Post-Print

This is the accepted version of:

P. Massioni, M. Massari
Convex Optimisation Approach to Constrained Fuel Optimal Control of Spacecraft in Close Relative Motion
Advances in Space Research, Vol. 61, N. 9, 2018, p. 2366-2376
doi:10.1016/j.asr.2018.02.020

Convex optimisation approach to constrained fuel optimal control of spacecraft in close relative motion

Paolo Massioni*

Laboratoire Ampère UMR CNRS 5005, INSA Lyon, Université de Lyon, Villeurbanne, France

Mauro Massari

Department of Aerospace Science and Technology, Politecnico di Milano, Via La Masa 34, 20156, Milan, Italy

---

Abstract

This paper describes an interesting and powerful approach to the constrained fuel-optimal control of spacecraft in close relative motion. The proposed approach is well suited for problems under linear dynamic equations, therefore perfectly fitting to the case of spacecraft flying in close relative motion. If the solution of the optimisation is approximated as a polynomial with respect to the time variable, then the problem can be approached with a technique developed in the control engineering community, known as "Sum Of Squares" (SOS), and the constraints can be reduced to bounds on the polynomials. Such a technique allows rewriting polynomial bounding problems in the form of convex optimisation problems, at the cost of a certain amount of conservatism. The principles of the techniques are explained and some application related to spacecraft flying in close relative motion are shown.

Keywords: fuel optimal control; constrained optimal control; close relative motion; convex optimisation; sum of squares.

---

*Corresponding author

Email addresses: `paolo.massioni@insa-lyon.fr` (Paolo Massioni), `mauro.massari@polimi.it` (Mauro Massari)

1. Introduction

The search for fuel-optimal manoeuvres is a classical problem in space engineering (Scharf et al., 2003), which is still thoroughly investigated by the aerospace community in search of more efficient and reliable methods, for different mission profiles (Li, 2016; Bolle and Circi, 2012; Qi and Jia, 2012). The problem is of critical interest due to the hard constraints on the quantity of fuel (and consequently, of delta-v) that a spacecraft can carry at launch. The classical analytical approach is based on Pontryagin's principle, which yields the classical bang-off-bang solutions (Kirk, 2012). Nevertheless, closed form solutions of fuel-optimal problems are often impossible to find, which makes it necessary the use of numerical optimisation methods.

The numerical solution of the optimal control problem, which is central to the fuel-optimal problem, can be found in two different ways, using indirect methods or direct methods. Indirect methods are based on the writing of the Hamiltonian function and on the solution of the Euler-Lagrange differential equation. In general they lead to very accurate results with the use of few variables. On the contrary, direct methods are based on the transcription of the differential problem into a pure parametric problem which can be solved using direct optimization methods. This kind of methods can lead to solutions as accurate as indirect methods but requires the use of many more variables. In both cases, the discrete problem can be faced with the algorithms developed for parameter optimization which are typically based on the Newton method (Betts, 1998). Example of indirect methods can be seen in (Casalino et al., 1999) and (Zhang et al., 2015), while example of direct methods can be seen in (Massari and Bernelli-Zazzera, 2009) and (Massari et al., 2003).

In general, both indirect and direct methods are very powerful, but being based on the Newton method, they require an initial solution guess to start the iterations. Moreover, this solution should be near enough to a local minimum to guarantee the convergence of the method to a solution. This shows also a second drawback of those methods, only local minima can be reached, no information on the globality of the optimum can be achieved.

The method presented in this paper belongs to the class of convex optimisation based methods, as do those based on Linear Programming (LP) (Magnani and Boyd, 2009) and moment measures (Claeys et al., 2014), which have also been applied to the problems described above. In this article, we explore an approach based on a technique known as Sum Of Squares (SOS)

(Parrilo, 2003), which lets one formulate polynomial optimisation problems in the form of a convex optimisation without any need of discretising the dynamical equations. With this technique, assuming that the solution has a polynomial expression, the problem can be cast into the form of an optimisation under Linear Matrix Inequality (LMI) constraints or Semi-Definite Programming (SDP), a form of convex optimisation that has been developed in the last decades in the context of automatic control (Boyd et al., 1994). The interest of this method is that it turns the problem into a convex one, in a very direct and simple way which is easily understandable even for the non-experts of the specific optimisation techniques involved. For this reason, this paper has also an introductory or tutorial part which allows a better understanding of the fundamentals.

As it will be explained later on, the reformulation of the problem required by the technique is done at the cost of a loss of precision, but on the other hand, the convex formulation does not require any initial guess, and it does not feature the risk of yielding local optima. The proposed technique clearly brings advantages with respect to classical indirect or direct approach to the solution of optimal control problems.

The paper is organised as follows. Section 2 introduces and formulates the problem. Section 3 and Section 4 contain a short tutorial for explaining the ideas behind Sum Of Squares (SOS) and Linear Matrix Inequalities (LMIs) techniques, which we think improve the readability of this paper, but they can be skipped by those who are already familiar. Section 5 contains the baseline algorithm that is the main result of this article, whereas Section 6 introduces a few variants on it. Section 7 shows a set of application to spacecraft in close relative motion and finally Section 8 draws the conclusions.

Notation

We denote by $\mathbb{N}$ the set of non-negative integers, by $\mathbb{R}$ the set of real numbers and by $\mathbb{R}^{n\times m}$ the set of real $n \times m$ matrices. $\mathbf{R}_m[x]$ is the set of real-valued polynomials of degree m in the entries of x, $A^\top$ indicates the transpose of a matrix A; the notation $A \succeq 0$ (resp. $A \preceq 0$) indicates that all the eigenvalues of the symmetric matrix A are positive (resp. negative) or equal to zero. The symbol $\begin{pmatrix} n \\ k \end{pmatrix}$ indicates the binomial coefficient, for which we have

$$\begin{pmatrix} n \\ k \end{pmatrix} = \frac{n!}{k!(n-k)!}.$$

For the reader's convenience, all the symbols of this paper with exception of those used in the examples are listed at the end in the Appendix.

2. Problem formulation

We consider linear dynamic equations describing the motion of one or more point masses, of the kind

$$\ddot{x}(t) = f(x(t)) + u(t) \tag{1}$$

where t is the time variable, $x(t) = [x_1(t), \ldots, x_n(t)]^\top \in \mathbb{R}^n$ the position vector (with $n \in \mathbb{N}$), $u(t) = [u_1(t), \ldots, u_n(t)]^\top \in \mathbb{R}^n$ a vector of control actions and $f(x(t)) = [f_1(x(t)), \ldots, f_n(x(t))]^\top$ a vector-valued linear function coming from the physics of the problem. The typical fuel-optimal problem consists in finding a trajectory $x^*(t)$ which brings the state from an initial position x_0 and velocity v_0 at time $t = 0$, to a final position x_f and velocity v_f at a fixed time t_f, minimising the time integral of a one-norm of $u^*(t) = \ddot{x}^*(t) - f(x^*(t))$. This can be formulated formally as follows.

Problem 1 (Fuel-optimal control). *Given (1), $t_f > 0$, $u_{max,i} > 0$, x_0, v_0, x_f, v_f, find a continuous and derivable function $x^*(t) : [0, t_f] \mapsto \mathbb{R}^n$ such that*

$$\int_0^{t_f} \sum_i |u_i^*(t)| \, dt \text{ is minimised} \tag{2}$$

under $x^*(0) = x_0$, $\dot{x}^*(0) = v_0$, $x^*(t_f) = x_f$, $\dot{x}^*(t_f) = v_f$, $|u_i^*(t)| \leqslant u_{max,i}$, with $u_i^*(t) = \ddot{x}_i^*(t) - f_i(x^*(t))$.

Notice that by setting one of the $u_{max,i}$ as very small or close to zero, one can take into account situations where not all the directions of the space are directly actuated, i.e. the cases in which $u_i(t) = 0$ for a few (not all) values of i.

The methods discussed in this paper cannot deal directly with Problem 1, but rather with a relaxation of it. By "relaxing a problem", we mean replacing the original problem with a second one that converges to the first under certain hypotheses. The advantage of doing so is that the second problem is amenable to a new approach, and it is formulated as follows.

Problem 2 (Relaxed fuel-optimal control). *Given* (1), $t_f > 0$, $u_{max,i} > 0$, x_0, v_0, x_f, v_f, $N \in \mathbb{N}$, $d \in \mathbb{N}$, *find a piecewise-polynomial vector-valued function* $x^*(t) : [0, t_f] \mapsto \mathbb{R}^n$ *defined as*

$$x_i^*(t) = p_{i,j}(t) \text{ for } (j-1)t_f/N \leqslant t < jt_f/N, i = 1, ..., n \tag{3}$$

with $p_{i,j}(t) \in \mathbf{R}_{2d}[t]$, *such that*

$$J = \frac{t_f}{N} \sum_{i,j} \gamma_{i,j} \text{ is minimised} \tag{4}$$

under

1. $x^*(0) = x_0$, $\dot{x}^*(0) = v_0$
2. $p_{i,j}(jt_f/N) = p_{i,j+1}(jt_f/N)$, $\dot{p}_{i,j}(jt_f/N) = \dot{p}_{i,j+1}(jt_f/N)$ *for* $j = 1, ..., N-1$, $i = 1, ..., n$
3. $x^*(t_f) = x_f$, $\dot{x}^*(t_f) = v_f$,
4. $-\gamma_{i,j} \leqslant u_i^*(t) \leqslant \gamma_{i,j}$, *for* $(j-1)t_f/N \leqslant t < jt_f/N$
5. $0 \leqslant \gamma_{i,j} \leqslant u_{max,i}$, *for* $i = 1 ..., n$, $j = 1, ..., N$,

again with $u_i^*(t) = \ddot{x}_i^*(t) - f_i(x^*(t))$.

The quantity J in (4) is the upper bound to the fuel consumption, to be minimised. The minimisation has been shifted from the absolute value of u, a non-convex function, to the sum of a set of decision variables $\gamma_{i,j}$, which is convex; this is a standard trick for yielding convex optimisation problems, as explained for example in (Boyd and Vandenberghe, 2004). The time interval $[0, t_f]$ is divided into N intervals, where each function $x_i^*(t)$ is assumed to be of a different polynomial form $p_{i,j}(t)$ with respect to time, with degree $2d$ (the factor 2 simplifies the notation later on). The first and the third constraints set the initial and final values of the function $x^*(t)$ and its derivative. The second constraint forces $x^*(t)$ to be continuous and derivable (i.e. its derivative is continuous). The fourth constraint states that the control action u_i is bounded in modulus by $\gamma_{i,j}$ in each interval where $x_i^*(t) = p_{i,j}(t)$; the fifth constraint states that $\gamma_{i,j}$ is bounded too by the maximum control action $u_{max,i}$. The optimal control action $u^*(t)$ is retrieved from the optimal trajectory $x^*(t)$ by inversion of (1) (as commonly done in inverse dynamics techniques).

Notice that for $N \to \infty$, the polynomial approximation of x^* can converge to any continuous derivable function, and the sum in (4) converges to a tight upper bound for the integral in (4) of Problem 1. In this sense, Problem 2 converges to Problem 1 for $N \to \infty$.

As it will be shown in Section 5, Problem 2 can be formulated as a Sum Of Squares (SOS) problem, which can be turned into an optimisation under linear matrix inequality (LMI) constraints. LMI optimisations are convex problems, which means that the global optimal solution can be found efficiently by an appropriate solver, without any risk of finding local optima and without the need of providing an initial guess.

The next two sections provide some background for the reader who is unfamiliar with SOS and LMIs. They can be skipped by the experts.

3. Sum Of Squares

The basic idea of the Sum Of Squares (SOS) (Parrilo, 2003) technique is very simple: one can prove that a polynomial is non-negative (positive or zero) for any values of its variables if it can be written as the sum of square terms.

Example 1 (Polynomial as sum of squares). *Prove that* $p_{ex1}(x) = x_1^2 - 2x_1x_2 - 2x_1 + 2x_2^2 + 3 \geqslant 0$ *for all* x_1, x_2.
Solution*:* $p_{ex1}(x) = (x_1 - x_2 - 1)^2 + (x_2 - 1)^2 + 1^2$.

The problem of finding a lower bound for the minimum value of a polynomial can be also cast as an SOS problem, i.e. the lower bound σ of the minimum value of a polynomial $p(x)$ (such as $p(x) \geqslant \sigma \; \forall x$) is obtained by solving the following optimisation problem

$$\text{maximize } \sigma \text{ such as } p(x) - \sigma \text{ is SOS.} \tag{5}$$

Example 2 (Lower bound of a polynomial). *Find the biggest value of* σ *for which* $p_{ex1}(x) \geqslant \sigma$.
Solution*:* $p_{ex1}(x) - \sigma = (x_1 - x_2 - 1)^2 + (x_2 - 1)^2 + 1^2 - \sigma$*; the maximum* σ *allowed is* 1*, in fact for* $\sigma > 1$ *and* $x_1 = 2, x_2 = 1$*,* $p_{ex1}(x) - \sigma$ *becomes negative.*

One last interesting class of problems consist in assessing whether a polynomial is positive for values of its variables in a given interval or satisfying a set of constraints. Such problems can be cast into SOS form by using a simple property known as S-procedure in the automatic control literature (or also known as Positivstellensatz in the context of the theory of polynomials). The S-procedure allows restricting a variable-dependent inequality to the subset of variables satisfying another inequality.

Lemma 1 (S-procedure).

$$\begin{array}{c} f(x) - \tau g(x) \geqslant 0 \quad \forall x, \tau \geqslant 0 \\ \Downarrow \\ f(x) \geqslant 0 \quad \textit{for } x \in \{x | g(x) \geqslant 0\}. \end{array} \tag{6}$$

Example 3 (Lower bound for bounded variables). *Find the biggest σ for which $p_{ex2}(x) = -x_1^2 + 2x_1 \geqslant \sigma$, with $||x_1|| \leqslant 1$.*
Solution*: $p_{ex2}(x) - \sigma = -x_1^2 + 2x_1 - \sigma \geqslant 0$ for $1 - x_1^2 \geqslant 0$ is implied by (S-procedure) $-x_1^2 + 2x_1 - \sigma - \tau_1(1 - x_1^2) \geqslant 0$ with $\tau_1 \geqslant 0$. We notice that if we pick $\tau_1 = 2$, we have $-x_1^2 + 2x_1 - \sigma - \tau_1(1 - x_1^2) = x_1^2 + 2x_1 - 2 - \sigma = (x_1 + 1)^2 - 3 - \sigma$ which is SOS for $\sigma \leqslant -3$. So -3 is the lower bound of the polynomial in the interval $-1 \leqslant x_1 \leqslant 1$.*

These three examples can be convincing about the fact that SOS can deliver results, but on the other hand, it is indeed not always obvious how to find an SOS decomposition of a complex multivariate polynomial, if it exists. Moreover, the result might be affected by "conservatism", i.e. it might be that the polynomial is indeed always positive but no SOS formulation exists (or, if it exists, we cannot find it), or the lower bound might be underestimated.

The good news is that the search for an SOS decomposition can be done numerically, and even better, the optimisation problems which stem from such a search turn out to be convex. In fact, we can prove that an SOS formulation of a polynomial can be found through an optimisation under Linear Matrix Inequality (LMI) (Boyd et al., 1994) constraints, a convex problem well known in the automatic control literature.

4. Linear matrix inequalities and sum of squares

The optimisation problems under LMI constraints in which we are interested can be formalised as

$$\text{minimise } g(\nu) \text{ under } F(\nu) \succeq 0 \tag{7}$$

where ν is a vector of unknowns, g is a scalar function and F a symmetric matrix valued function, both affine in ν; the expression $A \geq 0$ ($A \leq 0$) indicates that the symmetric matrix A is positive (negative) semi-definite, i.e. its eigenvalues are positive (negative) or equal to 0. Such problems are convex and can be solved efficiently; a very popular approach relies on using Matlab-based solvers, typically SeDuMi (Sturm, 1999) with the help of the Yalmip (Löfberg, 2004) user-friendly interface. Notice that equality constraints of the kind $h(\nu) = 0$, with h affine in ν, can be taken into account as well by a simple change of base for the space of the unknowns.

The key point here is that SOS problems can be turned into LMI problems. First of all, we notice that for a polynomial $p_{2m}(x)$ of degree $2m$, we can always find a "quadratic" formulation of the kind $p_{2m}(x) = \chi(x)^\top P\chi(x)$, with $\chi(x)$ a vector containing all the possible monomials in the x variables from degree 0 to degree m; P is a square symmetric matrix, which is not uniquely defined thanks to the fact that products of different couples of entries in $\chi(x)$ can yield the same result. So in general $P = P(\mu) = P_0 + \sum_{i=1}^{\iota} \mu_i Q_i$, with $\mu \in \mathbb{R}^\iota$ a vector of so-called "slack variables" which can assume any value, as they multiply constant matrix terms Q_i simplifying to 0. The following example clarifies this notion.

Example 4 (Slack variables). *Express the polynomial* $p_{ex3}(x) = 5x_1^4 + 4x_1^2 + 2x_1 + 3$ *as* $\chi(x_1)^\top P(\mu)\chi(x_1)$.
Solution*: considering that the polynomial is of degree* 4*, we need* χ *at least of degree* 2*, i.e.* $\chi(x_1) = [1\ x_1\ x_1^2]^\top$. *Noticing that* $x_1 \cdot x_1 = 1 \cdot x_1^2 = x_1^2 \cdot 1$*, we can then write*

$$p_{ex3}(x)=\begin{bmatrix} x_1^2 \\ x_1 \\ 1 \end{bmatrix}^\top \begin{bmatrix} 5 & 0 & \mu_1+2 \\ 0 & -2\mu_1 & 1 \\ \mu_1+2 & 1 & 3 \end{bmatrix} \begin{bmatrix} x_1^2 \\ x_1 \\ 1 \end{bmatrix} \tag{8}$$

with μ_1 *a slack variable* ($\iota = 1$ *in this case*).

Once the quadratic formulation has been found, it is enough to find one value μ^* of the slack variables μ for which the matrix $P(\mu^*) \geq 0$ for proving that the polynomial is SOS. In fact, first of all, if we have $p_n(x) = \chi(x)^\top P(\mu^*)\chi(x)$, with $P(\mu^*) \geq 0$, it is obvious that $p_n(x) \geqslant 0$ for all χ and so for all x; at the same time, if $P(\mu^*) \geq 0$ it means that there exists a matrix square root S such as $S^\top S = P(\mu^*)$, so if we define $\theta = S\chi$, then $p_n(x) = \theta^\top\theta$, i.e. the sum of the squares of the entries of the column vector θ. In this way, an SOS problem becomes the problem of finding a μ for which $P(\mu) \geq 0$, which is called an LMI feasibility problem (as it is not really an optimisation, no function is minimised). Adding additional constraints with the S-procedure does not change the type of problem, it just adds the additional unknowns (or "decision variables") τ. Adding an objective function to minimise (e.g. the lower bound times -1) turns the problem into a true optimisation under LMI constraints, as shown in the following last example.

Example 5 (SOS as LMI). *Estimate the lower bound σ for $p_{ex3}(x)$, with $||x_1|| \geqslant 1$.*
Solution*: adding the S-procedure term $-\tau_1(1-x_1^2)$, with $\tau_1 \geqslant 0$, we have that $p_{ex3}(x) - \sigma \geqslant 0$ for*

$$\begin{bmatrix} 5 & 0 & \mu_1+2 & 0 \\ 0 & -2\mu_1+\tau_1 & 1 & 0 \\ \mu_1+2 & 1 & 3-\sigma-\tau_1 & 0 \\ 0 & 0 & 0 & \tau_1 \end{bmatrix} \succeq 0 \tag{9}$$

with μ_1, τ_1, σ unknown. This time we code the problem into Matlab with the help of Yalmip:

```
>> m1 = sdpvar(1);
>> t1 = sdpvar(1);
>> s = sdpvar(1);
>> P = [5,0,m1+2,0; 0,t1-2*m1,1,0; m1+2,1,3-s-t1,0; 0,0,0,t1];
>> lmi = (P >= 0);
```

and we solve it with SeDuMi, minimising $-\sigma$ (which means, maximising σ):

```
>> optimize(lmi, -s, sdpsettings('solver','sedumi'));
>> value(s)
```

The solver finds the result of $\sigma = 2.7653$, consistent with the plot of $p_{ex3}(x)$ in the $[-1, 1]$ interval. (Fig. 1). The convexity of the problem ensures that this global optimal value is obtained for one unique value of all the decision variables, which can be found efficiently by the solver.

5. Fuel-optimal problems as sum of squares optimisation

After reading the previous sections, the reader should be aware that any optimisation where the cost function is linear in the unknowns, under inequality constraints concerning polynomials (whose coefficients are also linear in the unknowns) can be cast into an SOS/LMI form. The polynomial

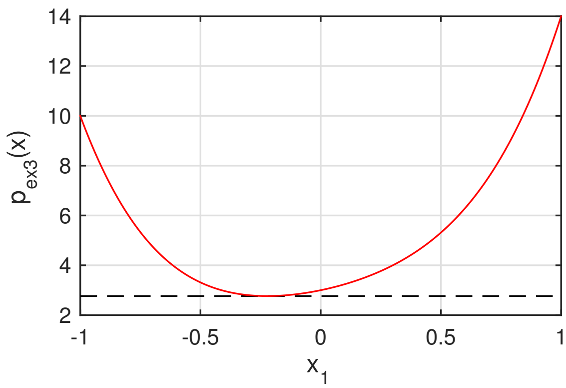

Figure 1: Plot of $p_{ex3}(x)$ and its lower bound (dashed).

constraints can be specialised for t belonging to an interval by the use of the S-procedure (Lemma 1). The coefficient τ used in the S-procedure can also be a positive polynomial $\tau(x)$, still leading to an LMI formulation of the problem. Looking at Problem 2, it is apparent that the cost function is linear in the unknowns $\gamma_{i,j}$, and that all the constraints are of the kind allowed in an SOS problem. The piecewise-polynomial function $x_i^*(t)$ in (3) has to be formulated by assuming

$$p_{i,j}(t) = \xi(t)^\top P_{i,j} \xi(t) \tag{10}$$

with $\xi(t) = [1,\, t,\, \dots t^d]^\top \in \mathbb{R}^{d+1}$. We are in the case of univariate polynomials (in t), for which the number of slack matrices ι is given by (see Parrilo, 2003):

$$\iota = \frac{1}{2}\left(\binom{d+1}{d}^2 + \binom{d+1}{d}\right) - \binom{1+2d}{2d} = \frac{d^2-d}{2}. \tag{11}$$

The steps that need to be taken to reformulate Problem 2 as a convex optimisation are expressed intuitively in the list that follows (the rigorous expressions are reported just afterwards).

- Express the optimal trajectory x^* as a piecewise polynomial function of time t of unknown coefficients (to be determined); this function is made of N polynomial segments of degree $2d$.
- Enforce the constraints in the optimal trajectory, i.e. initial position, final position, continuity, derivability. This results in linear relation

between the unknown coefficients, which are equivalent to the removal of some of the unknowns.

- The optimal control action u^* can be retrieved by inverting (1); this is a piecewise polynomial function too, whose coefficients depend linearly on the coefficients of x^*.
- Set an upper bound for each $|u_i^*|$, for each segment of the piecewise polynomial function; this can be done, for each polynomial segment of index j, turning the inequality $|u_i^*| \leqslant \gamma_{j,i}$ (nonconvex) into $-\gamma_{j,i} \leqslant u_i^* \leqslant \gamma_{j,i}$ (convex).
- Set the upper bound for each $\gamma_{j,i}$, i.e. $\gamma_{j,i} \leqslant u_{max,i}$.
- By using the SOS technique and the S-procedure, the inequalities $-\gamma_{j,i} \leqslant u_i^* \leqslant \gamma_{j,i}$ for each time interval are turned equivalently into a set of LMIs.
- Minimising the sum of all the $\gamma_{j,i}$ under the constraints found above allows finding the optimal solution through a convex optimisation problem.

All of this leads to Problem 3, summarised here.

Problem 3 (Relaxed fuel-optimal control, convex formulation). *Given* (1), $t_f > 0$, $u_{max,i} > 0$, x_0, v_0, x_f, v_f, $N \in \mathbb{N}$, $d \in \mathbb{N}$, *find a value for the following unknowns:*

- $P_{i,j} = P_{i,j}^\top \in \mathbb{R}^{(d+1)\times(d+1)}$, *for* $j = 1, ..., N$, $i = 1, ..., n$;
- $\Omega_{i,j,k} = \Omega_{i,j,k}^\top \in \mathbb{R}^{(d+1)\times(d+1)}$, *for* $j = 1, ..., N$, $i = 1, ..., n$, $k = 1, 2$;
- $\mu_{i,j,k,l}$ *for* $j = 1, ..., N$, $i = 1, ..., n$, $k = 1, 2$, $l = 1, ..., \iota$;
- $\gamma_{i,j}$, *for* $j = 1, ..., N$, $i = 1, ..., n$;

under

1. $\xi(0)^\top P_{i,1}\xi(0) = (x_0)_i$, $\xi(0)^\top P'_{i,1}\xi(0) = (v_0)_i$, *for* $i = 1, ..., n$ *(initial conditions, equality constraints);*
2. $\xi(jt_f/N)^\top(P_{i,j} - P_{i,j+1})\xi(jt_f/N) = 0$, $\xi(jt_f/N)^\top(P'_{i,j} - P'_{i,j+1})\xi(jt_f/N) = 0$, *for* $j = 1, ..., N-1$, $i = 1, ..., n$ *(continuity and derivability, equality constraints);*

3. $\xi(t_f)^\top P_{i,N}\xi(t_f) = (x_f)_i$, $\xi(t_f)^\top P'_{i,N}\xi(t_f) = (v_f)_i$, *for* $i = 1, ..., n$ *(final conditions, equality constraints);*

4.1.1. $U_{i,j} + \mathbb{I}\gamma_{i,j} + \sum_{l=1}^{\iota} \mu_{i,j,1,l} Q_l - \Xi_{i,j,1} \geq 0$, *for* $i = 1, ..., n$, $j = 1, ..., N$ *(linear matrix inequalities)*

4.1.2. $-U_{i,j} + \mathbb{I}\gamma_{i,j} + \sum_{l=1}^{\iota} \mu_{i,j,2,l} Q_l - \Xi_{i,j,2} \geq 0$, *for* $i = 1, ..., n$, $j = 1, ..., N$ *(linear matrix inequalities)*

4.2.1. $\xi(t)^\top \Omega_{i,j,k}\xi(t) \in \mathbf{R}_{2d-2}[t]$, *for* $j = 1, ..., N$, $i = 1, ..., n$, $k = 1, 2$ *(equality constraints)*

4.2.2. $\Omega_{i,j,k} \geq 0$, *for* $j = 1, ..., N$, $i = 1, ..., n$, $k = 1, 2$ *(linear matrix inequalities)*

5.1. $\gamma_{i,j} \geqslant 0$, *for* $j = 1, ..., N$, $i = 1, ..., n$ *(scalar inequalities)*

5.2. $\gamma_{i,j} \leqslant u_{max,i}$, *for* $j = 1, ..., N$, $i = 1, ..., n$ *(scalar inequalities)*

such that

$$J = \frac{t_f}{N} \sum_{i,j} \gamma_{i,j} \text{ is minimised} \tag{12}$$

with

- $P'_{i,j}$ *such that* $\xi(t)^\top P'_{i,j}\xi(t) = \frac{d}{dt}(\xi(t)^\top P_{i,j}\xi(t))$,
- $U_{i,j}$ *such that* $\xi(t)^\top U_{i,j}\xi(t) = \frac{d^2}{dt^2}(\xi(t)^\top P_{i,j}\xi(t) - f_i(x(t)))$,
- $\mathbb{I}$ *such that* $\xi(t)^\top \mathbb{I}\xi(t) = 1$,
- $\Xi_{i,j,k}$ *such that* $\xi(t)^\top \Xi_{i,j,k}\xi(t) =$
  $\xi(t)^\top \Omega_{i,j,k}\xi(t)^\top (1 - (2tN/t_f - (2j-1))^2)$,
- Q_l *for* $l = 1, ..., \iota$ *such that* $\xi(t)^\top Q_l \xi(t) = 0$ *(slack matrices).*

The optimal solution sought is given by

$$x_i^*(t) = \xi(t)^\top P_{i,j}\xi(t) \text{ for } (j-1)t_f/N \leqslant t < jt_f/N, i = 1, ..., n. \tag{13}$$

Notice that the terms $P'_{i,j}$ and $U_{i,j}$ are just linear recombinations of the unknowns in the matrices $P_{i,j}$, so linear matrix constraints containing them are linear also in the original unknowns. The constraints in Problem 3 are directly related to those in Problem 2.

The term J in (12) is the integral of the sum of all the upper bounds ($\gamma_{i,j}$) of fuel consumption in each interval. It gives an upper bound for the global propellant consumption, which has to be minimised.

Constraints 1 to 3 are obvious.

Constraint 4 of Problem 2 is split in two inequalities: 4.1.1 assures that $u_i(t) \geqslant -\gamma_{i,j}$ in the interval, whereas 4.1.2 assures that $u_i(t) \leqslant \gamma_{i,j}$. The terms $\Xi_{i,j,k}$ are the S-procedure terms for the interval bounds of the polynomials. First notice that $(1-(2tN/t_f-(2j-1))^2)$ is positive if and only if t is in the j^{th} time interval; $\Xi_{i,j,k}$ is obtained multiplying this expression with the positive multiplier $\xi(t)^\top \Omega_{i,j,k}\xi(t)$. Notice that a slack term $\sum_{l=1}^{\iota} \mu_{i,j,k,l} Q_l$ is present in 4.1.1 and in 4.1.2. To better see how 4.1.1 and 4.1.2 work, it is sufficient to multiply them on both sides by $\xi(t)$; looking for example at 4.1.1, we have that $\xi(t)^\top U_{i,j}\xi(t)$ is basically u_i in the j^{th} interval; $\xi(t)^\top \mathbb{I}\gamma_{i,j}\xi(t) = \gamma_{i,j}$; $\xi(t)^\top \sum_{l=1}^{\iota} \mu_{i,j,2,l} Q_l \xi(t) = 0$ by definition of Q_l, and $\xi(t)^\top \Xi_{i,j,k}\xi(t)$ is positive when $((1-(2tN/t_f-(2j-1))^2)$ is positive, i.e. when t is in the j^{th} interval. This boils down to forcing $u_i+\gamma_{i,j}$ to be positive when t is in the j^{th} interval, which implies $u_i \geqslant -\gamma_{i,j}$. A similar reasoning leads 4.1.2 to imply $u_i \leqslant \gamma_{i,j}$ in the j^{th} interval.

Constraint 4.2.2 assures the positivity of the polynomial multipliers, once constraint 4.2.1 has assured that their degree is two less than the maximum which can be expressed by the quadratic formulation (i.e. $2d$). This makes sure that no terms of order greater than $2d$ appear when multiplying $\xi(t)^\top \Omega_{i,j,k}\xi(t)$ with the (quadratic) constraint on the time, making sure that the terms $\Xi_{i,j,k}$ exist.

Constraints 5.1 and 5.2 translate constraint 5 of Problem 2 (they are scalar inequalities, which are a special case of linear matrix inequalities).

The optimal trajectory in (13) is obtained by combining the optimal trajectory in each time interval, remembering that $P_{i,j}$ is the matrix formulation of its polynomial expression as explained in Section 4 or in (10). From the optimal trajectory, one can recover the optimal control action as $u_i^*(t) = \ddot{x}_i^*(t) - f_i(x^*(t))$.

Remark 1 (on the conservatism of Problem 3). *As said, the SOS formulation of an optimisation problem is conservative, in the sense that a certain degree of effectiveness or precision is lost when a positive polynomial is approximated by an SOS polynomial. In the univariate case though, which is the case here, all positive polynomials are SOS (Lasserre, 2009, page 22), so no loss of precision is due to the SOS formulation itself when* *going from Problem 2 (SOS) to Problem 3 (LMIs). On the other hand, the S-procedure (Lemma 1) is in most cases conservative, but Markov-Lukacs's thorem (Genin et al., 2000) states that there is no conservatism in our special case here, with univariate polynomials in t and multipliers of degree $2d-2$. So the passage from SOS to LMI is exact, i.e. Problem 3 is equivalent to Problem 2 (we do not go into the details here as we think that it would be out of the scope of the paper). The only approximations are in the relaxation from Problem 1 to Problem 2, which becomes more and more precise with increasing N and d. In practice it can be pointed out that no significant problems arise even at low d (e.g. $d = 2$), for a sufficient value of N.*

Remark 2 (computational complexity). *The computational cost of an LMI optimisation in terms of number of Floating Point Operations (FLOPs) depends on the specific solver used; we can estimate in general that it is proportional to the third power of the number of scalar unknowns (Gahinet et al., 1994). Such a number can be quickly computed. Looking at Problem 3, there are*

- $(2d+1)Nn$ *unknowns in the* $P_{i,j}$ *matrices;*
- $(d^2+d)Nn$ *unknowns in the* $\Omega_{i,j,k}$ *matrices;*
- $(d^2-d)Nn$ *unknowns in the* $\mu_{i,j,k,l}$ *variables;*
- Nn *unknowns in the* $\gamma_{i,j}$ *variables,*

where ι is given by (11) as a function of d. Notice that P', $U_{i,j}$ and $\Xi_{i,j,k}$ do not contribute to the number of the unknowns as their entries are just linear combination of the variables listed above. Notice also that the equality constraints (initial, final condition, continuity and derivability) remove $2(N+1)n$ unknowns. This leads to the following grand total for the number of unknowns

$$N_{unk} = (2d^2 + 2d - 1)Nn - 2n. \tag{14}$$

The number of unknown grows linearly with respect to N and n, once d is fixed. In general, LMI solvers can deal with a few thousands of unknowns without problems. Notice also that in most cases, it is convenient to operate a change in the time variable, replacing t with $t' = 2tN/t_f - (2j-1)$ in each time interval. This normalises the time within each interval, improving the precision of the numerical solution.

In the application section at the end of this paper (Section 7), the number of unknowns is reported as well as the time necessary to solve the optimisation with a standard quad-core personal computer with a 2.67 GHz processor and 8 GB of random access memory.

6. Additional features

Problem 3 can be upgraded in order to include additional features. The most relevant possibilities are listed here.

6.1. Extra state constraints

In each of the N intervals into which the polynomial function is divided, one can enforce any number of additional state constraints of the kind

$$L_k^\top x \geqslant c_k(t) \text{ for } (j(k)-1)t_f/N \leqslant t < j(k)t_f/N \tag{15}$$

where $j(k)$ tells in which interval j the k-th constraint is located, and with $L_k = [L_{k,1}, \dots L_{k,n}]^\top \in \mathbb{R}^n$ and $c_k(t) = \xi(t)^\top C_k \xi(t) \in \mathbf{R}_{2d}[t]$ (a polynomial in t, it can be chosen as a constant). The constraint is obtained by declaring the extra unknowns

- $\Delta_k = \Delta_k^\top \in \mathbb{R}^{(d+1)\times(d+1)}$, for $k = 1, \dots, k_{max}$, ;
- $\eta_{k,l}$, for $k = 1, \dots, k_{max}$, $l = 1, \dots, \iota$.

Defining Π_k such that $\xi(t)^\top \Pi_k \xi(t) = \xi(t)^\top \Delta_k \xi(t)^\top (1 - (2tN/t_f - (2j-1))^2)$ (notice that Π_k and Δ_k are analogous to $\Xi_{i,j,k}$ and $\Omega_{i,j,k}$), the constraints to be added to Problem 3 are

6.1 $\sum_{i=1}^{n} L_{k,i} P_{i,j} - C_k + \sum_{l=1}^{\iota} \eta_{k_j} Q_l - \Pi_k \geq 0$ for $k = 1, \dots, k_{max}$ (linear matrix inequalities)
6.2 $\xi(t)^\top \Delta_k \xi(t) \in \mathbf{R}_{2d-2}[t]$ for $k = 1, \dots, k_{max}$ (equality constraints)
6.3 $\Delta_k \geq 0$ for $k = 1, \dots, k_{max}$ (linear matrix inequalities).

Each state constraint of this kind limits the accessible zone for the trajectory to time-varying half-spaces. Constraints of a similar kind, which force a state at a specific time instant to belong to an half-space are also possible, and they are easily implemented with a simple scalar inequality (with no extra unknowns). The constraints can be combined together to obtain relevant meaningful constraints, like for example imposing safety distance constraints. The fourth application example in Section 7 shows how these kind of constraints can be exploited.

The ability to deal with inequality state constraints on the entire trajectory makes the approach especially suitable for the case of spacecraft in close relative motion. In those cases, the typical constraints can arise form collision avoidance as well as approach geometry in the case of docking.

6.2. Linear time-varying dynamics

Problem 3 can be also adjusted to account for linear time-varying dynamics, i.e. if (1) is replaced by

$$\ddot{x}(t) = f(x(t), t) + u(t) \tag{16}$$

where f depends explicitly from the time; if f is linear with respect to the state $x(t)$ and polynomial with respect to t, the optimisation problem can still be cast into an SOS form. The only care to be taken is that the expression of $f(x(t), t)$ is of degree smaller than the maximum degree $2d$ with respect to t (this might require constraining $x(t)$ to be of a degree lower than $2d$). The fifth application example in Section 7 shows a case of time-varying dynamics.

7. Application examples

This section contains a set of academic examples which show the effectiveness of the approach. The examples, although normalised and involving dimensionless quantities, are based on the real engineering problems which arise in spacecraft control. Problem 3 (including its variants) has been coded in Matlab and used to find solutions. In the context of this work, an *Ad-Hoc* SOS package has been used for the SOS-to-LMI conversion (the same one used in Ben-Talha et al., 2017), but several SOS toolboxes for Matlab are available for users, free of charge (as SOSTOOLS, see Papachristodoulou et al., 2013).

7.1. Second order integrator

As a first example, we test a standard benchmark case for which the solution is known and it can be computed analytically. Namely, we consider the double integrator, or a spacecraft in deep space with linear motion. The dynamic equation is the following scalar one:

$$\ddot{x} = u \tag{17}$$

with $t_f = 100$, $x(0) = 0$, $x(t_f) = 10000$, $\dot{x}(0) = \dot{x}(t_f) = 0$, $u_{max,1} = 10$. The solution of the fuel-optimal problem is the well known bang-off-bang solution, in which u is set to the positive maximum for a certain time, then set to 0, then set to the negative maximum until the spacecraft reaches the destination. We solve Problem 3 for $N = 10$, $d = 2$ (the number of unknowns is 108). The solver (SeDuMi) takes 0.5830 s to find the solution, which is shown in Figure 2. The result approximates exactly what expected, the bang-off-bang solution. The upper bound for the propellant consumption is $J = 228.57$.

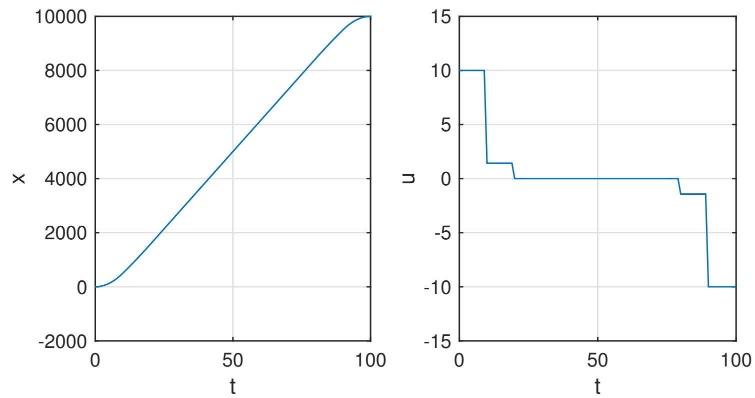

Figure 2: First application example (double integrator, or spacecraft in deep space), $N = 10$.

A more accurate solution is found for $N = 100$ (the number of unknowns is 1098), with a still acceptable solver time of 3.6142 s (see Figure 3). The upper bound for the propellant consumption is now $J = 225.45$. The exact bang-off-bang solution for this problem requires a propellant consumption of 225.4033 (maximum acceleration for 11.270166 s, coasting, then maximum deceleration for the same amount of time). It can be seen that with N increasing the solver tends to arrive at this solution.

7.2. Clohessy-Wiltshire equations, out-of-plane dynamic

We consider the proximity dynamics of a spacecraft in orbit, according to the the Clohessy-Wiltshire equations (Clohessy and Wiltshire, 1960), in the z direction:

$$\ddot{z} = -\omega^2 z + u \tag{18}$$

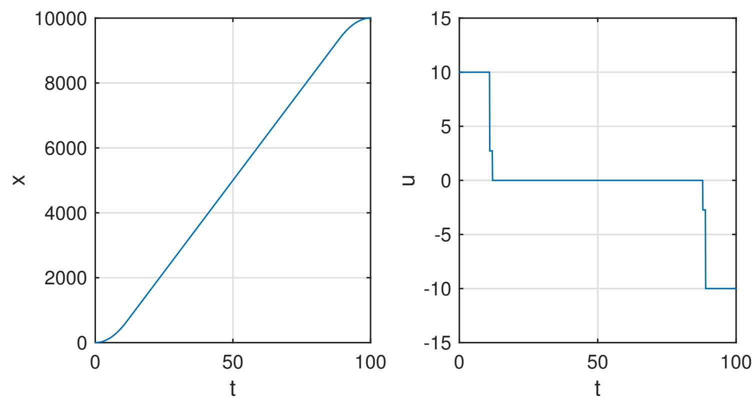

Figure 3: First application example (double integrator, or spacecraft in deep space), $N = 100$.

$\omega = 0.0314$, $t_f = 200$, $z(0) = 0$, $z(t_f) = 200$, $\dot{z}(0) = \dot{z}(t_f) = 0$, $u_{max,1} = 0.5$. For $N = 100$, $d = 2$, the number of unknowns is 1098, the solver time is 9.7057 s and the upper bound for the propellant consumption is $J = 3.1438$. See Figure 4 for the results, again a bang-off-bang optimal strategy is automatically found by the solver.

7.3. Clohessy-Wiltshire equations, in-plane dynamic

We consider now the same Clohessy-Wiltshire model for the x/y direction:

$$\begin{cases} \ddot{x} = 3\omega^2 x + 2\omega\dot{y} + u_x \\ \ddot{y} = -2\omega\dot{x} + u_y \end{cases} \tag{19}$$

with $\omega = 0.0314$, $t_f = 100$, $x_0 = y_0 = 0$, $x_f = 0$, $y_f = 1000$, $v_0 = v_f = 0$. $N = 50$, $d = 2$, $u_{max,1} = u_{max,2} = 100$. This optimisation describes a classical rendez-vous problem in given time. The solver time is 12.2629 s, $J = 3.1438$ see Figure 5 for the results.

7.4. Space station approach

We consider now a more complex example, representing an approach to a berthing box of a space station (Fehse, 2003). We consider, for $\omega = 0.1$, that the chaser has $x(0) = 0$, $y(0) = -1000$ and null initial velocity, and has to reach the target ($x(t_f) = y(t_f) = 0$) at $t_f = 500$ by approaching it from a corridor of $\pm 15°$ in the negative x direction. Additional safety constraints are added, for $t < 300$ the chaser must keep a distance of 200 to the station in the

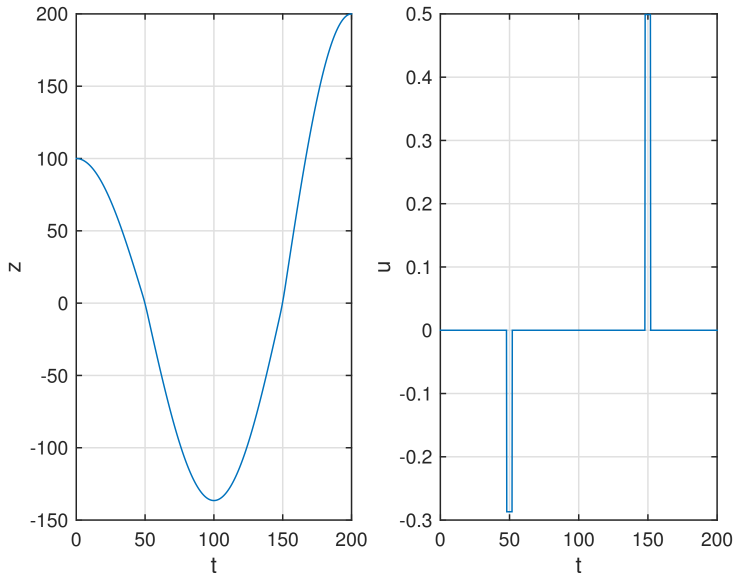

Figure 4: Second application example: Clohessy-Wiltshire equations, out-of-plane dynamics.

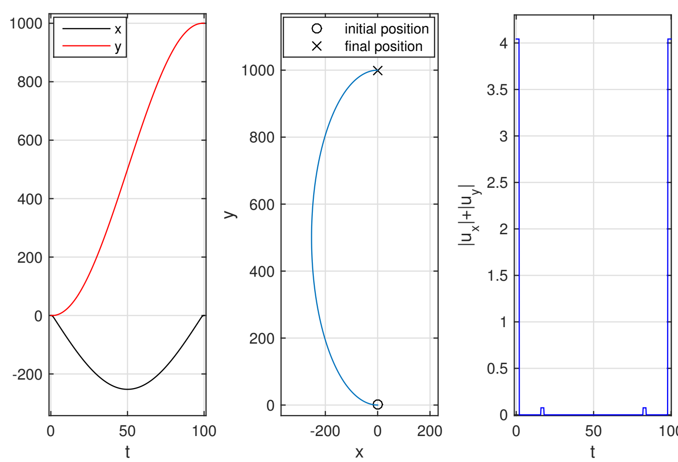

Figure 5: Third application example: Clohessy-Wiltshire equations, in-plane dynamics.

y direction, and for $300 < t < 350$ the chaser must have the same distance to the station in the x direction. The constraints are added as explained in Section 6. For $N = 40$ and $d = 3$, the number of unknowns is 2556, the solver time is 23.5913 s and $J = 15.7191$. The solution is shown in Figure 6 and Figure 7 (final approach). The strategy of the manoeuvre is indeed non trivial as the spacecraft moves away from the station first, in order to be able to approach it from the bottom later on.

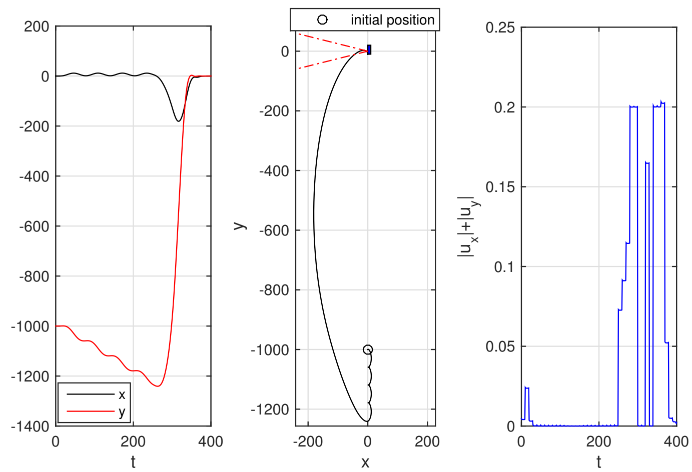

Figure 6: Fourth application example: approach to space station in berthing box direction.

7.5. Example 5: rendez-vous on elliptical orbit

We now consider the in-plane equations of close-motion for an elliptical orbit

$$\begin{cases} \ddot{x} = (2k_0\dot{\theta}^{3/2} + \dot{\theta}^2)x - \ddot{\theta}y - 2\dot{\theta}\dot{y} \\ \ddot{y} = (-k_0\dot{\theta}^{3/2} + \dot{\theta}^2)y + \ddot{\theta}x + \dot{\theta}\dot{x} \end{cases} \tag{20}$$

where x is the displacement with respect to the reference position in the radial direction (positive towards the center), y the displacement in the orthogonal planar direction, θ is the true anomaly and k_0 is a constant (for details see Yamanaka and Ankersen, 2002). These equations are linear time-varying due to the time-dependence of θ; this can be taken into account as explained in Section 6.

As an example, we have chosen an orbit with eccentricity 0.3; the orbit is normalised so that its period is $T = 80$, and the starting time corresponds to the position at the pericenter. We consider a rendez-vous problem with an initial position with a -100 offset in the y direction, and the final position is at the origin for $t_f = 100$. Choosing $N = 30$, the coefficients of all the

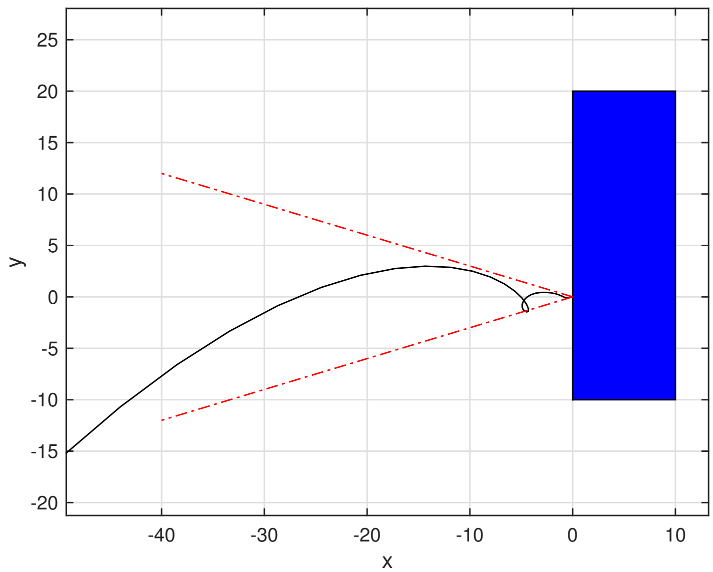

Figure 7: Fourth application example: detail of the final approach.

terms in (20) can be modeled as piecewise-linear function of the time with standard astrodynamics computations.

Taking $d = 3$, the number of unknowns is 1916, the solver time is 11.3731 s and $J = 5.2397$. The results are shown in Figure 8, the solution is non-trivial with three bangs. Notice that the second thrust arc is in proximity of the second passage at the pericenter, which correspond to the maximum efficiency zone for the thrusting.

8. Conclusions

This paper has demonstrated the applicability of the Sum Of Squares (SOS) approach for a set of fuel-optimal control problems under linear dynamics, with potential application to trajectory design for spacecraft in close relative motion. The approach is made possible by a simplification or relaxation of the fuel-optimal problem, but on the other hand the SOS formulation turns the problem into a convex optimisation problem, which can be solved efficiently by existing LMI solvers with guarantee of convergence to the global optimum. Moreover, the proposed framework allows the inclusion of inequality state constraints on the trajectory as well as on the control. The SOS

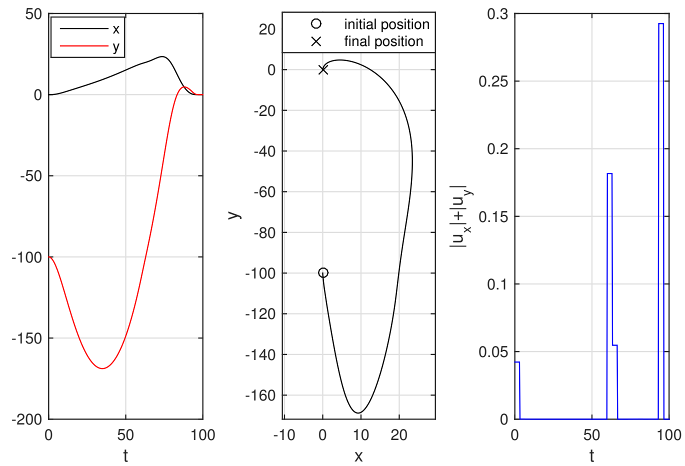

Figure 8: Fifth application example: rendez-vous on an elliptical orbit.

method of this paper can then either be used alone, or to provide an excellent first guess for non-convex numerical optimisation methods. Future research will focus on extending the approach to nonlinear dynamics.

Appendix

We report here, for extra clarity, a list of all the symbols concerning Problem 2 and Problem 3.

Symbol	**Explanation**
t	time
t_0	starting time
t_f	final time
n	number of dimensions
$x(t)$	position vector, $x(t) = [x_1(t), ..., x_n(t)]^\top$
x_0	starting position vector
x_f	final position vector
$\dot{x}(t)$	velocity vector, $\dot{x}(t) = [\dot{x}_1(t), ..., \dot{x}_n(t)]^\top$
v_0	starting velocity vector
v_f	final velocity vector
$f(x)$	dynamics function, $f(x) = [f_1(x), ..., f_n(x)]^\top$
$u(t)$	control input vector, $u(t) = [u_1(t), ..., u_n(t)]^\top$

u_{max}	bounds on control vector, $u_{max} = [u_{1,max}(t), ..., u_{n,max}(t)]^\top$
$x^*(t)$	optimal trajectory
$u^*(t)$	optimal control action
N	number of time intervals
$p_{i,j}(t)$	polynomial approximating $x_i^*(t)$ in the j^{th} time interval
d	half-degree of polynomials $p_{i,j}(t)$
$\gamma_{i,j}$	bound on $u_i(t)$ on the j^{th} time interval
$\xi(t)$	vector of monomials of t, $\xi(t) = [1, t, ..., t^d]^\top$
$P_{i,j}$	matrix representation of $p_{i,j}(t)$, i.e. $p_{i,j}(t) = \xi(t)^\top P_{i,j} \xi(t)$
$\Omega_{i,j,k}$	positive multiplier (equivalent to the "$\tau(t)$") for S-procedure
$\mu_{i,j,k,l}$	slack term taking into account non-uniqueness of $P_{i,j}$
$P'_{i,j}$	matrix representation of $\dot{p}_{i,j}(t)$, i.e. $\dot{p}_{i,j}(t) = \xi(t)^\top P'_{i,j} \xi(t)$
$U_{i,j}$	matrix representation of u_i in the j^{th} time interval, i.e. $u_i(t) = \xi(t)^\top U_{i,j} \xi(t)$ in such interval
$\mathbb{I}$	polynomial matrix representation of 1, i.e. $\xi(t)^\top \mathbb{I} \xi(t) = 1$
$\Xi_{i,j,k}$	S-procedure terms for limiting inequalities to j^{th} time interval, i.e. $\xi(t)^\top \Xi_{i,j,k} \xi(t) = \xi(t)^\top \Omega_{i,j,k} \xi(t)^\top (1 - (2tN/t_f - (2j-1))^2)$
ι	number of slack matrices for non-uniqueness of $P_{i,j}$
Q_l	slack matrices for non-uniqueness of $P_{i,j}$, i.e. $\xi(t)^\top Q_l \xi(t) = 0$ (for $l = 1, ..., \iota$)
Δk	positive multiplier (equivalent to the "$\tau(t)$") for S-procedure
$\eta_{k,l}$	slack term taking into account non-uniqueness of $P_{i,j}$
Π_k	S-procedure terms for limiting inequalities to j^{th} time interval, i.e. $\xi(t)^\top \Pi_k \xi(t) = \xi(t)^\top \Delta_k \xi(t)^\top (1 - (2tN/t_f - (2j-1))^2)$.

References

Ben-Talha, H., Massioni, P., and Scorletti, G. (2017). Robust simulation of continuous-time systems with rational dynamics. *Int. J. Robust Nonlin.*, 27:3097–3108. `http://dx.doi.org/10.1002/rnc.3728`.

Betts, J. (1998). Survey of numerical methods for trajectory optimization. *J. Guid. Control. Dyn.*, 21(2):193–207. `http://dx.doi.org/10.2514/2.4231`.

Bolle, A. and Circi, C. (2012). A hybrid, self-adjusting search algorithm for optimal space trajectory design. *Adv. Space Res.*, 50(4):471–488. `http://dx.doi.org/10.1016/j.asr.2012.04.026`.

Boyd, S., El Ghaoui, L., Feron, E., and Balakrishnan, V. (1994). *Linear matrix inequalities in system and control theory*, volume 15. SIAM.

Boyd, S. and Vandenberghe, L. (2004). *Convex optimization.* Cambridge university press.

Casalino, L., Colasurdo, G., and Pastrone, D. (1999). Optimal low-thrust escape trajectories using gravity assist. *J. Guid. Control. Dyn.*, 22(5):637–642. `http://dx.doi.org/10.2514/2.4451`.

Claeys, M., Arzelier, D., Henrion, D., and Lasserre, J.-B. (2014). Measures and LMIs for impulsive nonlinear optimal control. *IEEE Trans. Autom. Control*, 59(5):1374–1379. `http://dx.doi.org/10.1109/TAC.2013.2292735`.

Clohessy, W. and Wiltshire, R. (1960). Terminal guidance system for satellite rendezvous. *J. Aerosp. Sci.*, 27(9):653–658. `http://dx.doi.org/10.2514/8.8704`.

Fehse, W. (2003). *Automated rendezvous and docking of spacecraft*, volume 16. Cambridge university press.

Gahinet, P., Nemirovskii, A., Laub, A., and Chilali, M. (1994). The LMI control toolbox. In *33rd IEEE Conf. Decis. Control*, volume 3, pages 2038–2041. IEEE. `http://dx.doi.org/0.1109/CDC.1994.411440`.

Genin, Y., Hachez, Y., Nesterov, Y., and Van Dooren, P. (2000). Convex optimization over positive polynomials and filter design. In *Proc. 2000 UKACC Int. Conf. Control*, page 41.

Kirk, D. (2012). *Optimal control theory: an introduction.* Courier Corporation.

Lasserre, J. (2009). *Moments, positive polynomials and their applications*, volume 1. World Scientific.

Li, J. (2016). Fuel-optimal low-thrust formation reconfiguration via Radau pseudospectral method. *Adv. Space Res.*, 58(1):1–16. `http://dx.doi.org/10.1016/j.asr.2016.04.005`.

Löfberg, J. (2004). YALMIP: a toolbox for modeling and optimization in MATLAB. In *Proc. of the CACSD Conference*, pages 284–289, Taipei, Taiwan. http://dx.doi.org/10.1109/CACSD.2004.1393890.

Magnani, A. and Boyd, S. (2009). Convex piecewise-linear fitting. *Optim. Eng.*, 10(1):1–17. http://dx.doi.org/10.1007/s11081-008-9045-3.

Massari, M. and Bernelli-Zazzera, F. (2009). Optimization of low-thrust reconfiguration maneuvers for spacecraft flying in formation. *J. Guid. Control. Dyn.*, 32(5):1629–1638. http://dx.doi.org/10.2514/1.37335.

Massari, M., Bernelli-Zazzera, F., and Vasile, M. (2003). Trajectory optimization for a mission to NEOs, using low-thrust propulsion and gravity assist. *Adv. Astronaut. Sci.*, 114(SUPPL.):303–316.

Papachristodoulou, A., Anderson, J., Valmorbida, G., Prajna, S., Seiler, P., and Parrilo, P. (2013). *SOSTOOLS: Sum of squares optimization toolbox for MATLAB*. http://arxiv.org/abs/1310.4716. Available from http://www.eng.ox.ac.uk/control/sostools, http://www.cds.caltech.edu/sostools and http://www.mit.edu/~parrilo/sostools.

Parrilo, P. (2003). Semidefinite programming relaxations for semialgebraic problems. *Math. program.*, 96(2):293–320. http://dx.doi.org/10.1007/s10107-003-0387-5.

Qi, Y. and Jia, Y. (2012). Constant thrust fuel-optimal control for spacecraft rendezvous. *Adv. Space Res.*, 49(7):1140–1150. http://dx.doi.org/10.1016/j.asr.2012.01.002.

Scharf, D., Hadaegh, F., and Ploen, S. (2003). A survey of spacecraft formation flying guidance and control (part i): Guidance. In *Proceedings of the American Control Conference*, pages 1733–1739, Denver, Colorado, USA. http://dx.doi.org/10.1109/ACC.2003.1239845.

Sturm, J. (1999). Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optim. Method Softw.*, 11–12:625–653. http://dx.doi.org/10.1080/10556789908805766.

Yamanaka, K. and Ankersen, F. (2002). New state transition matrix for relative motion on an arbitrary elliptical orbit. *J. Guid. Control. Dyn.*, 25(1):60–66. `http://dx.doi.org/10.2514/2.4875`.

Zhang, C., Topputo, F., Bernelli-Zazzera, F., and Zhao, Y.-S. (2015). Low-thrust minimum-fuel optimization in the circular restricted three-body problem. *J. Guid. Control. Dyn.*, 38(8):1501–1510. `http://dx.doi.org/10.2514/1.G001080`.