(Preprint) AAS 25-882

POLYNOMIAL-BASED SOLUTION TO THE TARGETING PROBLEM FOR ONBOARD APPLICATIONS

Alberto Fossà,* Roberto Armellin,† Didier Henrion,‡§ and Renato Zanetti¶

This paper solves the targeting problem focusing on accuracy and efficiency. Given the initial spacecraft state, the objective is to compute the optimal maneuver to reach the target state in a fixed time interval. The problem is firstly recast as a polynomial optimization problem (POP) using high-order Taylor expansions of the dynamics and the constraints. Moment-sum-of-squares (SOS) optimization is leveraged to efficiently solve this POP. A convex formulation based on a second-order expansion of the dynamics is also proposed. The POP is more suitable for large maneuvers and long time spans, while the convex formulation is preferred when efficiency is paramount.

INTRODUCTION

As spacecraft operations are evolving towards complete onboard autonomy, the need for efficient and robust trajectory planning is becoming increasingly important. The deployment of fully autonomous systems is however constrained by the onboard computational resources and by the need for a guaranteed solution to the problem at hand. This is evident in the context of trajectory optimization, which mostly relies on the numerical solution of large nonlinear programming (NLP) problems. Recent research thus focused on more efficient methods that can provide a guaranteed convergence to the global optimum. One of such techniques is convex optimization,[1] whose main drawback is that most optimization problems in astrodynamics are intrinsically nonconvex. This limitation can be overcome by solving a sequence of relaxed convex sub-problems around a nominal trajectory, giving birth to the so-called successive convex optimization. Another class of optimization problems that received significant attention in recent years is that of polynomial optimization problems (POPs).[2–4] Although not yet common in astrodynamics, this class of problems positions itself between NLPs and convex optimization, allowing for more flexibility in the formulation of the trajectory optimization problem while still providing several advantages over traditional NLP. Dedicated algorithms such as moment-sum-of-squares (SOS) optimization can in fact solve POPs efficiently, while guaranteeing under mild assumptions that the global optimum is found.

The focus of this work is thus on the solution of the targeting problem leveraging these advanced optimization techniques. The design of an impulsive maneuver that steers the spacecraft from a given

*Postdoctoral Fellow, Oden Institute for Computational Engineering and Sciences, The University of Texas at Austin, Austin, TX, 78712.
†Professor, Te Pūnaha Ātea – Space Institute, The University of Auckland, Auckland, 1010, New Zealand.
‡Senior Researcher, LAAS-CNRS, Université de Toulouse, Toulouse, France.
§Professor, Faculty of Electrical Engineering, Czech Technical University in Prague, Prague, Czech Republic.
¶Associate Professor, Department of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, Austin, TX, 78712.

initial state to a desired final state is in fact a common problem in astrodynamics. This problem is typically solved by analyzing the first-order sensitivity of the final state with respect to deviations in the initial conditions (ICs), which limits its applicability to small maneuvers and short time spans. In contrast, this papers reformulates the targeting problem as a POP in which the dynamics and the constraints are expanded at higher order to better account for the effects of the nonlinear orbital dynamics. These expansions are computed efficiently using differential algebra (DA) techniques,[5] thus avoiding the need for the numerical integration of the variational equations along with the state. For those cases in which a second-order expansion of the dynamics is sufficiently accurate, the POP is further reformulated as a convex optimization problem, leading to an even more efficient solution that still outperforms traditional first-order methods in terms of robustness and optimality.

The paper is organized as follows. Firstly, the targeting problem is formulated as a NLP problem using the true dynamics. Secondly, a POP formulation is presented, detailing how DA techniques are used to compute the arbitrary order Taylor expansions of the dynamics and the constraints. Then, a convex relaxation of the original problem is derived from a second-order expansion of the dynamics, and some considerations are made to improve the numerical conditioning of the problem. Finally, a numerical example is presented to demonstrate the performance of the proposed methods compared to the solution of the original NLP problem.

PROBLEM STATEMENT

Consider the targeting problem where the objective is to minimize the initial impulsive maneuver required to steer the spacecraft from a given initial state to a desired final state in a fixed time interval. This problem can be formulated as follows:

$$\min_{\Delta \boldsymbol{v}} \|\Delta \boldsymbol{v}\|_2 \tag{1}$$

subject to the equality constraints:

$$\boldsymbol{x}(t_f) = \boldsymbol{x}(t_0) + \int_{t_0}^{t_f} \boldsymbol{f}(\boldsymbol{x}(t), t)\,\mathrm{d}t\,, \tag{2a}$$

$$\boldsymbol{x}(t_0) = \boldsymbol{x}_0^- + \begin{bmatrix} \mathbf{0}_{n\times 1} \\ \Delta \boldsymbol{v} \end{bmatrix}, \tag{2b}$$

and inequality constraint:

$$[\boldsymbol{x}(t_f) - \bar{\boldsymbol{x}}_f]^T \boldsymbol{P}^{-1} [\boldsymbol{x}(t_f) - \bar{\boldsymbol{x}}_f] \leq d^2, \tag{3}$$

where $\boldsymbol{f}(\boldsymbol{x}(t), t)$ are the ballistic dynamics, $\boldsymbol{x}_0^- \in \mathbb{R}^{6\times 1}$ is the initial state before the maneuver, $\bar{\boldsymbol{x}}_f \in \mathbb{R}^{6\times 1}$ is the target state, t_0 and t_f are the initial and final times, $\Delta \boldsymbol{v} \in \mathbb{R}^{3\times 1}$ is the impulsive maneuver, $\boldsymbol{P} \in \mathbb{R}^{6\times 6}$ is the target covariance matrix, and $d \in \mathbb{R}$ is the target distance. Equation (3) is an upper bound on the squared Mahalanobis distance between the final state $\boldsymbol{x}(t)$ and a probability distribution with mean $\bar{\boldsymbol{x}}_f$ and covariance matrix $\boldsymbol{P}$.

Equations (1), (2) and (3) define a NLP problem that can be readily solved with dedicated algorithms such as IPOPT.[6] As the objective function is not differentiable in $\Delta \boldsymbol{v} = \mathbf{0}$, Eq. (1) is replaced by its square to avoid potential issues during the numerical solution of the NLP problem. The objective thus becomes:

$$\min_{\Delta \boldsymbol{v}} \|\Delta \boldsymbol{v}\|_2^2 \tag{4}$$

subject to the equality constraints in Eq. (2) and inequality constraint in Eq. (3).

POLYNOMIAL OPTIMIZATION PROBLEM

Solving the NLP problem defined by Eqs. (2), (3) and (4) might become computationally intensive, as the dynamics in Eq. (2a) must be numerically integrated at each solver iteration. The proposed solution is to compute a high-order Taylor expansion of $\boldsymbol{x}(t_f)$ function of $\Delta\boldsymbol{v}$, and evaluate this expression in place of Eq. (2a). As Eqs. (2b), (3) and (4) are already polynomial functions of $\Delta\boldsymbol{v}$, the original NLP problem becomes a POP. The latter can be then solved efficiently using the techniques described in the next section.

Polynomial expansion of the constraints

Consider the spacecraft state $\boldsymbol{x}$ defined as:

$$\boldsymbol{x} = \begin{bmatrix}\boldsymbol{r}^T & \boldsymbol{v}^T\end{bmatrix}^T = \begin{bmatrix}x & y & z & v_x & v_y & v_z\end{bmatrix}^T \in \mathbb{R}^{6\times 1}, \tag{5}$$

and the impulsive maneuver $\Delta\boldsymbol{v}$ given by:

$$\Delta\boldsymbol{v} = \begin{bmatrix}\Delta v_x & \Delta v_y & \Delta v_z\end{bmatrix}^T \in \mathbb{R}^{3\times 1}. \tag{6}$$

A polynomial expansion of the final state can be obtained efficiently using DA techniques.[5] The impulsive maneuver is firstly initialized as:

$$[\Delta\boldsymbol{v}] = \boldsymbol{0}_{3\times 1} + \delta\boldsymbol{v}, \tag{7}$$

where the square brackets denote Taylor polynomials, and $\delta\boldsymbol{v} = \begin{bmatrix}\delta v_x & \delta v_y & \delta v_z\end{bmatrix}^T$ are the three independent DA variables that represent an infinitesimal maneuver along the x, y and z directions, respectively. Then, evaluating Eq. (2) in the DA framework results in the following expression for the final state:

$$[\boldsymbol{x}(t_f)] = \mathcal{T}^{(k)}_{\boldsymbol{x}(t_f)}\left(\delta\boldsymbol{v}\right), \tag{8}$$

which is a vector of k-th order Taylor polynomials in $\delta\boldsymbol{v}$. Substituting Eq. (8) into the right-hand side (RHS) of Eq. (3) finally yields to the following approximation of the squared Mahalanobis distance:

$$\begin{aligned}\left[d_M^2(t_f)\right] &= \left\{[\boldsymbol{x}(t_f)] - \bar{\boldsymbol{x}}_f\right\}^T \boldsymbol{P}^{-1}\left\{[\boldsymbol{x}(t_f)] - \bar{\boldsymbol{x}}_f\right\} \\ &= \mathcal{T}^{(k)}_{d_M^2(t_f)}\left(\delta\boldsymbol{v}\right),\end{aligned} \tag{9}$$

which is again a k-th order Taylor polynomial in $\delta\boldsymbol{v}$. As the flow of the dynamics is embedded in Eq. (8), Eqs. (2) and (3) are replaced by the single inequality constraint:

$$\mathcal{T}^{(k)}_{d_M^2(t_f)}\left(\Delta\boldsymbol{v}\right) \leq d^2. \tag{10}$$

and the original NLP problem is recast as the POP of minimizing Eq. (4) subject to Eq. (10).

Quadratic expansion of the constraints

Consider the polynomial expansion of the final state given by Eq. (8). Choosing $k = 2$ and expanding terms yields

$$[x_i(t_f)] = \sum_{|\boldsymbol{\alpha}|\leq 2} c_{i,\boldsymbol{\alpha}}\delta\boldsymbol{v}^{\boldsymbol{\alpha}} \qquad i \in [1, 6], \tag{11}$$

where $[x_i(t_f)]$ are the six components of the DA vector $[\boldsymbol{x}(t_f)]$, $c_{i,\boldsymbol{\alpha}}$ are the coefficients of the monomials function of $\delta\boldsymbol{v}^{\boldsymbol{\alpha}}$, and $\boldsymbol{\alpha} := (\alpha_j) \in \mathbb{N}^3$ is a multi-index with α_j corresponding to the exponent of δv_j for $j = x, y, z$. Equation (11) can then be rewritten in matrix form as

$$[\boldsymbol{x}(t_f)] = \boldsymbol{c}_0 + \boldsymbol{A}\boldsymbol{z}, \tag{12}$$

where $\boldsymbol{c}_0 \in \mathbb{R}^{6\times 1}$ is the vector of constant coefficients, $\boldsymbol{A} \in \mathbb{R}^{6\times 9}$ is the matrix of first and second order coefficients, and $\boldsymbol{z}$ is defined as

$$\begin{aligned} \boldsymbol{z} &= [z_{\boldsymbol{\alpha}}]_{|\boldsymbol{\alpha}|\leq 2} = \begin{bmatrix} z_{100} & z_{010} & z_{001} & z_{200} & z_{110} & z_{020} & z_{101} & z_{011} & z_{002} \end{bmatrix}^T \\ &= [\delta\boldsymbol{v}^{\boldsymbol{\alpha}}]_{|\boldsymbol{\alpha}|\leq 2} = \begin{bmatrix} \delta v_x & \delta v_y & \delta v_z & \delta v_x^2 & \delta v_x\delta v_y & \delta v_y^2 & \delta v_x\delta v_z & \delta v_y\delta v_z & \delta v_z^2 \end{bmatrix}^T. \end{aligned} \tag{13}$$

Substituting Eq. (12) into Eq. (3) yields

$$[\boldsymbol{A}\boldsymbol{z} - (\bar{\boldsymbol{x}}_f - \boldsymbol{c}_0)]^T \boldsymbol{P}^{-1} [\boldsymbol{A}\boldsymbol{z} - (\bar{\boldsymbol{x}}_f - \boldsymbol{c}_0)] \leq d^2, \tag{14}$$

which can be rewritten as

$$\boldsymbol{z}^T\boldsymbol{Q}\boldsymbol{z} + \boldsymbol{c}^T\boldsymbol{z} + r \leq 0, \tag{15}$$

where

$$\boldsymbol{Q} = \boldsymbol{A}^T\boldsymbol{P}^{-1}\boldsymbol{A}, \tag{16a}$$

$$\boldsymbol{c} = -2\left(\bar{\boldsymbol{x}}_f - \boldsymbol{c}_0\right)\boldsymbol{P}^{-1}\boldsymbol{A}, \tag{16b}$$

$$r = \left(\bar{\boldsymbol{x}}_f - \boldsymbol{c}_0\right)^T \boldsymbol{P}^{-1}\left(\bar{\boldsymbol{x}}_f - \boldsymbol{c}_0\right) - d^2. \tag{16c}$$

Equation (15) is a fourth-order approximation of the constraint on the squared Mahalanobis distance function of the DA variables $\delta\boldsymbol{v}$. It does however not coincide with Eq. (9) for $k = 4$, as the dynamics in Eqs. (11) and (12) are truncated at second order. Moreover, additional constraints must be enforced to ensure the consistency of the polynomial expansion according to Eq. (13):

$$\begin{aligned} z_{100}^2 - z_{200} &= 0 \qquad & z_{100}z_{010} - z_{110} &= 0 \\ z_{010}^2 - z_{020} &= 0 \qquad & z_{100}z_{001} - z_{101} &= 0 \\ z_{001}^2 - z_{002} &= 0 \qquad & z_{010}z_{001} - z_{011} &= 0. \end{aligned} \tag{17}$$

Equations (16c) and (17) provides the basis for the convex relaxation of the targeting problem presented later in this paper.

MOMENT-SUM-OF-SQUARES OPTIMIZATION

Although POPs can be solved using generic NLP solvers, these algorithms treat the objective and constraints functions as black-boxes, and do not exploit their polynomial structure. Moreover, most solvers are gradient-based, meaning that the provided solution might be only locally optimal. In contrast, moment-SOS optimization[2,4] exploits the polynomial structure of the problem and builds a hierarchy of convex problems that can be solved efficiently via semidefinite programming (SDP)[4] techniques. Their solutions provide increasingly tighter lower bounds of the cost function until the optimal value is attained. The optimal solution to the original POP is then recovered from the underlying relaxation.

Moment-sum-of-squares hierarchy

The following paragraphs recall the basic ingredients of the moment-SOS hierarchy. Readers interested in further details can consult the lecture notes[7] or the more technical and comprehensive monograph.[4]

Consider the POP

$$p^* := \min_{\boldsymbol{\chi}\in\mathscr{X}} p(\boldsymbol{\chi}) \tag{18}$$

where p is a given multivariate polynomial of the indeterminates $\boldsymbol{\chi} \in \mathbb{R}^n$, and

$$\mathscr{X} := \{\boldsymbol{\chi} \in \mathbb{R}^n : g_k(\boldsymbol{\chi}) \geq 0,\ k = 1, \dots, m\} \tag{19}$$

is a bounded set defined by a given vector $\boldsymbol{g} := (g_k)_{k=1,\dots,m}$ of polynomials. Since $\mathscr{X}$ is compact and p is continuous, a minimizer exists. Yet, the POP in Eq. (18) is a difficult global optimization problem with potentially many local and global optima.

The first step consists in reformulating the POP as a finite-dimensional linear programming (LP) problem. Let $\mathbb{R}[\boldsymbol{\chi}]_d$ denote the vector space of polynomials of $\boldsymbol{\chi}$ of degree at most d. It can be indexed by $\mathbb{N}^n_d := \{\boldsymbol{\alpha} \in \mathbb{N}^n : \sum_{i=1}^n \alpha_i \leq d\}$. Let $\boldsymbol{b}(\boldsymbol{\chi}) := (b_{\boldsymbol{\alpha}}(\boldsymbol{\chi}))_{\boldsymbol{\alpha}\in\mathbb{N}^n_d}$ denote a basis for this space, with the convention that $b_{\boldsymbol{0}}(\boldsymbol{\chi}) := 1$, so that every element $p \in \mathbb{R}[\boldsymbol{\chi}]_d$ can be expressed as a linear combination

$$p(\boldsymbol{\chi}) = \sum_{\boldsymbol{\alpha}\in\mathbb{N}^n_d} p_{\boldsymbol{\alpha}} b_{\boldsymbol{\alpha}}(\boldsymbol{\chi}). \tag{20}$$

Now consider a linear functional on $\mathbb{R}[\boldsymbol{\chi}]_d$. In basis $\boldsymbol{b}$, such a functional can be represented with a vector $\boldsymbol{y} := (y_{\boldsymbol{\alpha}})_{\boldsymbol{\alpha}\in\mathbb{N}^n_d}$ as follows:

$$\ell_{\boldsymbol{y}}(p) := \sum_{\boldsymbol{\alpha}\in\mathbb{N}^n_d} p_{\boldsymbol{\alpha}} y_{\boldsymbol{\alpha}}. \tag{21}$$

Informally, $\ell_{\boldsymbol{y}}$ linearizes polynomials, compare Eq. (20) and Eq. (21). Note that $\ell_{\boldsymbol{y}}(1) = 1$. Given any vector $\boldsymbol{\chi} \in \mathscr{X}$, note that for the specific choice $\boldsymbol{y} = \boldsymbol{b}(\boldsymbol{\chi})$, the linear functional $\ell_{\boldsymbol{y}}(p) = p(\boldsymbol{\chi})$ coincides with the point evaluation at $\boldsymbol{\chi}$, and the expression $p(\boldsymbol{\chi})$ (which is nonlinear in $\boldsymbol{\chi}$) can be replaced with the expression $\ell_{\boldsymbol{y}}(p)$ (which is linear in $\boldsymbol{y}$). The constraint

$$\boldsymbol{y} \in \boldsymbol{b}(\mathscr{X}) := \{\boldsymbol{b}(\boldsymbol{\chi}) : \boldsymbol{\chi} \in \mathscr{X}\} \tag{22}$$

is however nonconvex. Since $\ell_{\boldsymbol{y}}(p)$ is linear in $\boldsymbol{y}$, Eq. (22) can be replaced with the convex constraint

$$\boldsymbol{y} \in \mathscr{M}(\mathscr{X}) := \operatorname{conv} \boldsymbol{b}(\mathscr{X}) \tag{23}$$

where conv denotes the convex hull*. Now consider the finite-dimensional LP problem

$$p^*_M := \min_{\boldsymbol{y}\in\mathscr{M}(\mathscr{X})} \ell_{\boldsymbol{y}}(p). \tag{24}$$

It turns out that solving the LP in Eq. (24) is equivalent to solving the original nonconvex POP in Eq. (18):

Lemma 1 $p^* = p^*_M$.

*The convex hull of a set $\mathscr{A}$ is the smallest closed convex set containing $\mathscr{A}$.

Proof: On one hand, for any global minimizer $\boldsymbol{\chi}^*$ of Eq. (18), notice that $\boldsymbol{y}^* := \boldsymbol{b}(\boldsymbol{\chi}^*) \in \mathscr{M}(\mathscr{X})$ is admissible for Eq. (24), with value $\ell_{\boldsymbol{y}^*}(p) = p(\boldsymbol{\chi}^*) = p^* \geq p^*_M$. On the other hand, it holds that $p(\boldsymbol{\chi}) \geq p^*$ for all admissible $\boldsymbol{\chi} \in \mathscr{X}$, and hence $\ell_{\boldsymbol{y}}(p) \geq \ell_{\boldsymbol{y}}(p^*) = p^*$ for any admissible $\boldsymbol{y} \in \mathscr{M}(\mathscr{X})$. In particular this is true for a minimizer, implying $p^*_M \geq p^*$. □

The difficulty in solving Eq. (24) is however concentrated into the constraint set $\mathscr{M}(\mathscr{X})$. Despite being finite-dimensional and convex, this set is difficult to manipulate. Just determining whether it contains a given vector can be challenging. The main idea behind the moment-SOS hierarchy consists of approximating $\mathscr{M}(\mathscr{X})$ with a family of increasingly tight semidefinite representable* convex sets. Optimization of linear functions on semidefinite representable sets is the subject of SDP, and it can be done efficiently with interior-point algorithms.[8]

A hierarchy of semidefinite representable outer approximations of $\mathscr{M}(\mathscr{X})$ can be constructed as follows. Let $g_0 := 1$, let $r_{\boldsymbol{g}}$ be the smallest integer larger than $\max_k \frac{\deg g_k}{2}$, let $r \geq r_{\boldsymbol{g}}$ be any integer and, for each k, let r_k be the smallest integer larger than $r - \frac{\deg g_k}{2}$. Then define

$$\begin{aligned}\mathscr{M}(\mathscr{X})_r &:= \left\{(y_{\boldsymbol{\alpha}})_{\boldsymbol{\alpha}\in\mathbb{N}^n_d} : \ell_{\boldsymbol{y}}(1) = 1,\ \ell_{\boldsymbol{y}}(q^2 g_k) \geq 0,\ \forall q \in \mathbb{R}[\boldsymbol{\chi}]_{r_k},\ k = 0, 1, \dots, m\right\} \\ &= \left\{(y_{\boldsymbol{\alpha}})_{\boldsymbol{\alpha}\in\mathbb{N}^n_d} : \ell_{\boldsymbol{y}}(1) = 1,\ \mathbf{M}_{r_k}(g_k \boldsymbol{y}) \succeq 0,\ k = 0, 1, \dots, m\right\}\end{aligned} \tag{25}$$

where the matrix $\mathbf{M}_{r_k}(g_k\boldsymbol{y})$ represents the quadratic form $q \mapsto \ell_{\boldsymbol{y}}(g_k q^2)$ in basis $\boldsymbol{b}$, i.e.

$$\mathbf{M}_{r_k}(g_k\boldsymbol{y}) := \ell_{\boldsymbol{y}}(g_k b_{\boldsymbol{\alpha}_1} b_{\boldsymbol{\alpha}_2})_{\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2\in\mathbb{N}^n_{r_k}}. \tag{26}$$

The notation reflects the fact that the matrix depends linearly on g_k (for given $\boldsymbol{y}$) and also linearly on $\boldsymbol{y}$ (for given g_k). In particular, the positive semidefiniteness constraint $\mathbf{M}_{r_k}(g_k\boldsymbol{y}) \succeq 0$ is a linear matrix inequality (LMI) in $\boldsymbol{y}$, which implies that $\mathscr{M}(\mathscr{X})_r$ is semidefinite representable. When $k = 0$, the matrix

$$\mathbf{M}_r(\boldsymbol{y}) := \ell_{\boldsymbol{y}}(\boldsymbol{b}\boldsymbol{b}^T) \tag{27}$$

is called the moment matrix, where the linear functional acts entrywise on the rank-one matrix $\boldsymbol{b}\boldsymbol{b}^T$ with T denoting the transpose. Given $\boldsymbol{\alpha} \in \mathbb{N}^n_d$, the scalar $y_{\boldsymbol{\alpha}}$ is called the pseudo-moment of degree $\boldsymbol{\alpha}$.

The following result, a consequence of the so-called Putinar Positivstellensatz, states that the convex semidefinite representable sets defined above are nested outer approximations converging (up to taking the closure) to $\mathscr{M}(\mathscr{X})$:

Lemma 2 $\mathscr{M}(\mathscr{X})_r \supset \mathscr{M}(\mathscr{X})_{r+1} \supset \cdots \supset \overline{\mathscr{M}(\mathscr{X})_\infty} = \mathscr{M}(\mathscr{X})$.

Now consider the following hierarchy of SDP problems called moment relaxations, indexed by the relaxation order r:

$$p^*_r := \min_{\boldsymbol{y}\in\mathscr{M}(\mathscr{X})_r} \ell_{\boldsymbol{y}}(p). \tag{28}$$

This hierarchy generates a monotonically non-decreasing converging sequence of lower bounds[2] on the POP in Eq. (18):

Theorem 1 $p^*_r \leq p^*_{r+1} \leq \cdots \leq p^*_\infty = p^*$.

*A set is semidefinite representable if it is the linear projection of a linear section of a finite-dimensional cone of positive semidefinite matrices.

Let $\boldsymbol{y}^*$ denote the vector of pseudo-moments obtained by solving the moment relaxation of Eq. (28) for some given r. The objective is to determine whether $p_r^* = p^*$. The first candidate for global optimality is the vector of first degree pseudo-moments:

Lemma 3 *Let* $\boldsymbol{\chi}^\star := (y^\star_{\boldsymbol{\alpha}})_{|\boldsymbol{\alpha}|=1}$. *If* $\boldsymbol{\chi}^\star \in \mathscr{X}$ *and* $p(\boldsymbol{\chi}^\star) = p_r^\star$ *then* $p_r^\star = p^\star$.

Proof: Every admissible vector for the POP yields an upper bound on the value of the POP. If this upper bound is a lower bound, it means that it is optimal. □

Another useful property to check is whether the moment matrix has rank one:

Lemma 4 *If* $\operatorname{rank} \mathbf{M}_r(\boldsymbol{y}^\star) = 1$ *then* $p_r^\star = p^\star$.

Proof: From the definition of the moment matrix in Eq. (27), if $\operatorname{rank} \mathbf{M}_r(\boldsymbol{y}^\star) = 1$ for some $\boldsymbol{y}^\star$, then $\mathbf{M}_r(\boldsymbol{y}^\star) = \boldsymbol{b}(\boldsymbol{\chi}^\star)\boldsymbol{b}(\boldsymbol{\chi}^\star)^T = \boldsymbol{y}^\star(\boldsymbol{y}^\star)^T$. Since $\boldsymbol{y}^\star = \boldsymbol{b}(\boldsymbol{\chi}^\star)$ is admissible for the moment relaxation in Eq. (28), it follows that for all $k = 1, \ldots, m$, $\mathbf{M}_{r_k}(g_k \boldsymbol{y}^\star) = g_k(\boldsymbol{\chi}^\star)\boldsymbol{b}(\boldsymbol{\chi}^\star)\boldsymbol{b}(\boldsymbol{\chi}^\star)^T \succeq 0$ and hence $g_k(\boldsymbol{\chi}^\star) \geq 0$, which means that $\boldsymbol{\chi}^\star \in \mathscr{X}$ is admissible for the POP in Eq. (18). Since the value of the moment relaxation $p_r^\star = p(\boldsymbol{\chi}^\star)$ is a lower bound on the value of the POP, i.e. $p^\star$, and $\boldsymbol{\chi}^\star$ is admissible for the POP, it follows that $\boldsymbol{\chi}^\star$ is optimal and therefore $p_r^\star = p^\star$. □

A more general result called flat extension states:

Theorem 2 *For some* $r^* \geq \max(r_{\boldsymbol{g}}, \frac{d}{2})$, *if vector* $\boldsymbol{y}^*$ *is such that*

$$\operatorname{rank} \mathbf{M}_{r^*-r_{\boldsymbol{g}}}(\boldsymbol{y}^*) = \operatorname{rank} \mathbf{M}_{r^*}(\boldsymbol{y}^*)$$

then $p^*_{r^*} = p^*$.

Note that Lemma 4 is a particular case of Theorem 2, since if $\operatorname{rank} \mathbf{M}_{r^*}(\boldsymbol{y}^*) = 1$ for some r^* then $\operatorname{rank} \mathbf{M}_r(\boldsymbol{y}^*) = 1$ for all r. Similarly, Lemma 3 is a particular case of Theorem 2, since in this case the vector $\boldsymbol{y}^* = ((\boldsymbol{\chi}^*)^{\boldsymbol{\alpha}})_{\boldsymbol{\alpha}}$ is such that $\operatorname{rank} \mathbf{M}_r(\boldsymbol{y}^*) = 1$ for all r.

If the rank condition of Theorem 2 is satisfied, then there are $s := \operatorname{rank} \mathbf{M}_{r^*}(\boldsymbol{y}^\star)$ weights $w_k \geq 0$, $\sum_{k=1}^s w_k = 1$ and points $\boldsymbol{\chi}_k^\star \in \mathscr{X}$ such that $\mathbf{M}_r(\boldsymbol{y}^\star) = \sum_{k=1}^s w_k \boldsymbol{b}(\boldsymbol{\chi}_k^\star)\boldsymbol{b}(\boldsymbol{\chi}_k^\star)^T$ and $p^\star = p(\boldsymbol{\chi}_k^\star)$ for all $k = 1, \ldots, s$. Numerical linear algebra algorithms can then be applied on $\mathbf{M}_r(\boldsymbol{y}^\star)$ to extract the points $\boldsymbol{\chi}_k^\star$, $k = 1, \ldots, s$, which are all globally optimal for the POP in Eq. (18).

In practice, for increasing values of the relaxation order r, the moment relaxations in Eq. (28) are constructed with the help of modeling software such as GloptiPoly[9] or `MomentOpt.jl`[*]. These relaxations are solved numerically with an SDP solver such as Mosek[10] or `Hypatia.jl`.[11] The numerical solution is then post-processed to detect global optimality using the conditions of Lemmas 3 and 4 and Theorem 2, and to extract the global minimizers.

Second-degree relaxation of the targeting problem

Consider the POP given by Eqs. (4) and (10). Setting the expansion order $k = 4$ and using the notation of the previous paragraphs, the problem becomes:

$$\min_{\boldsymbol{\chi}} p(\boldsymbol{\chi}) = \chi_1^2 + \chi_2^2 + \chi_3^2 \tag{29}$$

[*] https://github.com/lanl-ansi/MomentOpt.jl

subject to

$$g_1(\boldsymbol{\chi}) = d^2 - \mathcal{T}^{(4)}_{d^2_M(t_f)}(\boldsymbol{\chi}) \geq 0, \tag{30}$$

where $\boldsymbol{\chi} \coloneqq \Delta\boldsymbol{v}$. Equation (29) is a second-order polynomial in $\boldsymbol{\chi}$, while Eq. (30) is a fourth-order polynomial in the same variables. Therefore, $r_{\boldsymbol{g}} = 2$ and the first relaxation has degree $r = 2$. The moment vector in the monomial basis read:

$$\boldsymbol{y} = \left[y_{\boldsymbol{\alpha}}\right] = \left[\boldsymbol{\chi}^{\boldsymbol{\alpha}}\right], \qquad \boldsymbol{\alpha} \in \mathbb{N}^3_4 \tag{31}$$

and contains the $\binom{3+4}{3} = 35$ moments from $y_{000} = 1$ to $y_{004} = \chi_3^4$. The POP given by Eqs. (4) and (10) is thus equivalent to the following moment optimization problem (MOP):

$$\min_{\boldsymbol{y}} p_M(\boldsymbol{y}) = y_{200} + y_{020} + y_{002} \tag{32}$$

subject to

$$\mathbf{M}_2(\boldsymbol{y}) = \begin{bmatrix} y_{000} & \cdots & y_{002} \\ \vdots & \ddots & \vdots \\ y_{002} & \cdots & y_{004} \end{bmatrix} \succeq 0 \tag{33a}$$

$$\mathbf{M}_0(g_1\boldsymbol{y}) = \boldsymbol{c}_{g_1}^T \boldsymbol{y} \tag{33b}$$

$$y_{000} = 1, \tag{33c}$$

where $\mathbf{M}_2(\boldsymbol{y})$ is the moment matrix, which is a 10×10 symmetric matrix constrained to be positive semidefinite (PSD), and $\boldsymbol{c}_{g_1}$ are the coefficients of Eq. (30) corresponding to the monomials $\boldsymbol{\chi}^{\boldsymbol{\alpha}}$. The localizing matrix $\mathbf{M}_0(g_1\boldsymbol{y})$ is linear in the optimization variables $\boldsymbol{y}$, and the MOP can be efficiently solved by any SDP solver.

CONVEX OPTIMIZATION

Convex optimization has become very popular in astrodynamics, as it allows to solve large optimization problems efficiently, while guaranteeing that the global optimum is found.[1] It consists in minimizing a convex function over a convex set, and it thus requires both the objective and the constraints to be convex functions in the optimization variables. In this section, a convex relaxation of the targeting problem is derived from the quadratic approximation of the constraint in Eq. (15).

Convex relaxation of the problem

Consider the optimization variables $\boldsymbol{z}$ defined in Eq. (13). After substituting these variables into Eq. (4), the objective function becomes:

$$\min_{\boldsymbol{z}} z_{200} + z_{020} + z_{002} \tag{34}$$

which is a linear function of $\boldsymbol{z}$. The constraints in Eqs. (2) and (3) are then replaced by Eqs. (15) and (17). As the latter are nonconvex functions of $\boldsymbol{z}$, they are relaxed into the following LMI:

$$\begin{bmatrix} 1 & z_{100} & z_{010} & z_{001} \\ z_{100} & z_{200} & z_{110} & z_{101} \\ z_{010} & z_{110} & z_{020} & z_{011} \\ z_{001} & z_{101} & z_{011} & z_{002} \end{bmatrix} \succeq 0, \tag{35}$$

where the notation $\succeq 0$ requires the matrix on the left-hand side (LHS) to be PSD. This matrix corresponds to the moment matrix of degree one, denoted as $\mathbf{M}_1(\boldsymbol{y})$ in the previous section. Therefore, it must be singular with rank one for the equality constraints in Eq. (17) to be satisfied. The convex relaxation of the original problem is finally given by Eqs. (15), (34) and (35).

Numerically robust formulation

The convex optimization problem presented above might be numerically ill-conditioned, as the matrix $\boldsymbol{Q}$ is not full rank, and its entries might span several orders of magnitude. To improve the numerical conditioning of the problem, a first step is to compute $\boldsymbol{Q}$ in a more robust way, i.e.:

$$\boldsymbol{B} = \boldsymbol{D}\boldsymbol{A}, \tag{36a}$$

$$\boldsymbol{Q} = \boldsymbol{B}^T\boldsymbol{B}, \tag{36b}$$

where $\boldsymbol{D}$ is the square root of $\boldsymbol{P}^{-1}$ and $\boldsymbol{B} \in \mathbb{R}^{6\times 9}$. This formulation is used in all numerical examples presented in this paper.

For an even better conditioning, the inequality constraint in Eq. (15) can be reformulated as follows. Firstly, an additional optimization variable $\beta \in \mathbb{R}$ is introduced, and the vector of optimization variables $\boldsymbol{z}$ is augmented to include β:

$$\tilde{\boldsymbol{z}} = \begin{bmatrix} \boldsymbol{z} \\ \beta \end{bmatrix}. \tag{37}$$

Then, Eq. (15) is split into two constraints:[12]

$$\|\boldsymbol{R}\boldsymbol{z}\|_2^2 \leq \beta, \tag{38a}$$

$$\beta + \boldsymbol{c}^T\boldsymbol{z} + r = 0. \tag{38b}$$

with $\boldsymbol{R} \in \mathbb{R}^{9\times 9}$ the upper triangular factor of the QR decomposition of $\boldsymbol{B}$ such that $\boldsymbol{Q} = \boldsymbol{R}^T\boldsymbol{R}$. The convex optimization problem is then defined by Eqs. (34), (35) and (38).

NUMERICAL APPLICATIONS

This section presents two applications of the polynomial optimization techniques to the targeting problem in astrodynamics. The objective is to demonstrate their competitiveness compared to the solution of a NLP and the use of polynomial map inversion techniques. All algorithms are implemented in Julia using different modeling frameworks and solvers. The polynomial expansions are computed with `GTPSA.jl`*, a Julia interface to the Generalized Truncated Power Series Algebra (GTPSA) library.[13] The original NLP is modeled using `Optimization.jl`,[14] as it integrates seamlessly with the `DifferentialEquations.jl`[15] package used to solve Eq. (2a). The MOP is modeled using the `MomentOpt.jl`†[16] and `SumOfSquares.jl`[16,17] packages, while the convex optimization problems are modeled using `Convex.jl`.[18] The NLP problem is solved with IPOPT,[6,19] while the SDP and convex ones are solved with Mosek.[10]

*https://github.com/bmad-sim/GTPSA.jl
†https://github.com/lanl-ansi/MomentOpt.jl

Keplerian dynamics

The first test case is the scenario introduced in Eqs. (1), (2) and (3). In this example the ballistic dynamics in Eq. (2a) is that of the restricted two-body problem (R2BP). After the initial maneuver, the following equations of motion (EOMs) govern the motion of the spacecraft:

$$\boldsymbol{f}(\boldsymbol{x}(t), t) = \begin{bmatrix} \boldsymbol{v} \\ -\frac{\mu}{\|\boldsymbol{r}\|_2^3}\boldsymbol{r} \end{bmatrix}, \tag{39}$$

where $\boldsymbol{x}(t)$ is given by Eq. (5) and μ is the standard gravitational parameter of the central body. The units are chosen such that $\mu = 1$, the nominal orbit radius is $r = 1$, and the nominal orbit period is $T = 2\pi$. At time $t_0 = 0$, the spacecraft has deviated from its nominal orbit, and its state is given by:

$$\begin{aligned} \boldsymbol{x}_0^- &= \bar{\boldsymbol{x}}_0 + \Delta\boldsymbol{x}_0 \\ &= \begin{bmatrix} 1 & 0 & 0 & 0 & \sqrt{1/2} & \sqrt{1/2} \end{bmatrix}^T + \begin{bmatrix} 10^{-5} & 10^{-5} & 10^{-5} & 0 & 0 & 0 \end{bmatrix}^T. \end{aligned} \tag{40}$$

The objective is to steer the spacecraft back to its nominal orbit in half orbit period, i.e. target the final state:

$$\bar{\boldsymbol{x}}_f = \begin{bmatrix} -1 & 0 & 0 & 0 & -\sqrt{1/2} & -\sqrt{1/2} \end{bmatrix}^T. \tag{41}$$

at $t_f = \pi$. The constraint in Eq. (3) is thus enforced with target distance $d = 0.1$ and covariance matrix $\boldsymbol{P}$ given by:

$$\boldsymbol{P} = \text{diag}\left(\begin{bmatrix} \sigma_r^2 & \sigma_r^2 & \sigma_r^2 & \sigma_v^2 & \sigma_v^2 & \sigma_v^2 \end{bmatrix}\right). \tag{42}$$

where $\sigma_r = 0.1$ and $\sigma_v = 10^{-3}\sigma_r$ are the standard deviations on the components of the position and velocity vectors, respectively. To improve the numerical conditioning of the problem, the DA variables in Eq. (7) are scaled by a constant coefficient $s = 10^{-3}$, i.e.:

$$[\Delta\boldsymbol{v}] = \boldsymbol{0}_{2\times 1} + s \cdot \delta\boldsymbol{v}, \tag{43}$$

and the flow in Eq. (8) is computed using Eq. (43) rather than Eq. (7). The optimal maneuver $\Delta\boldsymbol{v}^*$ is thus retrieved by multiplying the optimal solution to the corresponding problem by s. Four different problems are solved to compare the optimization techniques described in this paper. They are summarized in Table 1.

Table 1: Formulations of the optimization problems in the Keplerian dynamics scenario.

#	Defining equations	Type	Framework	Solver
A	Eqs. (2), (3) and (4)	NLP	`Optimization.jl`	IPOPT
B	Eqs. (32) and (33)	MOP	`MomentOpt.jl`	Mosek
C	Eqs. (15), (34) and (35)	convex	`Convex.jl`	Mosek
D	Eqs. (34), (35) and (38)	convex	`Convex.jl`	Mosek

The solutions to these problems are given in Table 2, in which both the components and magnitude of the optimal maneuvers are reported.

The violation of the terminal constraint in Eq. (3) and of the equality constraints in Eq. (17) are then assessed by computing the following quantities:

$$\epsilon_1 = \left[\boldsymbol{x}^*(t_f) - \bar{\boldsymbol{x}}_f\right]^T \boldsymbol{P}^{-1} \left[\boldsymbol{x}^*(t_f) - \bar{\boldsymbol{x}}_f\right] - d^2 \tag{44a}$$

$$\epsilon_2 = \left\| \boldsymbol{\zeta}^* \left(\boldsymbol{\zeta}^*\right)^T - \boldsymbol{Z}^* \right\|_\infty \tag{44b}$$

Table 2: Optimal maneuvers in the Keplerian dynamics scenario.

#	Δv_x^*	Δv_y^*	Δv_z^*	$\Vert\Delta \boldsymbol{v}^*\Vert_2$
A	$-9.020\,764 \times 10^{-6}$	$-6.836\,396 \times 10^{-6}$	$-6.836\,396 \times 10^{-6}$	$1.322\,296 \times 10^{-5}$
B	$-9.020\,835 \times 10^{-6}$	$-6.836\,352 \times 10^{-6}$	$-6.836\,352 \times 10^{-6}$	$1.322\,297 \times 10^{-5}$
C	$-9.020\,777 \times 10^{-6}$	$-6.836\,390 \times 10^{-6}$	$-6.836\,390 \times 10^{-6}$	$1.322\,297 \times 10^{-5}$
D	$-9.020\,624 \times 10^{-6}$	$-6.836\,491 \times 10^{-6}$	$-6.836\,491 \times 10^{-6}$	$1.322\,297 \times 10^{-5}$

where $\boldsymbol{x}^*(t_f)$ is computed from Eq. (2) using the optimal maneuver $\Delta \boldsymbol{v}^*$, $\boldsymbol{\zeta}^* = \begin{bmatrix} 1 & [z^*_{\boldsymbol{\alpha}}]^T_{|\boldsymbol{\alpha}|=1} \end{bmatrix}^T$, and $\boldsymbol{Z}^*$ is the LHS of Eq. (35) evaluated at $\boldsymbol{z}^*$. A negative value of ϵ_1 indicates that the final state is inside the target ellipsoid. Their values are reported in Table 3.

Table 3: Constraints violations in the Keplerian dynamics scenario.

#	ϵ_1	ϵ_2
A	$9.982\,015 \times 10^{-9}$	$0.000\,000$
B	$-7.528\,311 \times 10^{-10}$	$0.000\,000$
C	$-1.085\,589 \times 10^{-9}$	$7.321\,633 \times 10^{-12}$
D	$-1.202\,561 \times 10^{-9}$	$1.768\,767 \times 10^{-12}$

Tables 2 and 3 demonstrate that all methods converge to the same solution within a very small margin of error. The NLP and SOS formulations do not augment the free vector with the squares of the $\Delta \boldsymbol{v}$ components, and thus ϵ_2 is identically zero for problems A and B. The solutions to problems C and D also satisfy Eq. (17) within the tolerance of the underlying solver, meaning that the solution to the original problem is recovered with high accuracy. These results demonstrate that the proposed methods are a viable alternative to more widespread techniques, as they combine the accuracy of the NLP formulation with the efficiency and convergence guarantee needed for onboard applications.

CONCLUSIONS

This paper presented a reformulation of the targeting problem based on polynomial and convex optimization techniques. It also demonstrated that these techniques provide a solution that is as accurate as that obtained by solving the original NLP problem, while being more efficient and robust. The proposed methods are thus suitable for onboard applications, where computational resources are limited and convergence guarantees are paramount. Future work will focus on the extension of the proposed methods to multi-impulsive maneuvers scenarios and on their application to more complex dynamics such as those in cislunar regime.

ACKNOWLEDGMENT

This material is based on research sponsored by the Air Force Office of Scientific Research (AFOSR) under award number: FA9550-23-1-0646.

REFERENCES

[1] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, UK: Cambridge University Press, 1 ed., Mar. 2004, 10.1017/CBO9780511804441.

[2] J. B. Lasserre, "Global Optimization with Polynomials and the Problem of Moments," *SIAM Journal on Optimization*, Vol. 11, Jan. 2001, pp. 796–817, 10.1137/S1052623400366802.

[3] D. Henrion and J.-B. Lasserre, "Detecting Global Optimality and Extracting Solutions in GloptiPoly," *Positive Polynomials in Control* (D. Henrion and A. Garulli, eds.), Vol. 312 of *Lecture Notes in Control and Information Science*, pp. 293–310, Berlin, Heidelberg: Springer, 2005, 10.1007/10997703_15.

[4] J. B. Lasserre, *Moments, Positive Polynomials And Their Applications*, Vol. 1 of *Imperial College Press Optimization Series*. London, UK: Imperial College Press, 1 ed., 2009.

[5] M. Berz, *Modern Map Methods in Particle Beam Physics*. London, UK: Academic Press, 1999.

[6] A. Wächter and L. T. Biegler, "On the Implementation of an Interior-Point Filter Line-Search Algorithm for Large-Scale Nonlinear Programming," *Mathematical Programming*, Vol. 106, Mar. 2006, pp. 25–57, 10.1007/s10107-004-0559-y.

[7] D. Henrion, *Moments for polynomial optimization - An illustrated tutorial*. Recent Trends in Computer Algebra, Institut Henri Poincaré, Paris, 2023.

[8] A. Ben-Tal and A. Nemirovski, *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications*. MPS-SIAM Series on Optimization, Philadelphia, PA: Society for Industrial and Applied Mathematics (SIAM), 2001, 10.1137/1.9780898718829.

[9] D. Henrion, J.-B. Lasserre, and J. Löfberg, "GloptiPoly 3: Moments, Optimization and Semidefinite Programming," Sept. 2008.

[10] MOSEK ApS, "MOSEK version 11.0," 2025.

[11] C. Coey, L. Kapelevich, and J. P. Vielma, "Solving Natural Conic Formulations with Hypatia.jl," *INFORMS Journal on Computing*, Vol. 34, Sept. 2022, pp. 2686–2699, 10.1287/ijoc.2022.1202.

[12] MOSEK ApS, "MOSEK Modeling Cookbook," Sept. 2024.

[13] L. Deniau and C. Tomoiagă, "Generalised Truncated Power Series Algebra for Fast Particle Accelerator Transport Maps," *Proc. 6th International Particle Accelerator Conference (IPAC'15), Richmond, VA, USA, May 3-8, 2015*, International Particle Accelerator Conference, Geneva, Switzerland, JACoW, June 2015, pp. 374–377, 10.18429/JACoW-IPAC2015-MOPJE039.

[14] V. K. Dixit and C. Rackauckas, "Optimization.jl: A Unified Optimization Package," Zenodo, Mar. 2023, 10.5281/zenodo.7738525.

[15] C. Rackauckas and Q. Nie, "DifferentialEquations.jl – A Performant and Feature-Rich Ecosystem for Solving Differential Equations in Julia," *Journal of Open Research Software*, Vol. 5, May 2017, pp. 1–10, 10.5334/jors.151.

[16] T. Weisser, B. Legat, C. Coey, L. Kapelevich, and J. P. Vielma, "Polynomial and Moment Optimization in Julia and JuMP," *JuliaCon*, Baltimore, MD, July 2019.

[17] B. Legat, C. Coey, R. Deits, J. Huchette, and A. Perry, "Sum-of-Squares Optimization in Julia," *JuMP Developers Meetup/Workshop*, Cambridge, MA, June 2017.

[18] M. Udell, K. Mohan, D. Zeng, J. Hong, S. Diamond, and S. Boyd, "Convex Optimization in Julia," *2014 First Workshop for High Performance Technical Computing in Dynamic Languages*, New Orleans, LA, IEEE, Nov. 2014, pp. 18–28, 10.1109/HPTCDL.2014.5.

[19] "HSL. A collection of Fortran codes for large scale scientific computation," 2023.