applicability of the technique.

Note

1. Available at the following link: https://github.com/Corrado-possieri/polMin

Disclosure statement

No potential conflict of interest was reported by the author(s).

ORCID

Corrado Possieri http://orcid.org/0000-0003-2528-3935

References

Alon, N., & Naor, A. (2006). Approximating the cut-norm via Grothendieck's inequality. *SIAM Journal on Computing*, *35*(4), 787–803. https://doi.org/10.1137/S0097539704441629

Bertsekas, D. P. (1999). *Nonlinear programming*. Athena Scientific.

Bertsekas, D. P., & Tsitsiklis, J. N. (1989). *Parallel and distributed computation: Numerical methods* (Vol. 23). Prentice Hall.

Bhatia, N. P., & Szegö, G. P. (2002). *Stability theory of dynamical systems*. Springer.

Bloemen, H. H. J., T. J. J. van den Boom, & Verbruggen, H. B. (2004). Optimization algorithms for bilinear model–based predictive control problems. *AIChE Journal*, *50*(7), 1453–1461. https://doi.org/10.1002/(ISSN)1547-5905

Borwein, P., & Erdélyi, T. (2012). *Polynomials and polynomial inequalities*. Springer Science & Business Media.

Byrd, R. H., Gilbert, J. C., & Nocedal, J. (2000). A trust region method based on interior point techniques for nonlinear programming. *Mathematical Programming*, *89*(1), 149–185. https://doi.org/10.1007/PL00011391

Calafiore, G. C., & Possieri, C. (2018). A variation on a random coordinate minimization method for constrained polynomial optimization. *IEEE Control Systems Letters*, *2*(3), 531–536. https://doi.org/10.1109/LCSYS.2018.2843165

Chesi, G., Tesi, A., Vicino, A., & Genesio, R. (2001). An LMI approach to constrained optimization with homogeneous forms. *Systems & Control Letters*, *42*(1), 11–19. https://doi.org/10.1016/S0167-6911(00)00072-4

Collins, G. E., & Akritas, A. G (1976). Polynomial real root isolation using Descarte's rule of signs. In *Proceedings of the third ACM symposium on symbolic and algebraic computation* (pp. 272–275).

Conn, A. R., Gould, N. I. M., & Toint, P. (1991). A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds. *SIAM Journal on Numerical Analysis*, *28*(2), 545–572. https://doi.org/10.1137/0728030

Conn, A. R., Gould, N. I. M., & Toint, P. (1997). A globally convergent Lagrangian barrier algorithm for optimization with general inequality constraints and simple bounds. *Mathematics of Computation*, *66*(217), 261–288. https://doi.org/10.1090/S0025-5718-97-00777-1

Cox, D. A., Little, J., & O'Shea, D. (2015). *Ideals, varieties, and algorithms*. Springer.

De Klerk, E. (2008). The complexity of optimizing over a simplex, hypercube or sphere: A short survey. *Central European Journal of Operations Research*, *16*(2), 111–125. https://doi.org/10.1007/s10100-007-0052-9

Dorato, P. (2000). Quantified multivariate polynomial inequalities. The mathematics of practical control design problems. *IEEE Control Systems*, *20*(5), 48–58. https://doi.org/10.1109/37.872903

Engwerda, J. (2005). *LQ dynamic optimization and differential games*. John Wiley & Sons.

Gelfand, I. M., Kapranov, M., & Zelevinsky, A. (2008). *Discriminants, resultants, and multidimensional determinants*. Springer.

Goebel, R., Sanfelice, R. G., & Teel, A. R. (2012). *Hybrid dynamical systems*. Princeton University Press.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley Longman Publishing.

Grammatico, S., Subbaraman, A., & Teel, A. R. (2013). Discrete-time stochastic control systems: A continuous Lyapunov function implies robustness to strictly causal perturbations. *Automatica*, *49*(10), 2939–2952. https://doi.org/10.1016/j.automatica.2013.06.021

Grippo, L., & Sciandrone, M. (2000). On the convergence of the block nonlinear Gauss–Seidel method under convex constraints. *Operations Research Letters*, *26*(3), 127–136. https://doi.org/10.1016/S0167-6377(99)00074-7

Grüne, L., & Pannek, J (2011). Nonlinear model predictive control. In *Nonlinear model predictive control: Theory and algorithms* (pp. 43–66). Springer.

Hanzon, B., & Jibetean, D. (2003). Global minimization of a multivariate polynomial using matrix methods. *Journal of Global Optimization*, *27*(1), 1–23. https://doi.org/10.1023/A:1024664432540

Henrion, D., & Garulli, A. (Eds.). (2005). *Positive polynomials in control*. Springer.

Henrion, D., & Korda, M. (2014). Convex computation of the region of attraction of polynomial control systems. *IEEE Transactions on Automatic Control*, *59*(2), 297–312. https://doi.org/10.1109/TAC.2013.2283095

Henrion, D., & Lasserre, J. B.. (2003). GloptiPoly: Global optimization over polynomials with Matlab and SeDuMi. *ACM Transactions on Mathematical Software (TOMS)*, *29*(2), 165–194. https://doi.org/10.1145/779359.779363

Henrion, D., & Lasserre, J. B.. (2004). Solving nonconvex optimization problems. *IEEE Control Systems*, *24*(3), 72–83. https://doi.org/10.1109/MCS.2004.1299534

Jacobi, T., & Prestel, A. (2001). Distinguished representations of strictly positive polynomials. *Journal für die Reine und Angewandte Mathematik*, *532*, 223–83. https://doi.org/10.1515/crll.2001.023

Karush, W (1939). *Minima of functions of several variables with inequalities as side conditions* [Master's thesis]. University of Chicago.

Kearfott, R. B., & Walster, G. W. (2000). On stopping criteria in verified nonlinear systems or optimization algorithms. *ACM Transactions on Mathematical Software*, *26*(3), 373–389. https://doi.org/10.1145/358407.358418

Kuhn, H. W., & Tucker, A. W (1951). Nonlinear programming. In *Proceedings of the Berkeley symposium on mathematical statistics and probability*. Berkeley, CA: University California Press.

Lasserre, J. B. (2001). Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, *11*(3), 796–817. https://doi.org/10.1137/S1052623400366802

Lasserre, J. B. (2015). *An introduction to polynomial and semi-algebraic optimization* (Vol. 52). Cambridge University Press.

Li, Z., He, S., & Zhang, S. (2012). *Approximation methods for polynomial optimization: Models, algorithms, and applications*. Springer.

Li, Q., Yuan, J., Zhang, B., & Gao, C. (2017). Model predictive control for autonomous rendezvous and docking with a tumbling target. *Aerospace Science and Technology*, *69*, 700–711. https://doi.org/10.1016/j.ast.2017.07.022

Löfberg, J. (2009). Pre-and post-processing sum-of-squares programs in practice. *IEEE Transactions on Automatic Control*, *54*(5), 1007–1011. https://doi.org/10.1109/TAC.2009.2017144

Mayne, D. Q., Rawlings, J. B., Rao, C. V., & Scokaert, P. O. (2000). Constrained model predictive control: Stability and optimality. *Automatica*, *36*(6), 789–814. https://doi.org/10.1016/S0005-1098(99)00214-9

Menini, L., Possieri, C., & Tornambe, A. (2018a). Algebraic methods for multi-objective optimal design of control feedbacks for linear systems. *IEEE Transactions on Automatic Control*, *63*(12), 4188–4203. https://doi.org/10.1109/TAC.2018.2800784

Menini, L., Possieri, C., & Tornambe, A. (2018b). Dead-beat regulation of mechanical juggling systems. *Asian Journal of Control*, *20*(1), 1–11. https://doi.org/10.1002/asjc.v20.1

MOSEK ApS (2017). *The MOSEK optimization toolbox for MATLAB manual* (Version 8.1) [Computer software manual]. http://docs.mosek.com/8.1/toolbox/index.html.

Necoara, I., & Clipici, D. (2013). Efficient parallel coordinate descent algorithm for convex optimization problems with separable constraints: Application to distributed MPC. *Journal of Process Control*, *23*(3), 243–253. https://doi.org/10.1016/j.jprocont.2012.12.012

Nesterov, Y. (1998). Semidefinite relaxation and nonconvex quadratic optimization. *Optimization Methods and Software*, *9*(1-3), 141–160. https://doi.org/10.1080/10556789808805690

Nesterov, Y. (2012). Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, *22*(2), 341–362. https://doi.org/10.1137/100802001

Nesterov, Y. (2014). Subgradient methods for huge-scale optimization problems. *Mathematical Programming*, *146*(1–2), 275–297. https://doi.org/10.1007/s10107-013-0686-4

Papachristodoulou, A., Anderson, J., Valmorbida, G., Prajna, S., Seiler, P., & Parrilo, P (2013). *SOSTOOLS version 3.00 sum of squares optimization toolbox for MATLAB*. arXiv:1310.4716.

Parrilo, P. A (2000). *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization* [Unpublished doctoral dissertation]. California Institute of Technology.

Parrilo, P. A., & Sturmfels, B (2003). Minimizing polynomial functions. In S. Basu & L. González-Vega (Eds.), *Algorithmic and quantitative real algebraic geometry: DIMACS series in discrete mathematics and theoretical computer science* (pp. 83–99). American Mathematical Society.

Possieri, C., & Sassano, M. (2016). On polynomial feedback Nash equilibria for two-player scalar differential games. *Automatica*, *74*, 23–29. https://doi.org/10.1016/j.automatica.2016.08.006

Possieri, C., & Teel, A. R. (2017). A Lyapunov theorem certifying global weak reachability for stochastic difference inclusions with random inputs. *Systems & Control Letters*, *109*, 37–42. https://doi.org/10.1016/j.sysconle.2017.09.007

Powell, M. J. (1973). On search directions for minimization algorithms. *Mathematical Programming*, *4*(1), 193–201. https://doi.org/10.1007/BF01584660

Rastrigin, L. A. (1963). The convergence of the random search method in the extremal control of a many parameter system. *Automation and Remote Control*, *24*, 1337–1342. https://doi.org/10.1007/BF00935752

Rheinboldt, W. C., Mesztenyi, C. K., & Fitzgerald, J. M. (1977). On the evaluation of multivariate polynomials and their derivatives. *BIT*, *17*(4), 437–450. https://doi.org/10.1007/BF01933453

Richtárik, P., & Takáč, M. (2014). Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming*, *144*(1–2), 1–38. https://doi.org/10.1007/s10107-012-0614-z

Rockafellar, R. T., & Wets, R. J. B. (2009). *Variational analysis*. Springer.

Rouillier, F. (1999). Solving zero-dimensional systems through the rational univariate representation. *Applicable Algebra in Engineering, Communication and Computing*, *9*(5), 433–461. https://doi.org/10.1007/s002000050114

Rudin, W. (1964). *Principles of mathematical analysis*. McGraw-Hill.

Schumer, M., & Steiglitz, K. (1968). Adaptive step size random search. *IEEE Transactions on Automatic Control*, *13*(3), 270–276. https://doi.org/10.1109/TAC.1968.1098903

Shor, N. Z. (1987). Class of global minimum bounds of polynomial functions. *Cybernetics and Systems Analysis*, *23*(6), 731–734. https://doi.org/10.1007/BF01070233

Solis, F. J., & Wets, R. J. B. (1981). Minimization by random search techniques. *Mathematics of Operations Research*, *6*(1), 19–30. https://doi.org/10.1287/moor.6.1.19

Spall, J. C. (2005). *Introduction to stochastic search and optimization: Estimation, simulation, and control* (Vol. 65). John Wiley & Sons.

Subbaraman, A., & Teel, A. R. (2013). A converse Lyapunov theorem for strong global recurrence. *Automatica*, *49*(10), 2963–2974. https://doi.org/10.1016/j.automatica.2013.07.001

Szegohuml;, G. (1939). *Orthogonal polynomials*. American Mathematical Society.

Teel, A. R. (2013). A Matrosov theorem for adversarial Markov decision processes. *IEEE Transactions on Automatic Control*, *58*(8), 2142–2148. https://doi.org/10.1109/TAC.2013.2250073

Teel, A. R., Hespanha, J. P., & Subbaraman, A. (2014). A converse Lyapunov theorem and robustness for asymptotic stability in probability. *IEEE Transactions on Automatic Control*, *59*(9), 2426–2441. https://doi.org/10.1109/TAC.2014.2322431

Tempo, R., Calafiore, G., & Dabbene, F. (2012). *Randomized algorithms for analysis and control of uncertain systems*. Springer.

Thygesen, U. H (1997). *A survey of Lyapunov techniques for stochastic differential equations* (Tech. Rep. No. IMM 18-1997). Lyngby: Department of Mathematical Modeling, Technical University of Denmark.

Tseng, P. (2001). Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, *109*(3), 475–494. https://doi.org/10.1023/A:1017501703105

Tütüncü, R. H., Toh, K. C.., & Todd, M. J. (2003). Solving semidefinite-quadratic-linear programs using SDPT3. *Mathematical Programming*, *95*(2), 189–217. https://doi.org/10.1007/s10107-002-0347-5

Verschelde, J (1999). *Polynomial homotopies for dense, sparse and determinantal systems*. arXiv:math/9907060.

Wächter, A., & Biegler, L. T. (2006). On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. *Mathematical Programming*, *106*(1), 25–57. https://doi.org/10.1007/s10107-004-0559-y

Weiss, A., Baldwin, M., Erwin, R. S., & Kolmanovsky, I. (2015). Model predictive control for spacecraft rendezvous and docking: Strategies for handling constraints and case studies. *IEEE Transactions on Control Systems Technology*, *23*(4), 1638–1647. https://doi.org/10.1109/TCST.2014.2379639

Wolkowicz, H., Saigal, R., & Vandenberghe, L. (2012). *Handbook of semidefinite programming: Theory, algorithms, and applications* (Vol. 27). Springer.

Wright, S. J. (2015). Coordinate descent algorithms. *Mathematical Programming*, *151*(1), 3–34. https://doi.org/10.1007/s10107-015-0892-3

Zabinsky, Z. B (2008). Global optimization: Hit and run methods. In C. Floudas & P. Pardalos (Eds.), *Encyclopedia of optimization*. Boston, MA: Springer.

Zadeh, N. (1970). A note on the cyclic coordinate ascent method. *Management Science*, *16*(9), 642–644. https://doi.org/10.1287/mnsc.16.9.642

Appendix. Proofs of main results

A.1 Proof of Lemma 3.1

The first two parts of the proof are just sketched, since they follow the same lines of Calafiore and Possieri (2018), with minor adaptations due to the fact that the set Ω is not assumed to be convex here. We begin by showing that sub-level sets of f are compact. Toward this end, let $f^\star = f(\mathbf{x}^\star)$ where $\mathbf{x}^\star$ is any point in $\mathcal{A}$. First, notice that the sub-level set $\mathcal{L}_c$ of f,

$$\mathcal{L}_c \doteq \{\mathbf{x} \in \mathbb{R}^n : f(\mathbf{x}) - f^\star \leqslant c\} \cap \Omega, \tag{A1}$$

is compact for all $c \in \mathbb{R}_{\geqslant 0}$, see Calafiore and Possieri (2018). Second, we show that $f(\mathbf{g}) \leqslant f(\mathbf{x})$ for all $\mathbf{g} \in G(\mathbf{x}, \mathbf{s})$ and all $(\mathbf{x}, \mathbf{s}) \in \Omega \times \mathbb{S}^n$. Indeed, assume by contradiction that there exists $(\mathbf{x}, \mathbf{s}) \in \Omega \times \mathbb{S}^n$ such that $f(\mathbf{g}) > f(\mathbf{x})$ for some $\mathbf{g} \in G(\mathbf{x}, \mathbf{s})$. This implies that, for such a pair $(\mathbf{x}, \mathbf{s}) \in \Omega \times \mathbb{S}^n$, it holds that $f(\mathbf{x} + \mathbf{s} \arg\min_{\lambda \in \mathcal{I}} f(\mathbf{x} + \lambda\, \mathbf{s})) > f(\mathbf{x})$ leading to a contradiction of the definition of the $\arg\min(\cdot)$ function and of the fact that $0 \in \mathcal{I}$ since $\mathbf{x}$ is in Ω. Therefore, due to the boundedness of the sub-level sets $\mathcal{L}_c$, the map $G : \Omega \times \mathbb{S}^n \to \Omega$ is locally bounded. Moreover, since the set Ω is either compact or $f \in \mathrm{ru}(\Omega)$, by the same reasoning given above about the compactness of the sets $\mathcal{L}_c$, the mapping $\mathbf{s} \mapsto \mathrm{graph}(G(\cdot, \mathbf{s})) \doteq \{(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^n \times \mathbb{R}^n : y \in G(\mathbf{x}, \mathbf{s})\}$ has closed values. Hence, measurability of such a mapping follows by Example 5.22 and Exercise 14.9 of Rockafellar

and Wets (2009). Therefore, the map $G : \mathbb{R}^n \times \mathbb{S}^n \rightrightarrows \mathbb{R}^n$ satisfies Standing Assumption 1 of Subbaraman and Teel (2013), Teel et al. (2014), Possieri and Teel (2017), which guarantees the existence of maximal solutions in $\mathcal{R}(\mathbf{x}^0)$, provided that $\mathbf{x}^0 \in \Omega$.

We conclude the proof by showing that there does not exist $(\mathbf{x}, \mathbf{s}) \in \Omega \times \mathbb{S}^n$ such that $G(\mathbf{x}, \mathbf{s}) = \emptyset$. Given $\mathbf{x} \in \Omega$ and $\mathbf{s} \in \mathbb{S}^n$, define $\tilde{f} \in \mathbb{R}[\lambda]$ and $\tilde{\mathbf{h}} \in \mathbb{R}^m[\lambda]$ as $\tilde{f}(\lambda) \doteq f(\mathbf{x} + \lambda\, \mathbf{s})$ and $\tilde{\mathbf{h}}(\lambda) = [\tilde{h}_1(\lambda) \ \dots \ \tilde{h}_m(\lambda)]^\top \doteq \mathbf{h}(\mathbf{x} + \lambda\, \mathbf{s})$. Thus, $G(\mathbf{x}, \mathbf{s}) = \mathbf{x} + \lambda^\star\, \mathbf{s}$, where $\lambda^\star$ is the set of all the solutions to

$$\left| \begin{array}{ll} \min & \tilde{f}(\lambda), \\ \text{s.t.} & \lambda \in \tilde{\Omega}, \end{array} \right. \tag{A2}$$

where $\tilde{\Omega} \doteq \{\lambda \in \mathbb{R} : \tilde{h}_1(\lambda) \leqslant 0, \dots, \tilde{h}_m(\lambda) \leqslant 0\}$. Hence, if the PMP (A2) admits a solution, then $G(\mathbf{x}, \mathbf{s}) \neq \emptyset$. If Ω is compact, then $\tilde{\Omega}$ is compact, and hence the PMP (A2) admits a solution by the extreme value theorem, see Rudin (1964). Thus, assume that Ω is not compact, but $f \in \mathrm{ru}(\Omega)$. Since $\mathbf{x} \in \Omega$ implies that $0 \in \tilde{\Omega}$ and $\min_{\lambda \in \tilde{\Omega}} \tilde{f}(\lambda) \leqslant \tilde{f}(0)$, solving the PMP (A2) corresponds to solving the univariate PMP

$$\left| \begin{array}{ll} \min & \tilde{f}(\lambda), \\ \text{s.t.} & \lambda \in (\tilde{\Omega} \cap [\underline{\lambda}, \overline{\lambda}]), \end{array} \right. \tag{A3}$$

where $\underline{\lambda}$ and $\overline{\lambda}$ are the smallest and the largest real roots of $\tilde{f}(\lambda) - \tilde{f}(0)$, respectively. Thus, since $\tilde{\Omega} \cap [\underline{\lambda}, \overline{\lambda}]$ is compact, the PMP (A3) admits a solution. Therefore, the statement of the lemma follows from Proposition 1 of Teel et al. (2014).

A.2 Proof of Theorem 3.1

The proof is similar to the proof of Theorem 1 of Calafiore and Possieri (2018) with suitable adaptations due to the different distribution of the sequence of random variables $\{\mathbf{s}^k\}_{k \in \mathbb{N}}$, and therefore it is just sketched. First, note that, under the assumptions of Lemma 3.1, the set $\mathcal{A} = \mathcal{L}_0$ is compact. Hence, let $V : \mathbb{R}^n \to \mathbb{R}_{\geqslant 0}$ be any smooth function such that $V(\mathbf{x}) = f(\mathbf{x}) - f^\star$ for all $\mathbf{x} \in \Omega$ and $\lim_{\|\mathbf{x}\| \to +\infty} V(\mathbf{x}) = +\infty$. Such a function exists since either Ω is compact or $f \in \mathrm{ru}(\Omega)$. Since the set $\mathcal{A}$ is compact, $V(\mathbf{x}) = 0 \iff \mathbf{x} \in \mathcal{A}$, and V is radially unbounded, by Goebel et al. (2012, p. 54), there exist class $\mathcal{K}_\infty$ functions $\underline{\alpha}$ and $\overline{\alpha}$ such that

$$\underline{\alpha}(\|\mathbf{x}\|_{\mathcal{A}}) \leqslant V(\mathbf{x}) \leqslant \overline{\alpha}(\|\mathbf{x}\|_{\mathcal{A}}).$$

By the reasoning given in the proof of Lemma 3.1 about the fact that $f(\mathbf{g}) \leqslant f(\mathbf{x})$ for all $\mathbf{g} \in G(\mathbf{x}, \mathbf{s})$ and all $(\mathbf{x}, \mathbf{s}) \in \Omega \times \mathbb{S}^n$, it results that $\sup_{\mathbf{g} \in G(\mathbf{x}, \mathbf{s})} V(\mathbf{g}) \leqslant V(\mathbf{x})$ for all $(\mathbf{x}, \mathbf{s}) \in \Omega \times \mathbb{S}^n$. Thus, given $\mathbf{x} \in \Omega \setminus \mathcal{A}$, let $\Lambda = (f(\mathbf{x}) - f^\star)/2 \in \mathbb{R}_{>0}$ and let $\aleph = \min\{\Lambda, \upsilon^\star\}$. Since $(\mathcal{A} + \nu \mathbb{B}^n) \cap \Omega$ has nonzero measure for all $\nu \in (0, \upsilon^\star)$, $\mathcal{L}_\aleph$ has nonzero measure. Therefore, for each $\mathbf{x} \in \Omega \setminus \mathcal{A}$, there is a selection $\mathcal{S}_\mathbf{x}$ of $\mathbb{S}^n$, whose measure is not zero, such that $\sup_{\mathbf{g} \in G(\mathbf{x}, \mathbf{s})} V(\mathbf{g}) \leqslant V(\mathbf{x})/2$ for all $\mathbf{s} \in \mathcal{S}_\mathbf{x}$. Therefore, since $\mathbf{s}^k \sim (1-p)\mathrm{Uni}(\mathbb{S}^n) + p \sum_{i=1}^n \varpi(i) \delta(\mathbf{e}_i)$ and hence $\mu(\mathbf{e}_i) = p\varpi(i)$, $i = 1, \dots, n$, and $\mu(\mathcal{S}) = (1-p) \int_\mathcal{S} 1\, \mathrm{d}\mathbf{s}$ for each $\mathcal{S} \subset \mathbb{S}^n$ such that $\mathbf{e}_i \notin \mathcal{S}$, $i = 1, \dots, n$, we have that

$$\begin{aligned} &\int_{\mathbb{S}^n} \sup_{\mathbf{g} \in G(\mathbf{x}, \mathbf{s})} V(\mathbf{g}) \mu(\mathrm{d}\mathbf{s}) \\ &= (1-p)\Gamma \int_{\mathbb{S}^n} \sup_{\mathbf{g} \in G(\mathbf{x}, \mathbf{s})} V(\mathbf{g})\, \mathrm{d}\mathbf{s} + p \sum_{i=1}^n \varpi(i) \sup_{\mathbf{g} \in G(\mathbf{x}, \mathbf{e}_i)} V(\mathbf{g}) \\ &\leqslant (1-p)\Gamma \left(\frac{V(\mathbf{x})}{2} \int_{\mathcal{S}_\mathbf{x}} 1\, \mathrm{d}\mathbf{s} + V(\mathbf{x}) \int_{\mathbb{S}^n \setminus \mathcal{S}_\mathbf{x}} 1\, \mathrm{d}\mathbf{s} \right) + pV(\mathbf{x}) \\ &\leqslant V(\mathbf{x}) - \frac{1-p}{2} \Gamma V(\mathbf{x}) \int_{\mathcal{S}_\mathbf{x}} 1\, \mathrm{d}\mathbf{s}, \end{aligned} \tag{A4}$$

where $\Gamma = \frac{1}{2\pi^{n/2}} \int_0^\infty x^{\frac{n}{2}-1} e^{-x}\, \mathrm{d}x$, for all $\mathbf{x} \in \Omega$. Hence, letting $\varrho(\mathbf{x})$ be any function in $\mathcal{PD}(\mathcal{A})$ such that

$$\varrho(\mathbf{x}) \leqslant \frac{1-p}{2}\, \Gamma\, V(\mathbf{x}) \int_{\mathcal{S}_\mathbf{x}} 1\, \mathrm{d}\mathbf{s}, \tag{A5}$$

for all $\mathbf{x} \in \Omega$, whose existence is guaranteed by the fact that $\mathcal{S}_\mathbf{x}$ has nonzero measure, we have that, for all $\mathbf{x} \in \Omega$,

$$\int_{\mathbb{S}^n} \sup_{\mathbf{g} \in G(\mathbf{x}, \mathbf{s})} V(\mathbf{g})\, \mu(\mathrm{d}\mathbf{s}) \leqslant V(\mathbf{x}) - \varrho(\mathbf{x}).$$

Therefore, since, by Lemma 3.1, if $\mathbf{x}^0 \in \Omega$ then $\mathbf{x}^k \in \Omega$ for all $k \in \mathbb{N}$, by Theorem 1 of Teel (2013), the set $\mathcal{A}$ if asymptotically stable in probability from Ω.

Remark A.1: Since $V(\mathbf{x}) = f(\mathbf{x}) - f^\star$ for all $\mathbf{x} \in \Omega$, the function $\varrho(\mathbf{x})$ in (A5) provides information about the expected per-iteration decrease of the proposed algorithm. Namely, by construction, we have that $\mathbb{E}(f(\mathbf{x}^+)) \leqslant f(\mathbf{x}) - \varrho(\mathbf{x})$ for all $\mathbf{x} \in \Omega$.

A.3 Proof of Theorem 3.2

If the assumptions of Theorem 3.1 are met, then the stochastic difference inclusion (10) satisfies Standing Assumption 1 of Possieri and Teel (2017) and the set $\mathcal{L}_\eta$ defined in (A1) is compact. Thus the set $\mathcal{O}_1$ is open, and, by Lemma 1 of Possieri and Teel (2017), $\ell_{\subset \mathcal{O}_1}(k, \mathbf{x}^0) = \inf_{\{\mathbf{z}^k\}_{k \in \mathbb{N}} \in \mathcal{R}(\mathbf{x}^0)} \mathbb{E}[\prod_{i=1}^k \mathbb{I}_{\mathcal{O}_1}(\mathbf{z}^k)]$. Thus, since Ω is positively invariant with respect to the stochastic difference inclusion (10), the function $1 - \ell_{\subset \mathcal{O}_1}(K, \mathbf{x}^0)$ constitutes an upper bound over all the solutions from $\mathbf{x}^0$ for the probability of reaching the set $\mathcal{L}_\eta$. Thus, since, by Lemma 3.1, $\mathcal{L}_\eta$ is positively invariant with respect to inclusion (9), we have that

$$\begin{aligned} \mathbb{P}(f(\mathbf{x}^i) \leqslant \eta, \forall i \geqslant K) &= \mathbb{P}(\mathbf{x}^K \in \mathcal{L}_\eta) \\ &= \mathbb{P}(\exists i \leqslant K \text{s.t.} f(\mathbf{x}^i) \leqslant \eta) \leqslant 1 - \ell_{\subset \mathcal{O}_1}(K, \mathbf{x}^0). \end{aligned}$$

On the other hand, we have that $f(\mathbf{x}) \leqslant \eta$ for all $\mathbf{x} \in \mathcal{O}_2$. By Possieri and Teel (2017, Lem. 2), since the stochastic inclusion (10) satisfies Standing Assumption 1 of Possieri and Teel (2017), we have $\ell_{\cap \mathcal{O}_2}(k, \mathbf{x}^0) = \inf_{\{\mathbf{z}^k\}_{k \in \mathbb{N}} \in \mathcal{R}(\mathbf{x}^0)} \mathbb{E}[\max_{i \in \{1 \dots, k\}} \mathbb{I}_{\mathcal{O}_2}(\mathbf{z}^k)]$, i.e. $\ell_{\cap \mathcal{O}_2}(k, \mathbf{x}^0)$ constitute a lower bound over all the random solutions from $\mathbf{x}^0 \in \Omega$ for the probability of reaching the set $\mathcal{O}_2$ in k iterations. Thus, since $f(\mathbf{x}^{k+1}) \leqslant f(\mathbf{x}^k)$ and hence $\mathbb{P}(f(\mathbf{x}^i) \leqslant \eta,\ \forall i \geqslant K | \exists j \in \{1, \dots, K\}$ such that $\mathbf{x}^j \in \mathcal{O}_2) = 1$, it results that $\mathbb{P}(f(\mathbf{x}^i) \leqslant \eta,\ \forall i \geqslant K) \geqslant \mathbb{P}(f(\mathbf{x}^i) \leqslant \eta,\ \forall i \geqslant K \wedge \exists j \in \{1, \dots, K\}$ such that $\mathbf{x}^j \in \mathcal{O}_2) = \mathbb{P}(\exists j \in \{1, \dots, K\}$ such that $\mathbf{x}^j \in \mathcal{O}_2) \geqslant \ell_{\cap \mathcal{O}_2}(K, \mathbf{x}^0)$, for all random solutions $\{\mathbf{x}^k\}_{k \in \mathbb{N}} \in \mathcal{R}(\mathbf{x}^0)$.

A.4 Proof of Theorem 4.1

By using a construction similar to the one made in the proof of Theorem 3.1 (with $\mathcal{A} + \nu \mathbb{B}^n$ substituted by $\breve{\Omega}_{e,\tau^\star}$), it can be easily derived that the set $\breve{\Omega}_{e,\tau}$ is asymptotically stable in probability from Ω_e. Hence, by definition, for each $\varepsilon \in \mathbb{R}_{>0}$, $\sigma \in \mathbb{R}_{>0}$ and $\Delta \in \mathbb{R}_{>0}$, there is an integer K such that

$$\begin{aligned} &\mathbf{x}_e^0 \in (\breve{\Omega}_{e,\tau^\star} + \Delta\, \mathbb{B}^n) \cap \Omega_e, \{\mathbf{x}_e^k\}_{k \in \mathbb{N}} \in \mathcal{R}_e(\mathbf{x}_e^0) \\ &\quad \implies \mathbb{P}(\mathbf{x}_e^k(\psi) \in \breve{\Omega}_{e,\tau^\star} + \varepsilon\, \mathbb{B}_o^n,\ k \geqslant K) \geqslant 1 - \sigma, \end{aligned}$$

where $\mathcal{R}_e(\mathbf{x}^0)$ denotes the set of all the maximal solutions to the stochastic difference inclusion $\mathbf{x}_e^{k+1} \in G_e(\mathbf{x}_e, \mathbf{s}_e)$, starting at $\mathbf{x}_e^0$, and $G_e(\mathbf{x}_e, \mathbf{s}_e)$ is the set-valued mapping obtained adapting the map $G(\mathbf{x}, \mathbf{s})$ given in (10) to the PMP (12). By considering that the entries of $\mathbf{h}(\mathbf{x})$ are continuous functions and $h_i(\mathbf{x}) \leqslant -\tau^\star$ for all $\mathbf{x}_e \in \breve{\Omega}_{e,\tau^\star}$, $i = 1, \dots, m$, there exists a sufficiently small $\varepsilon^\star \in \mathbb{R}_{>0}$ such that $\mathbf{x}_e^k \in \breve{\Omega}_{e,\tau^\star} + \varepsilon^\star\, \mathbb{B}_o^n \implies h_i(\mathbf{x}^k) \leqslant 0$, $i = 1, \dots, m$.

A.5 Proof of Proposition 4.1

By Rheinboldt et al. (1977), in order to determine the value of the monomials $\mathbf{x}_{-i}^{\boldsymbol{\alpha}_{-i}}$, one has to carry out up to $n^2\, d$ elementary operations. The proof is concluded by the fact that there are $\sum_{r=0}^d \binom{n+r-1}{r} = \frac{d+1}{n} \binom{n+d}{d+1}$ vectors

$\boldsymbol{\alpha} \in \mathbb{N}^n$ such that $|\boldsymbol{\alpha}| \leqslant d$ and that, at Step 9 of Algorithm 1, we need to compute $m+1$ coordinate-wise polynomials.

A.6 Proof of Proposition 4.2

Note that, in order to determine each of the vectors $\boldsymbol{\zeta}_i^{\alpha_i}(x_i^k, s_i^k)$ and $\boldsymbol{\omega}_{\boldsymbol{\alpha}}(x_i^k, s_i^k)$, one has to carry out up to $2d^3(d+1)$ and $\frac{5}{2}n\,(d+1)(d\,n+d+2)$ elementary operations, respectively. Therefore, in view of (14), the function $\tilde{f}$ and each entry of the polynomial vector $\tilde{\mathbf{h}}$ can be determined by carrying out up to $\frac{1}{2}(d+1)^2(4d^3+5d\,n+5d+10)\binom{d+n}{d+1}$ elementary operations, thus concluding the proof.