POLITECNICO DI MILANO
Facoltà di Ingegneria Industriale
Master of Science in
Space Engineering

A COMBINED DYNAMIC-ALGEBRAIC METHOD TO SOLVE NONLINEAR OPTIMAL CONTROL PROBLEMS WITH APPLICATIONS

Relatore: **Prof. Franco Bernelli Zazzera**
Correlatore: **Ing. Francesco Topputo**

Thesis by:
MARTINO MARIA MIANI
Matr. n. 765793

Academic Year 2012-2013

Ringraziamenti

Vorrei innanzitutto ringraziare l'Ing. Francesco Topputo, per la pazienza e la disponibilità con cui mi ha seguito durante il periodo di tesi: posso dire di aver imparato davvero tanto grazie a questo lavoro. Inoltre ringrazio la mia famiglia per la serenità con cui mi è stato concesso di lavorare, i miei amici per l'incoraggiamento a fare del mio meglio e soprattutto Giulia, senza la quale questo periodo di tesi sarebbe stato sicuramente più faticoso e meno ricco.

Abstract

Space-related control problems pose a challenging task due to the strict requirements in terms of precision and efficiency and to the highly nonlinear dynamics involved. The classical methods for solving nonlinear optimal control problems require a good starting guess in order to converge, while recently have been developed approximate methods that are able to find a solution without such a need. This thesis is focused on analysing and improving the Approximating Sequence of Riccati Equation (ASRE) method, that relies on solving a sequence of time-varying Linear Quadratic Regulator (LQR) problems based on a pseudo-factorization of the dynamics: the multiple-factorizations feature is exploited in order to make the most out of the approximate method, that is then refined using a classical Two-Point Boundary Value Problem (TPBVP) solver; the overall procedure is then applied to some typical space-related optimal control problems.

Keywords nonlinear optimal control, space applications, astrodynamics

Sommario

I problemi di controllo relativi all'ambito spaziale risultano essere particolarmente difficili da risolvere a causa dei requisiti stringenti in termini di precisione ed efficienza, oltre che alle dinamiche fortemente non lineari coinvolte. I metodi classici per risolvere i problemi di controllo ottimo non lineare richiedono una buona soluzione di primo tentativo per poter convergere, mentre metodi approssimati recentemente sviluppati non hanno questo bisogno. Questa tesi si concentra sull'analisi e il miglioramento del metodo Approximating Sequence of Riccati Equation (ASRE), che si basa sulla soluzione di una sequenza di problemi Linear Quadratic Regulator (LQR) tempo-varianti, ottenuti fattorizzando la dinamica: si sfrutta la possibilità di ricavare molteplici fattorizzazioni per ottimizzare il metodo approssimato, che viene poi rifinito con un classico solutore Two-Point Boundary Value Problem (TPBVP); la procedura completa è stata applicata per risolvere alcuni tipici problemi di controllo di ambito spaziale.

Parole chiave controllo ottimo non lineare, applicazioni spaziali, astrodinamica

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The optimal control problem

Optimal control deals with the problem of finding a set of functions that are able to control a process over a given time interval, such that an associated cost is minimized; this cost is, in general, a function of different variables:

states: those variables whose evolution is governed by a system of differential equations, representing the dynamics of the process itself, that depend on the control input;

controls: the input variables over which it is possible to optimize the whole process;

final states: some of the states that are not eventually specified at final time through which it is possible to modify the cost function.

The integration of the differential system requires some boundary conditions: all of the initial conditions must be specified, while the final conditions can be either prescribed or to be selected freely by the optimization process. Additionally, it could be that the final time is not specified, so that the optimization process is also in charge of selecting the optimal final time; furthermore, the control variables could be limited, introducing in the problem other constraints to be dealt with.

In general, the optimal control can be derived by applying *Pontryagin's maximum principle* (see [3]), that provides a necessary condition, or by solving the *Hamilton-Jacobi-Bellman equation*, that represents a sufficient condition descending from the *Hamilton-Jacobi* equation, which, in turn, is equivalent to the *Euler-Lagrange* equation. Actually, the application of these two principles typically requires to solve problems whose solutions are not trivial or even not existent, so that only for particular cases, e.g, linear optimal control problems, it is always possible to find a solution. For the generic nonlinear optimal control problem it is often necessary to apply *approximate methods*, that will be extensively presented in this thesis.

1.2 Why optimal control?

It must be said that control theory spans a multitude of possible approaches to the issue of being able to control a process so that it will behave as requested. Among the various possibilities (pole-placement, robust control, adaptive control, intelligent control, etc.), optimal control stands with the extreme simplicity of its statement: given a dynamical system and a cost function, the optimal control is such that the cost function is minimized, satisfying some boundary conditions. From the engineering point of view, this is probably the most suitable way of expressing a control problem, with an immediate feeling of what is the target of the controller, without being concerned with the direct choice of parameters that can be difficult to relate with the physics involved; the only requisites are a model of the dynamics and the definition of both cost function and boundary conditions.

Going into the specific field of space applications, optimal control becomes not only a good choice, but even a necessary one: the strict limitations in terms of mass and power are such that, if no optimization occurs, it may be impossible to accomplish the target of the mission. This critical aspect has been a strong motivation for developing the optimal control theory, that is now the main strategy adopted when dealing with space applications.

1.3 Aerospace applications

Optimal control has two main fields of applications for what concerns aerospace related issues:

guidance: deals with the determination of the nominal optimal trajectory to be followed;

control: deals with the adjustments and corrections needed for counteracting possible off-nominal conditions.

Optimal guidance is typically computed off-line, resorting to a complete model of the dynamics involved that is integrated over a *finite* time interval, while optimal control, in a strict sense, is done on-line employing the sensors aboard, solving an *infinite-horizon* problem.

Optimal guidance can be used to determine the trajectory to be followed during a launch (see figure 1.1) or during an orbital transfer, while optimal control provides a way of stabilizing the attitude of a spacecraft subject to disturbing torques of various kind or of counteracting the natural drift from the nominal location of the same spacecraft (see figure 1.2).

Figure 1.1: VEGA's first launch in February 2012

Figure 1.2: The International Space Station orbiting

Chapter 2

The nonlinear optimal control problem

2.1 Statement of the problem

The aim of this thesis is focused on finding a solution to the following problem. Given a set of n first-order, differential equations

$$\dot{\mathbf{x}} = \mathbf{f}\left[\mathbf{x}(t), \mathbf{u}(t), t\right], \tag{2.1}$$

it is required to find the piecewise-continuous, admissible set of m control functions $\mathbf{u}(t)$ defined over the given interval $[t_i, t_f]$, such that the scalar cost function

$$J = \varphi\left[\mathbf{x}(t_f), t_f\right] + \int_{t_i}^{t_f} L\left[\mathbf{x}(t), \mathbf{u}(t), t\right] \mathrm{d}t, \tag{2.2}$$

is minimized, satisfying $n + q$ boundary conditions of the form

$$\mathbf{x}(t_i) = \mathbf{x}_i \quad n \text{ b.c.'s}, \tag{2.3}$$

$$\mathbf{x}(t_f) = \mathbf{x}_f \quad q \text{ b.c.'s}, \tag{2.4}$$

where $0 \leq q \leq n$, and L is the *Lagrangian* of the system.

2.2 Solution by Euler-Lagrange equations

A straightforward way to solve the problem consists in adjoining the differential equations (2.1) to the cost function (2.2) with multiplier functions $\boldsymbol{\lambda}(t)$:

$$J = \varphi\left[\mathbf{x}(t_f), t_f\right] + \int_{t_i}^{t_f} \left[L\left[\mathbf{x}(t), \mathbf{u}(t), t\right] + \boldsymbol{\lambda}^T(t)\left\{\mathbf{f}\left[\mathbf{x}(t), \mathbf{u}(t), t\right] - \dot{\mathbf{x}}\right\}\right] \mathrm{d}t, \tag{2.5}$$

and also defining a scalar function H (the *Hamiltonian*) as follows:

$$H\left[\mathbf{x}(t),\mathbf{u}(t),t\right]=L\left[\mathbf{x}(t),\mathbf{u}(t),t\right]+\boldsymbol{\lambda}^{T}(t)\mathbf{f}\left[\mathbf{x}(t),\mathbf{u}(t),t\right]. \tag{2.6}$$

The fixed-time first-order variation of equation (2.5) is obtained by differentiation

$$\begin{aligned}\delta J=\left[\frac{\partial\varphi}{\partial\mathbf{x}}\delta\mathbf{x}\right]_{t_f}+\int_{t_i}^{t_f}\Bigg[&\frac{\partial L}{\partial\mathbf{x}}\delta\mathbf{x}+\frac{\partial L}{\partial\mathbf{u}}\delta\mathbf{u}+\delta\boldsymbol{\lambda}^{T}\left(\mathbf{f}-\dot{\mathbf{x}}\right)\\&+\boldsymbol{\lambda}^{T}\frac{\partial\mathbf{f}}{\partial\mathbf{x}}\delta\mathbf{x}+\boldsymbol{\lambda}^{T}\frac{\partial\mathbf{f}}{\partial\mathbf{u}}\delta\mathbf{u}-\boldsymbol{\lambda}^{T}\delta\dot{\mathbf{x}}\Bigg]\,\mathrm{d}t,\end{aligned} \tag{2.7}$$

and the last term of the right hand side can be integrated by parts to yield:

$$\begin{aligned}\delta J=\left[\frac{\partial\varphi}{\partial\mathbf{x}}\delta\mathbf{x}\right]_{t_f}-\left[\boldsymbol{\lambda}^{T}\delta\mathbf{x}\right]_{t_i}^{t_f}+\int_{t_i}^{t_f}\Bigg[&\frac{\partial L}{\partial\mathbf{x}}\delta\mathbf{x}+\frac{\partial L}{\partial\mathbf{u}}\delta\mathbf{u}+\delta\boldsymbol{\lambda}^{T}\left(\mathbf{f}-\dot{\mathbf{x}}\right)\\&+\boldsymbol{\lambda}^{T}\frac{\partial\mathbf{f}}{\partial\mathbf{x}}\delta\mathbf{x}+\boldsymbol{\lambda}^{T}\frac{\partial\mathbf{f}}{\partial\mathbf{u}}\delta\mathbf{u}+\dot{\boldsymbol{\lambda}}^{T}\delta\mathbf{x}\Bigg]\,\mathrm{d}t.\end{aligned} \tag{2.8}$$

Since the initial conditions are fixed, and no variation can thus occur, and also remembering equation (2.6), expression (2.8) can be rewritten as

$$\begin{aligned}\delta J=\left(\frac{\partial\varphi}{\partial\mathbf{x}_f}-\boldsymbol{\lambda}_f^{T}\right)\delta\mathbf{x}_f+\int_{t_i}^{t_f}\Bigg[&\left(\frac{\partial H}{\partial\mathbf{x}}+\dot{\boldsymbol{\lambda}}^{T}\right)\delta\mathbf{x}+\frac{\partial H}{\partial\mathbf{u}}\delta\mathbf{u}\\&+\delta\boldsymbol{\lambda}^{T}\left(\frac{\partial H}{\partial\boldsymbol{\lambda}}-\dot{\mathbf{x}}\right)\Bigg]\,\mathrm{d}t.\end{aligned} \tag{2.9}$$

In order to locate the minimum of the cost function, its first variation must be null for any arbitrary variation of $\mathbf{x}$, $\mathbf{u}$, $\boldsymbol{\lambda}$ and for any arbitrary variation of those final states that has not been fixed among the boundary conditions; this means that, to find the optimal control, the following differential-algebraic equations, called *Euler-Lagrange equations*, must be solved:

$$\dot{\mathbf{x}}=\frac{\partial H}{\partial\boldsymbol{\lambda}}=\mathbf{f}\left[\mathbf{x}(t),\mathbf{u}(t),t\right], \tag{2.10}$$

$$\dot{\boldsymbol{\lambda}}^{T}=-\frac{\partial H}{\partial\mathbf{x}}=-\frac{\partial L}{\partial\mathbf{x}}-\boldsymbol{\lambda}^{T}\frac{\partial\mathbf{f}}{\partial\mathbf{x}}, \tag{2.11}$$

$$\frac{\partial H}{\partial\mathbf{u}}=\frac{\partial L}{\partial\mathbf{u}}+\boldsymbol{\lambda}^{T}\frac{\partial\mathbf{f}}{\partial\mathbf{u}}=0, \tag{2.12}$$

enforcing the $n+q$ boundary conditions on the state

$$\mathbf{x}(t_i)=\mathbf{x}_i \quad n \text{ b.c.'s}, \tag{2.13}$$

$$\mathbf{x}(t_f)=\mathbf{x}_f \quad q \text{ b.c.'s}, \tag{2.14}$$

together with the $n-q$ boundary conditions on the co-state descending from the *transversality condition*

$$\boldsymbol{\lambda}_f^T = \frac{\partial \varphi}{\partial \mathbf{x}_f} \quad n-q \text{ b.c.'s}, \tag{2.15}$$

for the general case of having prescribed q states at final time. Solving equation (2.12) for the control $\mathbf{u}(t)$, and then substituting inside equations (2.10) and (2.11) leads to an *Hamiltonian system* with variables $\mathbf{x}(t)$ and $\boldsymbol{\lambda}(t)$:

$$\begin{Bmatrix} \dot{\mathbf{x}} \\ \dot{\boldsymbol{\lambda}} \end{Bmatrix} = \begin{Bmatrix} \dfrac{\partial H}{\partial \boldsymbol{\lambda}}(\mathbf{x}, \boldsymbol{\lambda}) \\ -\dfrac{\partial H}{\partial \mathbf{x}}(\mathbf{x}, \boldsymbol{\lambda}) \end{Bmatrix}, \tag{2.16}$$

with the appropriate boundary conditions for $\mathbf{x}$ and $\boldsymbol{\lambda}$. It must be remembered that those above are necessary conditions for minimizing the cost function, since also a maximum of the cost function would satisfy equations (2.16). Of course, the problem could be more complex: final time could be not assigned, and so it would have to be treated as a variable to be optimized according to the problem; the states at final time could be constrained implicitly, thus yielding additional equations to be adjoined to the cost function with additional Lagrange multipliers; the control input could be bounded within given limits, so that the problem should take into account possible *path constraints*; these and other aspects are dealt with in [1].

2.3 An example: maximum velocity transfer to a rectilinear path

The following example has been formulated and solved in [1]: consider a particle of mass m acted upon by a thrust force of constant magnitude ma. Assuming planar motion with velocity $[u, v]$, it is required to find the optimal time-history of the thrust angle β such that after a time T the particle is brought to an horizontal path at an height h having maximized the horizontal component u of the velocity, starting from the origin of the reference system of coordinates $[x, y]$ at time $t = 0$. The equations of motion for the system are

$$\begin{aligned} \dot{x} &= u, \\ \dot{y} &= v, \\ \dot{u} &= a\cos\beta, \\ \dot{v} &= a\sin\beta, \end{aligned}$$

while the cost function, to be maximized, is

$$J = \varphi(x_f) = u(T),$$

since it is required to extremalize a function of the final conditions only, and so the Hamiltonian H is simply

$$H = \boldsymbol{\lambda}^T \mathbf{f} = \begin{Bmatrix} \lambda_x & \lambda_y & \lambda_u & \lambda_v \end{Bmatrix} \begin{Bmatrix} u \\ v \\ a\cos\beta \\ a\sin\beta \end{Bmatrix}$$
$$= \lambda_x u + \lambda_y v + \lambda_u a\cos\beta + \lambda_v a\sin\beta.$$

At initial time the particle is located at the origin of the reference system and it is not moving, while the only prescribed final conditions are that the particle must be at an height h with no vertical component of velocity, so that the boundary conditions are

$$\left.\begin{array}{l} x = 0, \\ u = 0, \\ y = 0, \\ v = 0, \end{array}\right\}_{t=0} \qquad \left.\begin{array}{l} y = h, \\ v = 0. \end{array}\right\}_{t=T}$$

Enforcing *Euler-Lagrange equations* yields the dynamics of the multipliers,

$$\dot{\boldsymbol{\lambda}} = -\left(\frac{\partial H}{\partial \mathbf{x}}\right)^T = -\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)^T \boldsymbol{\lambda}$$
$$\Downarrow$$
$$\dot{\lambda}_x = 0,$$
$$\dot{\lambda}_y = 0,$$
$$\dot{\lambda}_u = -\lambda_x,$$
$$\dot{\lambda}_v = -\lambda_y,$$

the optimal control law,

$$\frac{\partial H}{\partial \beta} = -\lambda_u a\sin\beta + \lambda_v \cos\beta = 0$$
$$\Downarrow$$
$$\tan\beta = \frac{\lambda_v}{\lambda_u}$$
$$\Downarrow$$
$$\beta = \tan^{-1}\left(\frac{\lambda_v}{\lambda_u}\right),$$

and the remaining boundary conditions

$$\lambda_x(T) = \frac{\partial \varphi}{\partial x_f} = 0,$$
$$\lambda_u(T) = \frac{\partial \varphi}{\partial u_f} = 1.$$

Now the problem is reduced to solving the differential equations of states and multipliers

$$\begin{aligned}
\dot{x} &= u, \\
\dot{y} &= v, \\
\dot{u} &= a\cos\left[\tan^{-1}\left(\frac{\lambda_v}{\lambda_u}\right)\right], \\
\dot{v} &= a\sin\left[\tan^{-1}\left(\frac{\lambda_v}{\lambda_u}\right)\right], \\
\dot{\lambda}_x &= 0, \\
\dot{\lambda}_y &= 0, \\
\dot{\lambda}_u &= -\lambda_x, \\
\dot{\lambda}_v &= -\lambda_y,
\end{aligned}$$

satisfying the boundary conditions

$$\left.\begin{array}{l} x = 0, \\ y = 0, \\ u = 0, \\ v = 0, \end{array}\right\}_{t=0} \qquad \left.\begin{array}{r} y = h, \\ v = 0, \\ \lambda_x = 0, \\ \lambda_u = 1. \end{array}\right\}_{t=T}$$

The dynamics of the multipliers are easy to integrate

$$\begin{aligned}
\lambda_x &= c_1, \\
\lambda_y &= c_2, \\
\lambda_u &= -c_1 t + c_3, \\
\lambda_v &= -c_2 t + c_4,
\end{aligned}$$

and, applying the boundary conditions,

$$\begin{array}{l} \lambda_x = 0, \\ \lambda_u = 1, \end{array} \Rightarrow \begin{array}{l} c_1 = 0, \\ c_3 = 1, \end{array}$$

the control law can be rewritten as

$$\tan\beta = -c_2 t + c_4.$$

Simple manipulations allow to cast the previous expression in a different form:

$$\begin{aligned}
\tan\beta &= -c_2 t + c_4 \\
&= -c_2 T + c_4 + c_2 T - c_2 t \\
&= \lambda_v(T) + \lambda_y(T)\, T - \lambda_y(T)\, t \\
&= \tan\beta_0 - ct,
\end{aligned}$$

where $\tan\beta_0 = \lambda_v(T) + \lambda_y(T)\,T$ and $c = \lambda_y(T)$. Then, the system of equations can be integrated, using β as the independent variable instead of t, thus obtaining

$$\begin{aligned} x &= \frac{a}{c^2}\left(\sec\beta_0 - \sec\beta - \tan\beta\log\frac{\tan\beta_0+\sec\beta_0}{\tan\beta+\sec\beta}\right), \\ y &= \frac{a}{2c^2}\Bigg[(\tan\beta_0-\tan\beta)\sec\beta_0 - (\sec\beta_0-\sec\beta)\tan\beta \\ &\qquad - \log\frac{\tan\beta_0+\sec\beta_0}{\tan\beta+\sec\beta}\Bigg], \\ u &= \frac{a}{c}\log\frac{\tan\beta_0+\sec\beta_0}{\tan\beta+\sec\beta}, \\ v &= \frac{a}{c}\left(\sec\beta_0-\sec\beta\right). \end{aligned}$$

The constants β_0 and c are determined from the two final boundary conditions $v = 0$ and $y = h$, even though those relations turn out to be implicit:

$$\begin{aligned} \frac{4h}{aT^2} &= \frac{1}{\sin\beta_0} - \frac{1}{2\tan^2\beta_0}\log\frac{\sec\beta_0+\tan\beta_0}{\sec\beta_0-\tan\beta_0}, \\ c &= \frac{2\tan\beta_0}{T} \Rightarrow \tan\beta = \tan\beta_0\left(1-\frac{2t}{T}\right). \end{aligned}$$

In figure 2.1a it is shown the optimal trajectory for $h = 10$, $T = 10$ and $a = 1$, while in figures 2.1b, 2.1c and 2.1d are shown the time-profiles of the thrust angle and of the velocity's components, respectively.

Even though the problem solved above stems from very simple dynamics with just four equations that are even linear in the states, it is clear that the analytical solution is not an easy task to deal with. In general, for more complex problems, involving many more states and with increased nonlinearities, it is likely that analytical solutions do not exist; this leads to the need of developing numerical procedures to solve the optimal control problem. Those procedures, actually, only work with the necessary conditions, converging toward *local* minima; there is no way of knowing if the solution found is the *global* minimum.

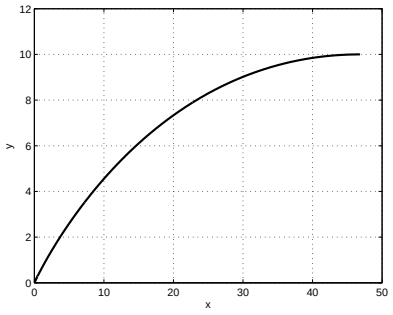
2.4 The nonlinear programming problem

The general Nonlinear Programming (NLP) problem can be stated as follows: find $\mathbf{x} \in \mathbb{R}^n$ to minimize the scalar objective function $F(\mathbf{x})$ subject to the m constraints

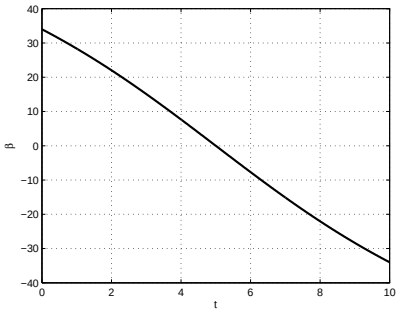
$$\mathbf{c}_L \leq \mathbf{c}(\mathbf{x}) \leq \mathbf{c}_U \tag{2.17}$$

and n simple bounds

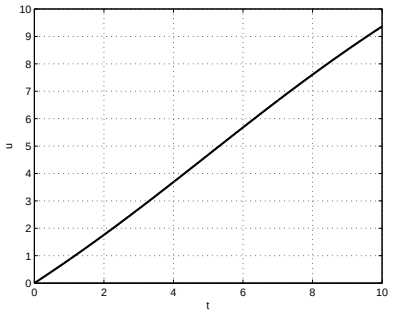
$$\mathbf{x}_L \leq \mathbf{x} \leq \mathbf{x}_U. \tag{2.18}$$



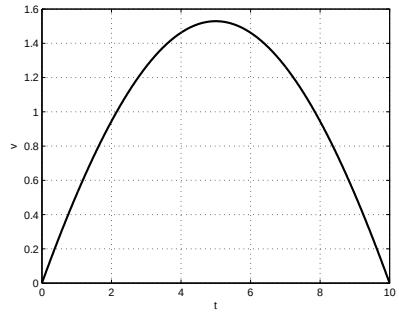
(a) Optimal trajectory



(b) Thrust-angle β time-profile



(c) Horizontal velocity time-profile



(d) Vertical velocity time-profile

Figure 2.1: Solution of the maximum-velocity transfer problem

This problem is a constrained minimization, with equality and inequality constraints; first, it is interesting to deal with the simpler unbounded, equality-constrained minimization, i.e., to impose $\mathbf{c}_L = \mathbf{c}_U = 0$, $\mathbf{x}_L = -\infty$ and $\mathbf{x}_U = +\infty$: the classical approach is to define the Lagrangian L

$$L(\mathbf{x}, \boldsymbol{\lambda}) = F(\mathbf{x}) - \boldsymbol{\lambda}^T \mathbf{c}(\mathbf{x}), \tag{2.19}$$

introducing the vector $\boldsymbol{\lambda} \in \mathbb{R}^m$ of *Lagrange multipliers*. Necessary conditions for the point $(\mathbf{x}^*, \boldsymbol{\lambda}^*)$ to be an optimum are

$$\nabla_x L(\mathbf{x}^*, \boldsymbol{\lambda}^*) = 0 \tag{2.20}$$

$$\nabla_\lambda L(\mathbf{x}^*, \boldsymbol{\lambda}^*) = 0. \tag{2.21}$$

The gradient of L with respect to $\mathbf{x}$ is

$$\nabla_x L = \mathbf{g} - \mathbf{G}^T \boldsymbol{\lambda} = \nabla F - \sum_{i=1}^{m} \lambda_i \nabla c_i \tag{2.22}$$

and the gradient of L with respect to $\boldsymbol{\lambda}$ is

$$\nabla_\lambda L = -\mathbf{c}(\mathbf{x}). \tag{2.23}$$

Typically those equations are solved iteratively, following Newton's method; to do this, first the expressions (2.22) and (2.23) are expanded as first-order Taylor series about a certain point $(\mathbf{x}, \boldsymbol{\lambda})$, so that the gradients become

$$\nabla_x L = \mathbf{g} - \mathbf{G}^T \boldsymbol{\lambda} + \mathbf{H}_L (\overline{\mathbf{x}} - \mathbf{x}) - \mathbf{G}^T \left(\overline{\boldsymbol{\lambda}} - \boldsymbol{\lambda}\right) \tag{2.24}$$

$$\nabla_\lambda L = -\mathbf{c} - \mathbf{G}^T (\overline{\mathbf{x}} - \mathbf{x}), \tag{2.25}$$

where $\mathbf{H}_L$ is the *Hessian* of the objective function. Next, these linear approximations are solved for $\overline{\mathbf{x}}$ and $\overline{\boldsymbol{\lambda}}$, assuming that the gradients are null; this leads to linearly-approximated solutions for the $\overline{\mathbf{x}}$ and $\overline{\boldsymbol{\lambda}}$ vectors, and the procedure is iterated until the difference between two consecutive iterations is less than a prescribed value. By defining a search-direction $\mathbf{p}$ for a step $\overline{\mathbf{x}} = \mathbf{x} + \mathbf{p}$, the system to be solved becomes

$$\begin{bmatrix} \mathbf{H}_L & \mathbf{G}^T \\ \mathbf{G} & 0 \end{bmatrix} \begin{Bmatrix} -\mathbf{p} \\ \overline{\boldsymbol{\lambda}} \end{Bmatrix} = \begin{Bmatrix} \mathbf{g} \\ \mathbf{c} \end{Bmatrix}, \tag{2.26}$$

which is known as the *Karush-Kuhn-Tucker* system. In this way the solution is based on a linear approximation of the constraints and a linear approximation of the gradients, equivalent to a quadratic approximation of the quantity to be optimized.

Inequality constraints are treated introducing a distinction between an *active set* $\mathcal{A}$ and an *inactive set* $\mathcal{A}'$: given a point $\mathbf{x}$, those constraints that are satisfied as equalities belong to $\mathcal{A}$, while those that are strictly satisfied

belong to $\mathcal{A}'$. Of course, active constraints can be treated as equalities, but it is still necessary to find out which constraints are active an which are not, employing an *active set strategy*. To do this, it must be checked the sign of the corresponding Lagrange multipliers: a negative Lagrange multiplier identifies a constraint that is not active, and thus that constraint can be left out of the Lagrangian.

Analogously with the optimal control problem, also for nonlinear programming finding closed-form solutions is not a trivial task; again, numerical procedures must be employed to solve most problems, as those related to space applications.

Chapter 3

Approximate methods

3.1 Why approximate methods?

As seen in the previous chapter, the nonlinear, optimal control problem is reduced to solving a so called Two-Point Boundary Value Problem (TPBVP), i.e., a system of differential equations for the state and the co-state, whose boundary conditions are given both at the initial and final time. This kind of problem is hard to solve analytically, because the differential equations are often non-trivial and also because the system can be very large. Of course, have been developed numerical procedures that are able to solve TPBVPs using finite difference schemes, but the downside of these algorithms is that they require a good starting guess in order to converge quickly to the optimal solution (*shooting technique*); differently, a starting guess that is not "near" the optimal solution will cause the algorithms to require many iterations or even not to converge to an optimal solution. Furthermore, TPBVPs related to optimal control are such that the user must not only provide a starting guess for the state (which can still prove to be a difficult task), but a starting guess for the co-state, which typically lacks of physical meaning, is also needed.

The above discussion leads to think that it would be convenient to define a procedure that has little or no need at all of a starting guess, even at the cost of having a sub-optimal solution; this is exactly the philosophy behind the *approximate methods* that have been developed over the past decades: the solution of the TPBVP is avoided by means of solving simpler problems iteratively.

Examples of approximate methods are: *direct transcription* or *direct optimization*, introduced in [24, 25, 26], which is based on representing states and/or control variables with piecewise-continuous polynomials and then using NLP to solve the control problem, instead of solving a TPBVP; another way is to employ *generating functions* [27], obtained by applying a canonical transformation to the Hamiltonian system (2.10)-(2.11)-(2.12) and then using

the *generating function* to compute the initial conditions of the co-state, so that the problem is solved with a simple forward-integration; lastly, taking advantage of the well-developed theory of *linear* optimal control, it is possible to rewrite the nonlinear problem in a pseudo-linear form, and then to apply the common procedure to solve linear optimal control problems: this is what State Dependent Riccati Equation (SDRE) and Approximating Sequence of Riccati Equation (ASRE) methods do in order to solve a nonlinear optimal control problem, the former over an *infinite time-horizon* and the latter over a *finite time-horizon*. In the following, ASRE method will be employed extensively, while the interested reader is directed to [15] and [16] for further explanations regarding the SDRE method.

3.2 Approximating Sequence of Riccati Equation

The ASRE method transforms nonlinear dynamics and cost function into pseudo-linear and quadratic-like forms respectively, using state-dependent functions. Then, these functions are evaluated using the solutions of the previous iteration, and the sequence of linear time-varying problems is solved until convergence. This method has been first developed by Banks and Dinesh in [10], then it has been further studied by Çimen and Banks in [11] and [12]. At each step, rather than dealing with a Riccati equation, it is possible to solve the problem with a State-Transition Matrix (STM) approach, as presented in [13].

The dynamic system

$$\dot{\mathbf{x}} = \mathbf{f}\left[\mathbf{x}(t), \mathbf{u}(t), t\right], \tag{3.1}$$

and the scalar cost function to be minimized

$$J = \varphi\left[\mathbf{x}(t_f), t_f\right] + \int_{t_i}^{t_f} L\left[\mathbf{x}(t), \mathbf{u}(t), t\right] \mathrm{d}t, \tag{3.2}$$

are rearranged as

$$\dot{\mathbf{x}} = \mathbf{A}\left(\mathbf{x}(t), t\right)\mathbf{x} + \mathbf{B}\left(\mathbf{x}(t), \mathbf{u}(t), t\right)\mathbf{u}, \tag{3.3}$$

$$\begin{aligned} J = \frac{1}{2}\mathbf{x}^T(t_f)\mathbf{S}\left(\mathbf{x}(t_f), t_f\right)\mathbf{x}(t_f) + \frac{1}{2}\int_{t_i}^{t_f} \Big[\mathbf{x}^T(t)\mathbf{Q}\left(\mathbf{x}(t), t\right)\mathbf{x}(t) \\ + \mathbf{u}^T(t)\mathbf{R}\left(\mathbf{x}(t), \mathbf{u}(t), t\right)\mathbf{u}(t)\Big]\mathrm{d}t, \end{aligned} \tag{3.4}$$

where the functions $\mathbf{A}$, $\mathbf{B}$, $\mathbf{Q}$ and $\mathbf{R}$ have appropriate dimensions, and, furthermore, $\mathbf{S}$ and $\mathbf{Q}$ must be symmetric and *positive-semidefinite*, while $\mathbf{R}$ has to be symmetric and *positive-definite*. The previous expressions define a linear, time-variant optimal control problem, and the algorithm is started

using the initial conditions $\mathbf{x}_i$ and no control to evaluate the state-dependent functions, yielding a *Problem* 0:

$$\dot{\mathbf{x}}^{[0]} = \mathbf{A}\left(\mathbf{x}_i, t\right)\mathbf{x}^{[0]} + \mathbf{B}\left(\mathbf{x}_i, 0, t\right)\mathbf{u}^{[0]}, \tag{3.5}$$

$$\begin{aligned} J^{[0]} = \frac{1}{2}\mathbf{x}^{[0]T}(t_f)\mathbf{S}\left(\mathbf{x}_i, t_f\right)\mathbf{x}^{[0]}(t_f) + \frac{1}{2}\int_{t_i}^{t_f}\Big[\mathbf{x}^{[0]T}(t)\mathbf{Q}\left(\mathbf{x}_i, t\right)\mathbf{x}^{[0]}(t) \\ +\mathbf{u}^{[0]T}(t)\mathbf{R}\left(\mathbf{x}_i, 0, t\right)\mathbf{u}^{[0]}(t)\Big]\,\mathrm{d}t, \end{aligned} \tag{3.6}$$

The solution of this first problem is used to evaluate the state-dependent functions at the next step; thus, at a generic, subsequent iteration, *Problem* k has to be solved:

$$\dot{\mathbf{x}}^{[k]} = \mathbf{A}\left(\mathbf{x}^{[k-1]}(t), t\right)\mathbf{x}^{[k]} + \mathbf{B}\left(\mathbf{x}^{[k-1]}(t), \mathbf{u}^{[k-1]}(t), t\right)\mathbf{u}^{[k]}, \tag{3.7}$$

$$\begin{aligned} J^{[k]} = \frac{1}{2}\mathbf{x}^{[k]T}(t_f)\mathbf{S}\left(\mathbf{x}^{[k-1]}(t_f), t_f\right)\mathbf{x}^{[k]}(t_f) \\ +\frac{1}{2}\int_{t_i}^{t_f}\Big[\mathbf{x}^{[k]T}(t)\mathbf{Q}\left(\mathbf{x}^{[k-1]}(t), t\right)\mathbf{x}^{[k]}(t) \\ +\mathbf{u}^{[k]T}(t)\mathbf{R}\left(\mathbf{x}^{[k-1]}(t), \mathbf{u}^{[k-1]}(t), t\right)\mathbf{u}^{[k]}(t)\Big]\,\mathrm{d}t. \end{aligned} \tag{3.8}$$

Problem k is again a linear time-varying problem, that employs the solutions $\mathbf{x}^{[k-1]}$ and $\mathbf{u}^{[k-1]}$ from previous iteration $k-1$ to evaluate the state-dependent functions $\mathbf{A}$, $\mathbf{B}$, $\mathbf{Q}$ and $\mathbf{R}$. Solving *Problem* k gives $\mathbf{x}^{[k]}$ and $\mathbf{u}^{[k]}$.

In the present implementation of the ASRE algorithm, the stopping criteria for the iterative process is based on the difference between two consecutive iterations; convergence is assumed to be reached when

$$\left\|\mathbf{x}^{[k]} - \mathbf{x}^{[k-1]}\right\|_\infty = \max_{t\in\left[t_i, t_f\right]}\left\{\left|x_j^{[k]}(t) - x_j^{[k-1]}(t)\right|,\ j = 1, \ldots, n\right\} \leq \varepsilon, \tag{3.9}$$

where ε is a prescribed tolerance. This approximate method has been proved to converge to the optimal solution by Çimen and Banks in [12]. The work-flow of the algorithm is shown in figure 3.1. In conclusion, the ASRE procedure is based on solving iteratively a sequence of linear, time-varying optimal control problems, i.e., Linear Quadratic Regulator (LQR) problems, whose methodology will be addressed in the next section.

3.3 Solution by the State-Transition Matrix

The classical LQR problem can be subject to three different kinds of boundary conditions:

Hard constrained problem: both initial and final conditions are prescribed for *all* the n states.

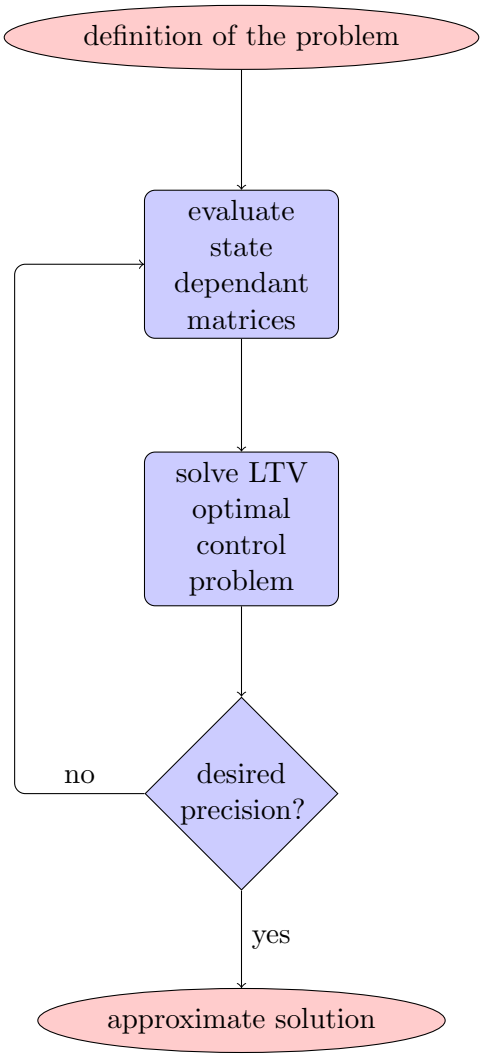


Figure 3.1: Work-flow of the ASRE algorithm

Soft constrained problem: only initial conditions are prescribed.

Mixed constrained problem: final conditions are given only for $q < n$ states.

Each of the three cases above is solved using the STM technique, but slight modifications occur to take into account the different boundary conditions. In general, when $x^{[k-1]}$ and $u^{[k-1]}$ are plugged into (3.3) and (3.4), the dynamics have the form

$$\dot{\mathbf{x}} = \mathbf{A}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{u}(t), \tag{3.10}$$

and the cost function is

$$J = \frac{1}{2}\mathbf{x}_f^T\mathbf{S}(t_f)\mathbf{x}_f + \frac{1}{2}\int_{t_i}^{t_f}\left[\mathbf{x}^T(t)\mathbf{Q}(t)\mathbf{x}(t) + \mathbf{u}^T(t)\mathbf{R}(t)\mathbf{u}(t)\right]\mathrm{d}t, \tag{3.11}$$

where the matrix $\mathbf{S}(t_f)$ varies according to the boundary conditions applied; still, in general, the Hamiltonian is

$$H = \mathbf{x}^T(t)\mathbf{Q}(t)\mathbf{x}(t) + \mathbf{u}^T(t)\mathbf{R}(t)\mathbf{u}(t) + \boldsymbol{\lambda}^T(t)\mathbf{A}(t)\mathbf{x}(t) + \boldsymbol{\lambda}^T(t)\mathbf{B}(t)\mathbf{u}(t), \tag{3.12}$$

so that the control, from equation (2.12), is

$$\frac{\partial H}{\partial \mathbf{u}} = \mathbf{R}(t)\mathbf{u}(t) + \mathbf{B}^T(t)\boldsymbol{\lambda}(t) = \mathbf{0} \tag{3.13}$$

$$\Downarrow$$

$$\mathbf{u}(t) = -\mathbf{R}^{-1}(t)\mathbf{B}^T(t)\boldsymbol{\lambda}(t) \tag{3.14}$$

and the Hamiltonian system of equation (2.16) will be

$$\begin{aligned}\begin{Bmatrix}\dot{\mathbf{x}}\\ \dot{\boldsymbol{\lambda}}\end{Bmatrix} &= \begin{Bmatrix}\dfrac{\partial H}{\partial \boldsymbol{\lambda}}(\mathbf{x},\boldsymbol{\lambda})\\ -\dfrac{\partial H}{\partial \mathbf{x}}(\mathbf{x},\boldsymbol{\lambda})\end{Bmatrix} = \begin{Bmatrix}\mathbf{A}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{u}(t)\\ -\mathbf{Q}(t)\mathbf{x}(t) - \mathbf{A}^T\boldsymbol{\lambda}\end{Bmatrix}\\ &= \begin{Bmatrix}\mathbf{A}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{R}^{-1}(t)\mathbf{B}^T(t)\boldsymbol{\lambda}(t)\\ -\mathbf{Q}(t)\mathbf{x}(t) - \mathbf{A}^T(t)\boldsymbol{\lambda}\end{Bmatrix},\end{aligned} \tag{3.15}$$

yielding the TPBVP

$$\begin{Bmatrix}\dot{\mathbf{x}}\\ \dot{\boldsymbol{\lambda}}\end{Bmatrix} = \begin{bmatrix}\mathbf{A}(t) & -\mathbf{B}(t)\mathbf{R}^{-1}(t)\mathbf{B}^T(t)\\ -\mathbf{Q}(t) & -\mathbf{A}^T(t)\end{bmatrix}\begin{Bmatrix}\mathbf{x}(t)\\ \boldsymbol{\lambda}(t)\end{Bmatrix}, \tag{3.16}$$

with the appropriate boundary conditions. Instead of solving the problem by the Riccati equation, it is possible to employ the STM $\boldsymbol{\Phi}(\tau, t)$, i.e., the matrix that, in general, solves

$$\begin{Bmatrix}\mathbf{x}(t)\\ \boldsymbol{\lambda}(t)\end{Bmatrix} = \boldsymbol{\Phi}(\tau, t)\begin{Bmatrix}\mathbf{x}(\tau)\\ \boldsymbol{\lambda}(\tau)\end{Bmatrix}. \tag{3.17}$$

For us it is more interesting to write the relation between initial conditions at time t_i and conditions at a generic time $t \in [t_i, t_f]$:

$$\begin{Bmatrix} \mathbf{x}(t) \\ \boldsymbol{\lambda}(t) \end{Bmatrix} = \boldsymbol{\Phi}(t_i, t) \begin{Bmatrix} \mathbf{x}(t_i) \\ \boldsymbol{\lambda}(t_i) \end{Bmatrix}. \tag{3.18}$$

To cut a long story short, the STM $\boldsymbol{\Phi}(\tau, t)$ accounts for the variations of the final state due to the variations of the initial state of a system with given dynamics. Moreover, for linear systems, this matrix maps also the state from a condition at time τ to a different condition at time t. In our case, the STM can be computed by solving the differential equation

$$\frac{\partial \boldsymbol{\Phi}(t_i, t)}{\partial t} = \begin{bmatrix} \mathbf{A}(t) & -\mathbf{B}(t)\mathbf{R}^{-1}(t)\mathbf{B}^T(t) \\ -\mathbf{Q}(t) & -\mathbf{A}^T(t) \end{bmatrix} \boldsymbol{\Phi}(t_i, t). \tag{3.19}$$

Of course, the transition from a condition to itself is mediated by an identity matrix, and this gives the initial condition

$$\boldsymbol{\Phi}_i = \boldsymbol{\Phi}(t_i, t_i) = \mathbf{I}. \tag{3.20}$$

For an autonomous system this computation would be easier, since in general

$$\frac{\partial \boldsymbol{\Phi}(t)}{\partial t} = \mathbf{M}\boldsymbol{\Phi}(t), \ \boldsymbol{\Phi}(t_i) = \mathbf{I} \tag{3.21}$$

$$\Downarrow$$

$$\boldsymbol{\Phi}(t) = \mathrm{e}^{\mathbf{M}t}, \tag{3.22}$$

while the the time-varying case requires the integration of the system matrix

$$\boldsymbol{\Phi}(t) = C\mathrm{e}^{\int \mathbf{M}(t)\mathrm{d}t}, \tag{3.23}$$

where C is a constant such that $\boldsymbol{\Phi}(t_i) = \mathbf{I}$. Equivalently, this can be done by partitioning the differential equation into four blocks

$$\begin{bmatrix} \dot{\boldsymbol{\Phi}}_{xx}(t_i, t) & \dot{\boldsymbol{\Phi}}_{x\lambda}(t_i, t) \\ \dot{\boldsymbol{\Phi}}_{\lambda x}(t_i, t) & \dot{\boldsymbol{\Phi}}_{\lambda\lambda}(t_i, t) \end{bmatrix} = \begin{bmatrix} \mathbf{A}(t) & -\mathbf{B}(t)\mathbf{R}^{-1}(t)\mathbf{B}^T(t) \\ -\mathbf{Q}(t) & -\mathbf{A}^T(t) \end{bmatrix} \begin{bmatrix} \boldsymbol{\Phi}_{xx}(t_i, t) & \boldsymbol{\Phi}_{x\lambda}(t_i, t) \\ \boldsymbol{\Phi}_{\lambda x}(t_i, t) & \boldsymbol{\Phi}_{\lambda\lambda}(t_i, t) \end{bmatrix}. \tag{3.24}$$

These four blocks can be studied separately

$$\dot{\boldsymbol{\Phi}}_{xx}(t_i, t) = \mathbf{A}(t)\boldsymbol{\Phi}_{xx}(t_i, t) - \mathbf{B}(t)\mathbf{R}^{-1}(t)\mathbf{B}^T(t)\boldsymbol{\Phi}_{\lambda x}(t_i, t), \ \boldsymbol{\Phi}_{xx}(t_i, t_i) = \mathbf{I}, \tag{3.25}$$

$$\dot{\boldsymbol{\Phi}}_{x\lambda}(t_i, t) = \mathbf{A}(t)\boldsymbol{\Phi}_{x\lambda}(t_i, t) - \mathbf{B}(t)\mathbf{R}^{-1}(t)\mathbf{B}^T(t)\boldsymbol{\Phi}_{\lambda\lambda}(t_i, t), \ \boldsymbol{\Phi}_{x\lambda}(t_i, t_i) = \mathbf{0}, \tag{3.26}$$

$$\dot{\boldsymbol{\Phi}}_{\lambda x}(t_i, t) = -\mathbf{Q}(t)\boldsymbol{\Phi}_{xx}(t_i, t) - \mathbf{A}^T(t)\boldsymbol{\Phi}_{\lambda x}(t_i, t), \ \boldsymbol{\Phi}_{\lambda x}(t_i, t_i) = \mathbf{0}, \tag{3.27}$$

$$\dot{\boldsymbol{\Phi}}_{\lambda\lambda}(t_i, t) = -\mathbf{Q}(t)\boldsymbol{\Phi}_{x\lambda}(t_i, t) - \mathbf{A}^T(t)\boldsymbol{\Phi}_{\lambda\lambda}(t_i, t), \ \boldsymbol{\Phi}_{\lambda\lambda}(t_i, t_i) = \mathbf{I}, \tag{3.28}$$

and integrated, to yield the four blocks of the STM

$$\mathbf{\Phi}_{xx}(t_i,t) = \mathbf{I} + \int_{t_i}^{t_f} \left[\mathbf{A}(t)\mathbf{\Phi}_{xx}(t_i,t) - \mathbf{B}(t)\mathbf{R}^{-1}(t)\mathbf{B}^T(t)\mathbf{\Phi}_{\lambda x}(t_i,t)\right] \mathrm{d}t, \tag{3.29}$$

$$\mathbf{\Phi}_{x\lambda}(t_i,t) = \int_{t_i}^{t_f} \left[\mathbf{A}(t)\mathbf{\Phi}_{x\lambda}(t_i,t) - \mathbf{B}(t)\mathbf{R}^{-1}(t)\mathbf{B}^T(t)\mathbf{\Phi}_{\lambda\lambda}(t_i,t)\right] \mathrm{d}t, \tag{3.30}$$

$$\mathbf{\Phi}_{\lambda x}(t_i,t) = \int_{t_i}^{t_f} \left[-\mathbf{Q}(t)\mathbf{\Phi}_{xx}(t_i,t) - \mathbf{A}^T(t)\mathbf{\Phi}_{\lambda x}(t_i,t)\right] \mathrm{d}t, \tag{3.31}$$

$$\mathbf{\Phi}_{\lambda\lambda}(t_i,t) = \mathbf{I} + \int_{t_i}^{t_f} \left[-\mathbf{Q}(t)\mathbf{\Phi}_{x\lambda}(t_i,t) - \mathbf{A}^T(t)\mathbf{\Phi}_{\lambda\lambda}(t_i,t)\right] \mathrm{d}t. \tag{3.32}$$

Having determined the STM $\mathbf{\Phi}(t_i,t)$, it is now possible to write the solution of the control problem as

$$\begin{Bmatrix}\mathbf{x}(t)\\ \boldsymbol{\lambda}(t)\end{Bmatrix} = \mathbf{\Phi}(t_i,t)\begin{Bmatrix}\mathbf{x}_i\\ \boldsymbol{\lambda}_i\end{Bmatrix}. \tag{3.33}$$

It is apparent that, in general, it is needed to evaluate $\boldsymbol{\lambda}_i$, and this is done requiring the enforcement of the boundary conditions; the three possible cases, *hard*, *soft* and *mixed* constrained problems, are discussed below.

3.3.1 Hard constrained problem

For the Hard Constrained Problem (HCP), boundary conditions are provided for *all* states both at initial and final time

$$\mathbf{x}(t_i) = \mathbf{x}_i, \tag{3.34}$$

$$\mathbf{x}(t_f) = \mathbf{x}_f, \tag{3.35}$$

and the value of $\boldsymbol{\lambda}_i$ can be obtained by simply enforcing the respect of the final conditions, writing

$$\begin{Bmatrix}\mathbf{x}(t_f)\\ \boldsymbol{\lambda}(t_f)\end{Bmatrix} = \mathbf{\Phi}(t_i,t_f)\begin{Bmatrix}\mathbf{x}_i\\ \boldsymbol{\lambda}_i\end{Bmatrix} \tag{3.36}$$

$$\Downarrow$$

$$\mathbf{x}(t_f) = \mathbf{x}_f = \mathbf{\Phi}_{xx}(t_i,t_f)\mathbf{x}_i + \mathbf{\Phi}_{x\lambda}(t_i,t_f)\boldsymbol{\lambda}_i \tag{3.37}$$

$$\Downarrow$$

$$\boldsymbol{\lambda}_i = \mathbf{\Phi}_{x\lambda}^{-1}(t_i,t_f)\left[\mathbf{x}_f - \mathbf{\Phi}_{xx}(t_i,t_f)\mathbf{x}_i\right], \tag{3.38}$$

so that the solutions of the problem for state and co-state will be

$$\mathbf{x}(t) = \mathbf{\Phi}_{xx}(t_i,t)\mathbf{x}_i + \mathbf{\Phi}_{x\lambda}(t_i,t)\mathbf{\Phi}_{x\lambda}^{-1}(t_i,t_f)\left[\mathbf{x}_f - \mathbf{\Phi}_{xx}(t_i,t_f)\mathbf{x}_i\right], \tag{3.39}$$

$$\boldsymbol{\lambda}(t) = \mathbf{\Phi}_{\lambda x}(t_i,t)\mathbf{x}_i + \mathbf{\Phi}_{\lambda\lambda}(t_i,t)\mathbf{\Phi}_{x\lambda}^{-1}(t_i,t_f)\left[\mathbf{x}_f - \mathbf{\Phi}_{xx}(t_i,t_f)\mathbf{x}_i\right], \tag{3.40}$$

and for the control

$$\mathbf{u}(t) = -\mathbf{R}^{-1}(t)\mathbf{B}^T(t)\boldsymbol{\lambda}(t) \tag{3.41}$$

$$\Downarrow$$

$$\begin{aligned}\mathbf{u}(t) = -\mathbf{R}^{-1}(t)\mathbf{B}^T(t)\boldsymbol{\Phi}_{\lambda x}(t_i,t)\mathbf{x}_i - \mathbf{R}^{-1}(t)\mathbf{B}^T(t)\boldsymbol{\Phi}_{\lambda\lambda}(t_i,t)\boldsymbol{\Phi}_{x\lambda}^{-1}(t_i,t_f)\mathbf{x}_f \\ +\mathbf{R}^{-1}(t)\mathbf{B}^T(t)\boldsymbol{\Phi}_{\lambda\lambda}(t_i,t)\boldsymbol{\Phi}_{x\lambda}^{-1}(t_i,t_f)\boldsymbol{\Phi}_{xx}(t_i,t_f)\mathbf{x}_i.\end{aligned} \tag{3.42}$$

A necessary condition for the existence of a solution is that the matrix $\boldsymbol{\Phi}_{x\lambda}$ must be invertible, i.e., non-singular.

3.3.2 Soft constrained problem

Soft Constrained Problem (SCP) only provide initial conditions for the state; this means that the final state will not be constrained, and thus the variations of the final state are taken into account inside the cost function using a weighting matrix $\mathbf{S}(t_f) \in \mathbb{R}^{n\times n}$:

$$J = \frac{1}{2}\mathbf{x}_f^T\mathbf{S}(t_f)\mathbf{x}_f + \frac{1}{2}\int_{t_i}^{t_f}\left[\mathbf{x}^T(t)\mathbf{Q}(t)\mathbf{x}(t) + \mathbf{u}^T(t)\mathbf{R}(t)\mathbf{u}(t)\right]\mathrm{d}t. \tag{3.43}$$

The control problem is defined by $2n$ differential equations, but only n boundary conditions are given; applying the *transversality condition* from equation (2.15), the missing boundary conditions are found through

$$\boldsymbol{\lambda}_f = \left(\frac{\partial\varphi}{\partial\mathbf{x}_f}\right)^T \tag{3.44}$$

$$\Downarrow$$

$$\boldsymbol{\lambda}(t_f) = \mathbf{S}(t_f)\mathbf{x}(t_f). \tag{3.45}$$

Enforcing the respect of the final conditions

$$\begin{Bmatrix}\mathbf{x}(t_f)\\ \boldsymbol{\lambda}(t_f)\end{Bmatrix} = \boldsymbol{\Phi}(t_i,t_f)\begin{Bmatrix}\mathbf{x}_i\\ \boldsymbol{\lambda}_i\end{Bmatrix} \tag{3.46}$$

$$\Downarrow$$

$$\begin{Bmatrix}\mathbf{x}(t_f)\\ \mathbf{S}(t_f)\mathbf{x}(t_f)\end{Bmatrix} = \boldsymbol{\Phi}(t_i,t_f)\begin{Bmatrix}\mathbf{x}_i\\ \boldsymbol{\lambda}_i\end{Bmatrix}, \tag{3.47}$$

yields an algebraic system of two vectorial equations with two vectorial unknowns, $\mathbf{x}(t_f)$ and $\boldsymbol{\lambda}_i$:

$$\mathbf{x}(t_f) = \boldsymbol{\Phi}_{xx}(t_i,t_f)\mathbf{x}_i + \boldsymbol{\Phi}_{x\lambda}(t_i,t_f)\boldsymbol{\lambda}_i, \tag{3.48}$$

$$\mathbf{S}(t_f)\mathbf{x}(t_f) = \boldsymbol{\Phi}_{\lambda x}(t_i,t_f)\mathbf{x}_i + \boldsymbol{\Phi}_{\lambda\lambda}(t_i,t_f)\boldsymbol{\lambda}_i. \tag{3.49}$$

Substituting expression (3.48) in (3.49) for $\mathbf{x}(t_f)$ and solving for $\boldsymbol{\lambda}_i$ gives

$$\boldsymbol{\lambda}_i = \left[\boldsymbol{\Phi}_{\lambda\lambda}(t_i,t_f) - \mathbf{S}(t_f)\boldsymbol{\Phi}_{x\lambda}(t_i,t_f)\right]^{-1}\left[\mathbf{S}(t_f)\boldsymbol{\Phi}_{xx}(t_i,t_f) - \boldsymbol{\Phi}_{\lambda x}(t_i,t_f)\right]\mathbf{x}_i, \tag{3.50}$$

so that the solution of the control problem is

$$\begin{aligned}\mathbf{x}(t) = \Big[&\boldsymbol{\Phi}_{xx}(t_i,t) + \boldsymbol{\Phi}_{x\lambda}(t_i,t)\left[\boldsymbol{\Phi}_{\lambda\lambda}(t_i,t_f) - \mathbf{S}(t_f)\boldsymbol{\Phi}_{x\lambda}(t_i,t_f)\right]^{-1}\\ &\left[\mathbf{S}(t_f)\boldsymbol{\Phi}_{xx}(t_i,t_f) - \boldsymbol{\Phi}_{\lambda x}(t_i,t_f)\right]\Big]\mathbf{x}_i,\end{aligned} \tag{3.51}$$

$$\begin{aligned}\boldsymbol{\lambda}(t) = \Big[&\boldsymbol{\Phi}_{\lambda x}(t_i,t) + \boldsymbol{\Phi}_{\lambda\lambda}(t_i,t)\left[\boldsymbol{\Phi}_{\lambda\lambda}(t_i,t_f) - \mathbf{S}(t_f)\boldsymbol{\Phi}_{x\lambda}(t_i,t_f)\right]^{-1}\\ &\left[\mathbf{S}(t_f)\boldsymbol{\Phi}_{xx}(t_i,t_f) - \boldsymbol{\Phi}_{\lambda x}(t_i,t_f)\right]\Big]\mathbf{x}_i,\end{aligned} \tag{3.52}$$

$$\mathbf{u}(t) = -\mathbf{R}^{-1}(t)\mathbf{B}^T(t)\boldsymbol{\lambda}(t). \tag{3.53}$$

In this case, a necessary condition for the existence of the solution is that the matrix

$$\boldsymbol{\Phi}_{\lambda\lambda}(t_i,t_f) - \mathbf{S}(t_f)\boldsymbol{\Phi}_{x\lambda}(t_i,t_f)$$

must be non-singular.

3.3.3 Mixed constrained problem

The Mixed Constrained Problem (MCP) is mid-way between the previous examined cases: $q < n$ states are prescribed both at initial and final time, while the others are given only at initial time. Taking advantage of the above discussions, both state $\mathbf{x}$ and co-state $\boldsymbol{\lambda}$ are partitioned into subsets, depending on whether the final condition is prescribed or not. Let the state be decomposed as $\mathbf{x} = (\mathbf{y}, \mathbf{z})$, where $\mathbf{y}$ are the q components known at final time, $\mathbf{y}(t_f) = \mathbf{y}_f$, and $\mathbf{z}$ are the remaining $n - q$ components of the state. The co-state shall be decomposed as $\boldsymbol{\lambda} = (\boldsymbol{\xi}, \boldsymbol{\eta})$ accordingly. The transversality condition becomes $\boldsymbol{\eta}_f = \mathbf{S}(t_f)\mathbf{z}_f$, where $\mathbf{S}(t_f) \in \mathbb{R}^{(n-q)\times(n-q)}$ is the weighting matrix of the final conditions. The STM is then partitioned, too, so that the solving system becomes

$$\begin{Bmatrix}\mathbf{y}(t)\\ \mathbf{z}(t)\\ \boldsymbol{\xi}(t)\\ \boldsymbol{\eta}(t)\end{Bmatrix} = \begin{bmatrix}\boldsymbol{\Phi}_{yy}(t_i,t) & \boldsymbol{\Phi}_{yz}(t_i,t) & \boldsymbol{\Phi}_{y\xi}(t_i,t) & \boldsymbol{\Phi}_{y\eta}(t_i,t)\\ \boldsymbol{\Phi}_{zy}(t_i,t) & \boldsymbol{\Phi}_{zz}(t_i,t) & \boldsymbol{\Phi}_{z\xi}(t_i,t) & \boldsymbol{\Phi}_{z\eta}(t_i,t)\\ \boldsymbol{\Phi}_{\xi y}(t_i,t) & \boldsymbol{\Phi}_{\xi z}(t_i,t) & \boldsymbol{\Phi}_{\xi\xi}(t_i,t) & \boldsymbol{\Phi}_{\xi\eta}(t_i,t)\\ \boldsymbol{\Phi}_{\eta y}(t_i,t) & \boldsymbol{\Phi}_{\eta z}(t_i,t) & \boldsymbol{\Phi}_{\eta\xi}(t_i,t) & \boldsymbol{\Phi}_{\eta\eta}(t_i,t)\end{bmatrix}\begin{Bmatrix}\mathbf{y}_i\\ \mathbf{z}_i\\ \boldsymbol{\xi}_i\\ \boldsymbol{\eta}_i\end{Bmatrix}. \tag{3.54}$$

Enforcing the respect of the final conditions gives an algebraic system of four vectorial equations with four vectorial unknowns $\mathbf{z}_f$, $\boldsymbol{\xi}_i$, $\boldsymbol{\xi}_f$ and $\boldsymbol{\eta}_i$

$$\begin{Bmatrix}\mathbf{y}_f\\ \mathbf{z}_f\\ \boldsymbol{\xi}_f\\ \mathbf{S}(t_f)\mathbf{z}_f\end{Bmatrix} = \begin{bmatrix}\boldsymbol{\Phi}_{yy}(t_i,t_f) & \boldsymbol{\Phi}_{yz}(t_i,t_f) & \boldsymbol{\Phi}_{y\xi}(t_i,t_f) & \boldsymbol{\Phi}_{y\eta}(t_i,t_f)\\ \boldsymbol{\Phi}_{zy}(t_i,t_f) & \boldsymbol{\Phi}_{zz}(t_i,t_f) & \boldsymbol{\Phi}_{z\xi}(t_i,t_f) & \boldsymbol{\Phi}_{z\eta}(t_i,t_f)\\ \boldsymbol{\Phi}_{\xi y}(t_i,t_f) & \boldsymbol{\Phi}_{\xi z}(t_i,t_f) & \boldsymbol{\Phi}_{\xi\xi}(t_i,t_f) & \boldsymbol{\Phi}_{\xi\eta}(t_i,t_f)\\ \boldsymbol{\Phi}_{\eta y}(t_i,t_f) & \boldsymbol{\Phi}_{\eta z}(t_i,t_f) & \boldsymbol{\Phi}_{\eta\xi}(t_i,t_f) & \boldsymbol{\Phi}_{\eta\eta}(t_i,t_f)\end{bmatrix}\begin{Bmatrix}\mathbf{y}_i\\ \mathbf{z}_i\\ \boldsymbol{\xi}_i\\ \boldsymbol{\eta}_i\end{Bmatrix}, \tag{3.55}$$

that, once solved, gives the initial co-state $\boldsymbol{\lambda}_i = (\boldsymbol{\xi}_i, \boldsymbol{\eta}_i)$, allowing, as before, to find the solution of the control problem.

Chapter 4

ASRE method: implementation issues and improvements

In the previous chapter, the ASRE methodology has been addressed in its theoretical formulation, but moving to the numerical implementation of the algorithm leads to the arousal of many issues, most of them related to the exploitation of the degrees of freedom that this method involves. In this chapter it will be then explored what are the features of ASRE method that can be exploited in order to improve the algorithm. From now on, for simplicity's sake, it will be assumed that the nonlinear dynamics are *control-affine*, i.e., the matrix $\mathbf{B}(\mathbf{x}, t)$ will be unique and not a function of the control $\mathbf{u}(t)$; this is made because dynamics that are not control-affine would led to further degrees of freedom that has been chosen not to explore. Also, the time-dependence of non-autonomous systems will be omitted to avoid a possible clumsiness of the notation.

4.1 Discretization of the solution

First of all, it is sensible to spend some words about the numerical discretization that is necessarily involved in the process: the time domain $[t_i, t_f]$ is divided into $N-1$ intervals using N internal points; those internal points constitute the discretized time-vector $\mathbf{t}$. Actually, the numerical integration of the STM, that is the core of the algorithm, is carried out using a *variable-step* numerical Ordinary Differential Equation (ODE) solver, that guarantees an higher precision, but the results, i.e., the state, the co-state and the control, are again formatted as vector (or matrices) containing samples taken at the time instants of the time-vector $\mathbf{t}$.

4.2 Dependence on the factorization

It must be said that the ASRE algorithm requires a factorization of the nonlinear dynamics with State Dependent Coefficients (SDCs). The reader will immediately notice that finding such a factorization can be a non-trivial task, mostly because, for systems with at least two states, there exists an infinite number of factorizations. Given a dynamical system

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \tag{4.1}$$

where $\mathbf{x}^T = \{x_1, x_2\}$, so that

$$\dot{x}_1 = f_1(x_1, x_2), \tag{4.2}$$
$$\dot{x}_2 = f_2(x_1, x_2), \tag{4.3}$$

it is possible, for example, to factorize the dynamics using a matrix $\mathbf{A}_1(\mathbf{x})$

$$\begin{Bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{Bmatrix} = \underbrace{\begin{bmatrix} \dfrac{f_1(x_1, x_2)}{x_1} & 0 \\ 0 & \dfrac{f_2(x_1, x_2)}{x_2} \end{bmatrix}}_{\mathbf{A}_1(\mathbf{x})} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix}, \tag{4.4}$$

or using a matrix $\mathbf{A}_2(\mathbf{x})$

$$\begin{Bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{Bmatrix} = \underbrace{\begin{bmatrix} 0 & \dfrac{f_1(x_1, x_2)}{x_2} \\ \dfrac{f_2(x_1, x_2)}{x_1} & 0 \end{bmatrix}}_{\mathbf{A}_2(\mathbf{x})} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix}. \tag{4.5}$$

Not only, as there exists a family of linear combinations of these two particular factorizations

$$\mathbf{A}_3(\mathbf{x}) = (1 - \alpha)\, \mathbf{A}_1(\mathbf{x}) + \alpha \mathbf{A}_2(\mathbf{x}), \quad 0 \leq \alpha \leq 1 \tag{4.6}$$

that will produce other factorizations of the initial system

$$\mathbf{A}_3(\mathbf{x})\mathbf{x} = [(1 - \alpha)\, \mathbf{A}_1(\mathbf{x}) + \alpha \mathbf{A}_2(\mathbf{x})]\, \mathbf{x} = (1 - \alpha)\, \mathbf{f}(\mathbf{x}) + \alpha \mathbf{f}(\mathbf{x}) = \mathbf{f}(\mathbf{x}), \tag{4.7}$$

and this procedure can be applied to any system with $n > 1$ states. One could think that choosing a factorization instead of another will not influence the solution to which the ASRE method converges, but, actually, this is not the case, as shown by the following example.

4.2.1 An example

Let us consider an optimal control problem, whose dynamics, taken from [18], are

$$\begin{aligned}\dot{x}_1 &= x_2,\\ \dot{x}_2 &= x_1 x_2 + u,\end{aligned}$$

with

$$\begin{aligned}\mathbf{x}(t_i) &= \begin{Bmatrix}2\\1\end{Bmatrix}, \quad t_i = 0,\\ \mathbf{x}(t_f) &= \begin{Bmatrix}4\\1\end{Bmatrix}, \quad t_f = 4.\end{aligned}$$

The cost function is

$$J = \int_{t_i}^{t_f} \left[x_1^2 + x_2^2 + u^2\right] \mathrm{d}t.$$

Since the dynamics of the problem are control-affine and the cost function is quadratic with reference to both states and control, the factorized form of the problem

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{A}\ \mathbf{x} + \mathbf{B}\ \mathbf{u},\\ J &= \int_{t_i}^{t_f} \left[\mathbf{x}^T \mathbf{Q}\ \mathbf{x} + \mathbf{u}^T \mathbf{R}\ \mathbf{u}\right] \mathrm{d}t,\end{aligned}$$

is unique (but from matrix $\mathbf{A}$) by using

$$\mathbf{B} = \begin{bmatrix}0\\1\end{bmatrix}, \quad \mathbf{Q} = \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}, \quad \mathbf{R} = \begin{bmatrix}1\end{bmatrix}.$$

The uncontrolled-dynamics are nonlinear, and thus they can be factorized in many forms; for example, two possible factorizations are

$$\mathbf{A}_1(\mathbf{x}) = \begin{bmatrix}0 & 1\\0 & x_1\end{bmatrix}, \quad \mathbf{A}_2(\mathbf{x}) = \begin{bmatrix}0 & 1\\x_2 & 0\end{bmatrix}.$$

As underlined in the previous section, given two factorizations, it is always possible to obtain infinite factorizations by linear combinations. Following this idea, a third factorization, or, better, a *family* of factorizations, is

$$\mathbf{A}_3\left(\alpha, \mathbf{x}\right) = \left(1 - \alpha\right) \mathbf{A}_1(\mathbf{x}) + \alpha \mathbf{A}_2(\mathbf{x}),$$

with $0 \leq \alpha \leq 1$. Figure 4.1 shows the trajectories obtained by solving the problem using the ASRE method; first of all, it is clear that the three different factorizations lead to three different solutions, and this of course

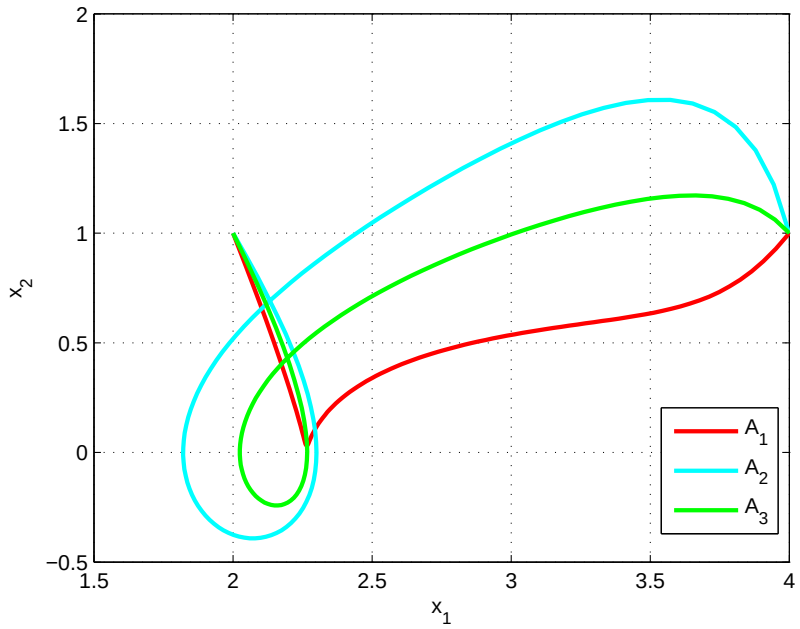


Figure 4.1: Trajectories of the solutions obtained with three different factorizations $\mathbf{A}_1$, $\mathbf{A}_2$ and $\mathbf{A}_3(\alpha = 0.5)$

means that either the three solutions are the same in terms of the cost function or one of them will be better than the others. Looking at the values of the cost function for the three factorizations in Table 4.1, it appears that the best one is $\mathbf{A}_1(\mathbf{x})$, and also its rate of convergence is the fastest; the trend is confirmed by factorization $\mathbf{A}_3(\mathbf{x})$, whose cost function and rate of convergence are bounded by the performances of the other two factorizations for $0 \leq \alpha \leq 1$.

The above example shows clearly that the choice of the factorization is of paramount importance for the ASRE method, because choosing the right factorization can lead to a better solution, while other factorizations can yield poor results or even not converge. In the next sections it will be studied what are the properties of a factorization that influence the convergence of the algorithm.

4.3 Consistency with boundary conditions

The first remark regarding the properties of a factorization is about *consistency*, i.e., the fact that it is possible for some factorizations to be not defined for some points of the state-space. In general, it is not possible to know *a priori* what will be the trajectory of the solution of a given problem,

Table 4.1: Performances of the three factorizations, termination tolerance 10^{-6}

	Objective function	Iterations
$\mathbf{A}_1$	52.24	12
$\mathbf{A}_2$	64.91	56
$\mathbf{A}_3$	53.13	21

so that a particular factorization cannot be checked for consistency along the trajectory, but, on the other hand, boundary conditions are prescribed, at least for initial time. Recalling *Problem 0* of the ASRE method for a control-affine problem

$$\dot{\mathbf{x}}^{[0]} = \mathbf{A}\left(\mathbf{x}_i, t\right)\mathbf{x}^{[0]} + \mathbf{B}\left(\mathbf{x}_i, t\right)\mathbf{u}^{[0]}, \tag{4.8}$$

$$\begin{aligned} J^{[0]} = \frac{1}{2}\mathbf{x}^{[0]T}(t_f)\mathbf{S}\left(\mathbf{x}_i, t_f\right)\mathbf{x}^{[0]}(t_f) + \frac{1}{2}\int_{t_i}^{t_f}\Big[\mathbf{x}^{[0]T}(t)\mathbf{Q}\left(\mathbf{x}_i, t\right)\mathbf{x}^{[0]}(t) \\ +\mathbf{u}^{[0]T}(t)\mathbf{R}\left(\mathbf{x}_i, t\right)\mathbf{u}^{[0]}(t)\Big]\mathrm{d}t, \end{aligned} \tag{4.9}$$

it is apparent that matrix $\mathbf{A}(\mathbf{x}, t)$ has to be defined for $\mathbf{x}(t_i) = \mathbf{x}_i$, otherwise any subsequent numerical manipulation would become cumbersome.

Regarding final conditions, that are assigned in the *hard constrained* and *mixed constrained* formulations, it must be noticed that the state-dependent matrices are never directly evaluated for $\mathbf{x}(t_f) = \mathbf{x}_f$; final conditions are enforced through the *state transition matrix*

$$\mathbf{x}(t) = \mathbf{\Phi}_{xx}(t_i, t)\mathbf{x}_i + \mathbf{\Phi}_{x\lambda}(t_i, t)\mathbf{\Phi}_{x\lambda}^{-1}(t_i, t_f)\left[\mathbf{x}_f - \mathbf{\Phi}_{xx}(t_i, t_f)\mathbf{x}_i\right]. \tag{4.10}$$

If the above expression was evaluated *analytically* for $t = t_f$, the state at time t will be exactly equal to the final condition enforced

$$\mathbf{x}(t_f) = \mathbf{\Phi}_{xx}(t_i, t_f)\mathbf{x}_i + \mathbf{\Phi}_{x\lambda}(t_i, t_f)\mathbf{\Phi}_{x\lambda}^{-1}(t_i, t_f)\left[\mathbf{x}_f - \mathbf{\Phi}_{xx}(t_i, t_f)\mathbf{x}_i\right] \tag{4.11}$$

$$= \mathbf{\Phi}_{xx}(t_i, t_f)\mathbf{x}_i + \mathbf{x}_f - \mathbf{\Phi}_{xx}(t_i, t_f)\mathbf{x}_i \tag{4.12}$$

$$= \mathbf{x}_f, \tag{4.13}$$

but, actually, the numerical procedure, knowing the expression of the time-function $\mathbf{\Phi}_{x\lambda}(t_i, t)$, first computes the numerical value of $\mathbf{\Phi}_{x\lambda}^{-1}(t_i, t_f)$, then evaluates the product of the previous functions

$$\mathbf{\Phi}_{x\lambda}(t_i, t)\mathbf{\Phi}_{x\lambda}^{-1}(t_i, t_f) = \mathbf{h}(t) \tag{4.14}$$

and lastly function $\mathbf{h}(t)$ is evaluated for $t = t_f$. Due to numerical approximations, it is possible that $\mathbf{h}(t_f) \neq \mathbf{I}$, and this means that the algorithm is

likely to produce trajectories that are nominally fixed at the final conditions, but actually only passing very near to them

$$\mathbf{x}(t_f) = \mathbf{x}_f + \boldsymbol{\delta}, \qquad \|\boldsymbol{\delta}\| \ll 1. \tag{4.15}$$

This fact, that could seem to be a lack of precision, in some cases allows to tolerate "nominal" inconsistency with final conditions, without worrying about the numerical procedure. Still, the consistency with final conditions is in general a critical issue.

Coming back to the choice of the factorizations, it is clear that the first required property of a factorization to be used within the ASRE method is the *consistency with initial conditions*, while the *consistency with final conditions* is, strictly speaking, not always necessary, but it is advisable to exclude those factorizations that are not consistent with final conditions.

4.4 Global controllability

A second property that the ASRE method requires in order to work at its best is *controllability*: a system is controllable if and only if always exists an admissible control that is able transfer the state from a condition to another in a finite time. For a non-controllable system, in general, it is not always possible to move from one state to another in a finite time, and this would pose some problems in terms of solving the optimal control problem associated, and so the choice of the factorization must be such that the system will be controllable.

Controllability is a property that can be checked for in different ways, depending on the system that has to be studied; in the following it will be presented how to check form controllability for nonlinear systems, linear time-variant systems and linear time-invariant systems.

4.4.1 Controllability check for nonlinear systems

In [4, 18] it has been proposed an algorithm for the nonlinear, control-affine system

$$\dot{\mathbf{x}} = \mathbf{a}(\mathbf{x}) + \mathbf{B}(\mathbf{x})\,\mathbf{u}, \tag{4.16}$$

where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{u} \in \mathbb{R}^m$, whose controllability is defined in terms of the dimension of the span of the smallest nonsingular and involutive distribution $\boldsymbol{\Delta}_c(\mathbf{x})$ containing the columns $\mathbf{b}_i$ of $\mathbf{B}(\mathbf{x})$, $1 \leq i \leq m$, and invariant under $\mathbf{a}(\mathbf{x})$ and the $\mathbf{b}_i$: a sufficient condition for system (4.16) to be controllable is

$$\operatorname{rank}\left[\boldsymbol{\Delta}_c(\mathbf{x})\right] = n \quad \forall \mathbf{x} \tag{4.17}$$

In order to evaluate $\boldsymbol{\Delta}_c(\mathbf{x})$ it is possible to use the following recursive algorithm:

1. Let $\mathbf{\Delta}_0 = \text{span}\,(\mathbf{B}) = \text{span}\,(\mathbf{b}_i)$.
2. Let $\mathbf{\Delta}_1 = \mathbf{\Delta}_0 + [\mathbf{a}, \mathbf{b}_i] + [\mathbf{b}_j, \mathbf{b}_i]$, $1 \leq i \leq m$, $1 \leq j \leq m$, where $[\mathbf{a}, \mathbf{g}]$ is the Lie bracket of $\mathbf{a}$ and $\mathbf{g}$, i.e.,
$$[\mathbf{a}, \mathbf{g}] = \frac{\partial \mathbf{g}}{\partial \mathbf{x}}\mathbf{a} - \frac{\partial \mathbf{a}}{\partial \mathbf{x}}\mathbf{g}, \tag{4.18}$$
and $+$ indicates the sum of the spans.
3. Let $\mathbf{\Delta}_k = \mathbf{\Delta}_{k-1} + [\mathbf{a}, \mathbf{d}_j] + [\mathbf{b}_i, \mathbf{d}_j]$, $1 \leq i \leq m$, $1 \leq j \leq m$, where $\{\mathbf{d}_j\}$ is a basis for $\mathbf{\Delta}_{k-1}$.
4. Terminate when $\mathbf{\Delta}_{k+1} = \mathbf{\Delta}_k$.

In this way it is possible to check for the controllability of a nonlinear, control-affine system, but in general it is not true that if the nonlinear system is controllable (*true controllability*) then any factorization will lead to a controllable system (*factored controllability*), as stated in [18]; this means that each factorization has to be checked for controllability.

4.4.2 Controllability check for linear time-variant systems

Similarly to the check for consistency, it must be verified whether a given factorization $\mathbf{A}(\mathbf{x}, t)$ is controllable; this is necessary because the ASRE algorithm is actually based on solving a sequence of LQR problems, which in general are Linear Time-Variant (LTV) problems, as can be seen by inspecting a generic iteration k of the sequence

$$\dot{\mathbf{x}}^{[k]} = \mathbf{A}\left(\mathbf{x}^{[k-1]}(t), t\right)\mathbf{x}^{[k]} + \mathbf{B}\left(\mathbf{x}^{[k-1]}(t), t\right)\mathbf{u}^{[k]}. \tag{4.19}$$

The controllability of the system depends only upon the choice of the factorization $\mathbf{A}\left(\mathbf{x}^{[k-1]}(t), t\right)$, which is actually only time dependent, since the solution $\mathbf{x}^{[k-1]}(t)$ of the previous iteration $k-1$ is known. The controllability property of the ASRE method for a given factorization is then reduced to the controllability of a LTV system.

The most natural way of checking the controllability of a LTV system is by employing the *controllability gramian*

$$\mathbf{P} = \int_{t_i}^{t_f} \mathbf{\Phi}(t_i, \tau)\mathbf{B}(\tau)\mathbf{B}^T(\tau)\mathbf{\Phi}^T(t_i, \tau)\mathrm{d}\tau, \tag{4.20}$$

where $\mathbf{\Phi}(t_i, \tau)$ is the STM, solution of

$$\dot{\mathbf{\Phi}}(t, \tau) = \mathbf{A}(t)\mathbf{\Phi}(t, \tau), \qquad \mathbf{\Phi}(t, t) = \mathbf{I}. \tag{4.21}$$

According to [5], a system is controllable over the domain $[t_i, t_f]$ if and only if the controllability gramian is non-singular, or, equivalently,

$$\text{rank}\,[\mathbf{P}] = n, \tag{4.22}$$

where n is the dimension of the state-space. Since the controllability gramian is a square matrix, it is sufficient to have

$$\det\left[\mathbf{P}\right] \neq 0, \tag{4.23}$$

to guarantee controllability on the whole time domain $[t_i, t_f]$. Actually, it would be better to know whether a factorization can lead to a non-controllable system *before* initializing the ASRE algorithm, i.e., to know if there is in general any point on the space-time domain that would lead to a loss of controllability; this can be done employing a figure that is explicitly function of the state and time, as shown in the next section.

4.4.3 Controllability check for linear time-invariant systems

The controllability criterion for linear time-invariant systems is a simplification of the general controllability criterion; for a system whose dynamics are

$$\dot{\mathbf{x}} = \mathbf{A}\ \mathbf{x} + \mathbf{B}\ \mathbf{u}, \tag{4.24}$$

where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{u} \in \mathbb{R}^m$, the controllability matrix is defined as

$$\mathbf{K}_c = \begin{bmatrix}\mathbf{B} & \mathbf{AB} & \mathbf{A}^2\mathbf{B} & \cdots & \mathbf{A}^{n-1}\mathbf{B}\end{bmatrix}, \tag{4.25}$$

and the system will be controllable if and only if

$$\operatorname{rank}\left[\mathbf{K}_c\right] = n. \tag{4.26}$$

This criterion stems directly from equation (4.22) applied to autonomous systems, and applying the criterion to state-dependent matrices $\mathbf{A}(\mathbf{x}, t)$ and $\mathbf{B}(\mathbf{x}, t)$ yields a controllability matrix that is again state-dependent

$$\mathbf{K}_c(\mathbf{x}, t) = \begin{bmatrix}\mathbf{B}(\mathbf{x}, t) & \mathbf{A}(\mathbf{x}, t)\,\mathbf{B}(\mathbf{x}, t) & \cdots & \mathbf{A}^{n-1}(\mathbf{x}, t)\,\mathbf{B}(\mathbf{x}, t)\end{bmatrix}, \tag{4.27}$$

and using this expression it is easy to investigate whether a given factorization $\mathbf{A}(\mathbf{x}, t)$ will produce a system that is controllable for any state and time, without having the need of knowing the actual trajectory of the solution. The resulting check is very conservative, because any factorization that can lead to an uncontrollable system will be discarded, even if the uncontrollability is located on a point of the space-time domain that is not actually near the trajectory of the solution.

4.5 Optimal controllability

Until now it has been studied what are the factorizations that, even though they are true representations of the nonlinear problem, do not allow to solve the optimal control problem using the ASRE algorithm. After discarding

the factorizations that are not consistent or not always controllable, still the amount of factorizations that can be employed is typically very large, but remembering section 4.2.1, it would be interesting to automatize the choice of the factorization so to find the best solution in terms of cost function. Speaking of how to get that "best" solution, again the property to work with is the controllability of the system: this is *maximized* by choosing a proper factorization.

The correlation between controllability and optimality is only conjectured, and it is based on the fact that the more a system is controllable, the less should be the control effort, as hinted in [19], with a benefit in terms of cost function, that is typically an increasing function of the control effort.

At this point, controllability must be regarded as a merit figure and no longer as a binary property of a system. How to characterize this continuous property? While the distinction between controllable and uncontrollable is very sharp, defining the level of controllability requires some thought; a good way to measure the controllability could be the *controllability radius*, defined in [20, 22] as the smallest 2-norm of a perturbation of $(\mathbf{A}, \mathbf{B})$ that makes the system uncontrollable

$$\begin{aligned} \rho(\mathbf{A}, \mathbf{B}) &= \min_{\Delta_A, \Delta_B} \|(\Delta_A, \Delta_B)\|, \\ \text{s.t. } &(\mathbf{A} + \Delta_A, \mathbf{B} + \Delta_B) \text{ is uncontrollable.} \end{aligned} \tag{4.28}$$

It has been proved in [21] that

$$\rho(\mathbf{A}, \mathbf{B}) = \min_{\lambda \in \mathbb{C}} \sigma_{min}\left[(\mathbf{A} - \lambda \mathbf{I}\mathbf{B})\right], \tag{4.29}$$

where the function σ_{min} returns the smallest singular value of its matrix argument. Actually this method is not the most suited for the ASRE, because it is based on evaluating the minimum perturbation that makes the system uncontrollable around a nominal point which is not so easy to determine.

A better way to characterize controllability for the ASRE algorithm is still based on evaluating the distance of the system from being uncontrollable, but this time the approach is different: starting from the notion that a system is controllable if the controllability gramian $\mathbf{P}(t_f, t_i)$ is not singular

$$\det\left[\mathbf{P}(t_f, t_i)\right] \neq 0, \tag{4.30}$$

it could seem reasonable to adopt the value of the determinant itself as a merit figure of controllability. Actually, as noticed in [20], such a choice would be misleading, since it is easy to prove that a matrix that is nearly singular can show a determinant much greater than zero:

$$\mathbf{A} = \begin{bmatrix} \dfrac{1}{\varepsilon^{2k+1}} & \varepsilon \\ \varepsilon & \varepsilon \end{bmatrix},$$

$$\det\left[\mathbf{A}\right] = \varepsilon^{1-2k-1} - \varepsilon^2 = \varepsilon^{-2k} - \varepsilon^2.$$

It is clear that if $\varepsilon \ll 1$, and this is typical of numerical procedures that approximate zeros with small values, then the matrix $\mathbf{A}$ will be almost singular, while the determinant will be much greater than one, for $k \geq 1$. To avoid this kind of problems, it can be remembered from linear algebra that the determinant of a matrix is equal to the product of its eigenvalues λ_i

$$\det\left[\mathbf{P}(t_f, t_i)\right] = \prod_{i=1}^{n} \lambda_i. \tag{4.31}$$

This means that among the controllability gramian's eigenvalues of an uncontrollable system, at least one one of them must be null. This fact can be exploited for a controllable system, assuming that the smallest absolute eigenvalue is a good index of how much the matrix is close to singularity; in fact, decomposing a gramian matrix using the spectral theorem

$$\mathbf{P} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}, \tag{4.32}$$

where $\mathbf{U}$ is the matrix whose column are the eigenvectors of $\mathbf{P}$ and $\mathbf{\Lambda}$ is the diagonal matrix of the eigenvalues of $\mathbf{P}$, evaluating its determinant

$$\det\left[\mathbf{P}\right] = \det\left[\mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}\right], \tag{4.33}$$

and applying the properties of the determinant operator

$$\det\left[\mathbf{X}\mathbf{Y}\right] = \det\left[\mathbf{X}\right]\det\left[\mathbf{Y}\right], \qquad \det\left[\mathbf{X}^{-1}\right] = \frac{1}{\det\left[\mathbf{X}\right]}, \tag{4.34}$$

$$\Downarrow$$

$$\det\left[\mathbf{P}\right] = \det\left[\mathbf{U}\right]\det\left[\mathbf{\Lambda}\right]\det\left[\mathbf{U}^{-1}\right] = \det\left[\mathbf{\Lambda}\right] = \det\begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix}, \tag{4.35}$$

makes clear that a matrix will be close to singularity only when at least an eigenvalue is close to zero, regardless of the determinant's value. Going beyond this, it must be taken into account that eigenvalues suffer of ill conditioning, i.e., small perturbations of a matrix can lead to big variations of its eigenvalues, as underlined by Paige in [20]; to overcome this issues it is recommended to employ *singular values* rather than eigenvalues: the singular values of a matrix $\mathbf{A}$ are the square roots of the eigenvalues of the matrix $\mathbf{A}^*\mathbf{A}$, where $\mathbf{A}^*$ is the *hermitian conjugate* of $\mathbf{A}$. The singular values of a matrix are positive real numbers that can be used instead of eigenvalues.

In conclusion, it has been verified that it is possible to define a controllability index based on the smallest singular value of the controllability gramian of a system stemming from a given factorization of the dynamics. Maximizing this controllability index allows to find the best factorization, as it will be shown in the next section.

4.6 Implementation of the modified ASRE

Bearing in mind all that has been said in the previous sections, it is now time to proceed in defining the details of how the new algorithm will work. Given the optimal control problem defined by the dynamics

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t), \quad \begin{cases} \mathbf{x} \in \mathbb{R}^n \\ \mathbf{u} \in \mathbb{R}^m \end{cases} \tag{4.36}$$

where the system is supposed to be control-affine, for the sake of simplicity, along with the cost function

$$J = \varphi(\mathbf{x}_f, t_f) + \int_{t_i}^{t_f} L[\mathbf{x}(t), \mathbf{u}(t), t]\, \mathrm{d}t, \tag{4.37}$$

where $L[\mathbf{x}(t), \mathbf{u}(t), t]$ is supposed to be a quadratic form of the control $\mathbf{u}(t)$, and the boundary conditions

$$\mathbf{x}(t_i) = \mathbf{x}_i \in \mathbb{R}^n, \tag{4.38}$$

$$\mathbf{x}(t_f) = \mathbf{x}_f \in \mathbb{R}^q, \tag{4.39}$$

where $0 \leq q \leq n$, the starting point is the set of h factorizations such that

$$\mathbf{f}(\mathbf{x}, \mathbf{u}, t) = \mathbf{A}_j(\mathbf{x}, t)\,\mathbf{x} + \mathbf{B}(\mathbf{x}, t)\,\mathbf{u}, \quad j = 1, 2, \ldots, h. \tag{4.40}$$

Before initializing the ASRE algorithm, the factorizations are checked for consistency with initial conditions, selecting only those SDC matrices that fulfil the condition

$$\mathbf{A}_j(\mathbf{x}_i, t_i) \in \mathbb{R}^{n \times n}, \tag{4.41}$$

while any other matrix is discarded; next, the subset of factorizations that are consistent with initial conditions is checked for global controllability, making use of controllability matrix $\mathbf{K}_c$: if it is possible to find at least one point $\left[\overline{\mathbf{x}}, \overline{t}\right]$ in the space-time domain that solves the equation

$$\det[\mathbf{M}(\mathbf{x}, t)] = 0, \tag{4.42}$$

where $\mathbf{M}(\mathbf{x}, t)$ is the square matrix

$$\mathbf{M}(\mathbf{x}, t) = \mathbf{K}_c(\mathbf{x}, t)\,\mathbf{K}_c^T(\mathbf{x}, t), \tag{4.43}$$

then the associated factorization is not always controllable. Expression (4.43) is needed because, as suggested in [15], the controllability matrix for multiple-input systems is not square, and so it is not possible to evaluate directly its determinant to check for uncontrollability, while the matrix $\mathbf{M}(\mathbf{x}, t)$ is a square matrix that clearly becomes singular when matrix $\mathbf{K}_c(\mathbf{x}, t)$ is not full rank or, equivalently, when the system is not controllable. Equation (4.42)

is, in general, a nonlinear algebraic equation, and is solved numerically with the *Levenberg-Marquardt method*, using as starting guess the initial boundary condition $\mathbf{x}_i$.

Then, those k matrices that are both consistent with initial conditions and always controllable are passed to the ASRE algorithm, that now has a different work-flow, because the method embeds the maximization of the controllability of the system:

1. As for the classical ASRE, the process is initialized using the initial conditions, while for any other subsequent iteration it is employed the solution of the previous iteration: at the first iteration the matrices $\mathbf{A}_j(\mathbf{x}, t)$ are evaluated for $\mathbf{x}(t) = \mathbf{x}_i$, yielding a set of matrices $\mathbf{A}_j^{[1]}(t) = \mathbf{A}_j(\mathbf{x}_i, t)$, while at a generic iteration $p > 1$ the SDC matrices are evaluated for $\mathbf{x}(t) = \mathbf{x}^{[p-1]}(t)$, thus obtaining a set of matrices $\mathbf{A}_j^{[p]}(t) = \mathbf{A}_j\left(\mathbf{x}^{[p-1]}(t), t\right)$.

2. Then, it is defined a matrix $\mathbf{A}^{[p]}(\boldsymbol{\alpha}, t)$ that is a linear combination of all the factorizations $\mathbf{A}_j^{[p]}(t)$

$$\mathbf{A}^{[p]}(t) = \prod_{j=1}^{k-1}(1-\alpha_j)\,\mathbf{A}_1^{[p]} + \sum_{i=2}^{k-1}\alpha_{i-1}\prod_{j=i}^{k-1}(1-\alpha_j)\,\mathbf{A}_i^{[p]} + \alpha_{k-1}\mathbf{A}_k^{[p]}, \tag{4.44}$$

where $\boldsymbol{\alpha}^T = \left[\alpha_1, \alpha_2, \ldots, \alpha_{k-1}\right]$ is a vector of coefficients. By properly choosing the value of $\boldsymbol{\alpha}$, it is possible to obtain any of the factorizations $\mathbf{A}_j^{[p]}(t)$, e.g., having three factorizations $\mathbf{A}_1(t)$, $\mathbf{A}_2(t)$ and $\mathbf{A}_3(t)$, matrix $\mathbf{A}(\boldsymbol{\alpha}, t)$ will be

$$\mathbf{A}(\boldsymbol{\alpha}, t) = (1-\alpha_1)(1-\alpha_2)\,\mathbf{A}_1(t) + \alpha_1(1-\alpha_2)\,\mathbf{A}_2(t) + \alpha_2\mathbf{A}_3(t), \tag{4.45}$$

with

$$\boldsymbol{\alpha} = \begin{Bmatrix}\alpha_1\\ \alpha_2\end{Bmatrix}. \tag{4.46}$$

The three "parent" factorizations are easily recovered:

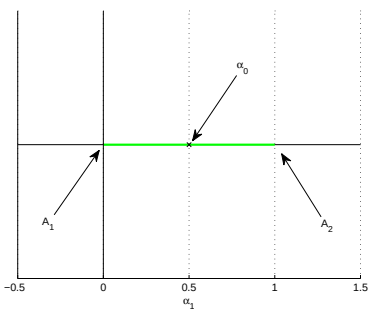
$$\boldsymbol{\alpha}_1 = \begin{Bmatrix}0\\ 0\end{Bmatrix} \quad \mathbf{A}(\boldsymbol{\alpha}_1, t) = \mathbf{A}_1(t) \tag{4.47}$$

$$\boldsymbol{\alpha}_2 = \begin{Bmatrix}1\\ 0\end{Bmatrix} \quad \mathbf{A}(\boldsymbol{\alpha}_2, t) = \mathbf{A}_2(t) \tag{4.48}$$

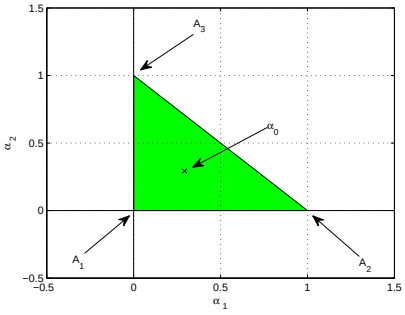
$$\boldsymbol{\alpha}_3 = \begin{Bmatrix}0\\ 1\end{Bmatrix} \quad \mathbf{A}(\boldsymbol{\alpha}_3, t) = \mathbf{A}_3(t) \tag{4.49}$$

This family of factorizations is consistent with the nonlinear problem, i.e.,

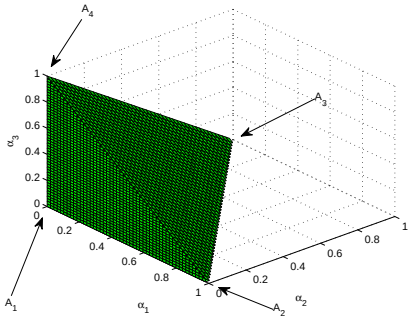
$$\dot{\mathbf{x}} = \mathbf{A}(\boldsymbol{\alpha}, t)\,\mathbf{x} + \mathbf{B}(t)\mathbf{u}, \tag{4.50}$$



(a) $\boldsymbol{\alpha}$ domain for $k = 2$



(b) $\boldsymbol{\alpha}$ domain for $k = 3$



(c) $\boldsymbol{\alpha}$ domain for $k = 4$

Figure 4.2: Domain of vector $\boldsymbol{\alpha}$

only if the 1-norm of $\boldsymbol{\alpha}$ is equal or less than one and its components are positive

$$\begin{cases} \|\boldsymbol{\alpha}\|_1 \leq 1 \\ \alpha_j \geq 0, \quad j = 1, 2, \ldots, k-1 \end{cases} \tag{4.51}$$

This means that, for the three factorizations case, the domain of $\boldsymbol{\alpha}$ will be a triangle whose vertices lie at the three point $\boldsymbol{\alpha}_1$, $\boldsymbol{\alpha}_2$ and $\boldsymbol{\alpha}_3$, as depicted in Figure 4.2b.

3. Next, matrix $\mathbf{A}^{[p]}(\boldsymbol{\alpha}, t)$ is used to determine which is the factorization that maximizes the controllability of the system; recalling the controllability index, defined as the minimum singular value of the controllability gramian, it is necessary to solve the following problem in order to optimize the choice of the factorization: given the linear, time-variant system

$$\dot{\mathbf{x}}^{[p]}(t) = \mathbf{A}^{[p]}(\boldsymbol{\alpha}, t)\,\mathbf{x}^{[p]}(t) + \mathbf{B}^{[p]}(t)\mathbf{u}^{[p]}(t) \tag{4.52}$$

find $\boldsymbol{\alpha}$ that maximizes the minimum singular value of the controllability

gramian

$$\mathbf{P}(\boldsymbol{\alpha}) = \int_{t_i}^{t_f} \boldsymbol{\Phi}\left(\boldsymbol{\alpha}, t_i, \tau\right) \mathbf{B}(\tau)\mathbf{B}^T(\tau)\boldsymbol{\Phi}^T\left(\boldsymbol{\alpha}, t_i, \tau\right) \mathrm{d}\tau, \tag{4.53}$$

where $\boldsymbol{\Phi}(\boldsymbol{\alpha}, t_i, \tau)$ is, as usual, the STM that solves for

$$\frac{\partial \boldsymbol{\Phi}\left(\boldsymbol{\alpha}, t, \tau\right)}{\partial t} = \mathbf{A}\left(\boldsymbol{\alpha}, t\right) \boldsymbol{\Phi}\left(\boldsymbol{\alpha}, t, \tau\right), \qquad \boldsymbol{\Phi}\left(\boldsymbol{\alpha}, t, t\right) = \mathbf{I}. \tag{4.54}$$

The constraints to which the optimization variable is subject are

$$\begin{cases} \|\boldsymbol{\alpha}\|_1 \leq 1 \\ \alpha_j \geq 0, \quad j = 1, 2, \ldots, k-1 \end{cases} \tag{4.55}$$

It could seem that the constraints are nonlinear, but actually they can be reformulated in a linear manner:

$$\begin{cases} \sum_j \alpha_j \leq 1 \\ \alpha_j \geq 0, \\ j = 1, 2, \ldots, k-1 \end{cases} \tag{4.56}$$

The above problem can be formally expressed as

$$\max_{\boldsymbol{\alpha}} \sigma_{min}\left[\mathbf{P}(\boldsymbol{\alpha})\right] \quad \text{subject to} \quad \begin{cases} \|\boldsymbol{\alpha}\|_1 \leq 1 \\ \alpha_j \geq 0, \quad j = 1, 2, \ldots, k-1 \end{cases} \tag{4.57}$$

and it is solved numerically using NLP, which requires a starting guess for $\boldsymbol{\alpha}$: the first iteration, $p = 1$, employs a value of $\boldsymbol{\alpha}$ that is located in the middle of the domain, as shown in Figure 4.2a and 4.2b, while for any iteration $p > 1$ the starting guess for $\boldsymbol{\alpha}$ is the optimal value of the previous iteration. The solution consists in a vector $\overline{\boldsymbol{\alpha}}$ for which the controllability of the system is maximized; this solution is associated to an optimal factorization $\mathbf{A}_{opt}^{[p]}(t) = \mathbf{A}^{[p]}\left(\overline{\boldsymbol{\alpha}}, t\right)$.

4. Now the algorithm proceeds with the classical ASRE iteration, solving the LTV optimal control problem

$$\dot{\mathbf{x}}^{[p]} = \mathbf{A}_{opt}^{[p]}(t)\mathbf{x}^{[p]} + \mathbf{B}^{[p]}(t)\mathbf{u}^{[p]}, \tag{4.58}$$

$$J^{[p]} = \frac{1}{2}\mathbf{x}^{[p]T}\mathbf{S}^{[p]}\mathbf{x}^{[p]} + \frac{1}{2}\int_{t_i}^{t_f} \left[\mathbf{x}^{[p]T}\mathbf{Q}^{[p]}(t)\mathbf{x}^{[p]} + \mathbf{u}^{[p]T}\mathbf{R}^{[p]}(t)\mathbf{u}^{[p]}\right] \mathrm{d}t. \tag{4.59}$$

to find $\mathbf{x}^{[p]}(t)$ and $\mathbf{u}^{[p]}(t)$.

As before, the routine is repeated until the termination criterion is satisfied. This modified ASRE (MASRE) algorithm is now able to be fed with many factorization of the nonlinear problem, and *at each iteration* it will be selected the optimal factorization in terms of the controllability index. The work-flow of the MASRE algorithm is shown in Figure 4.3.

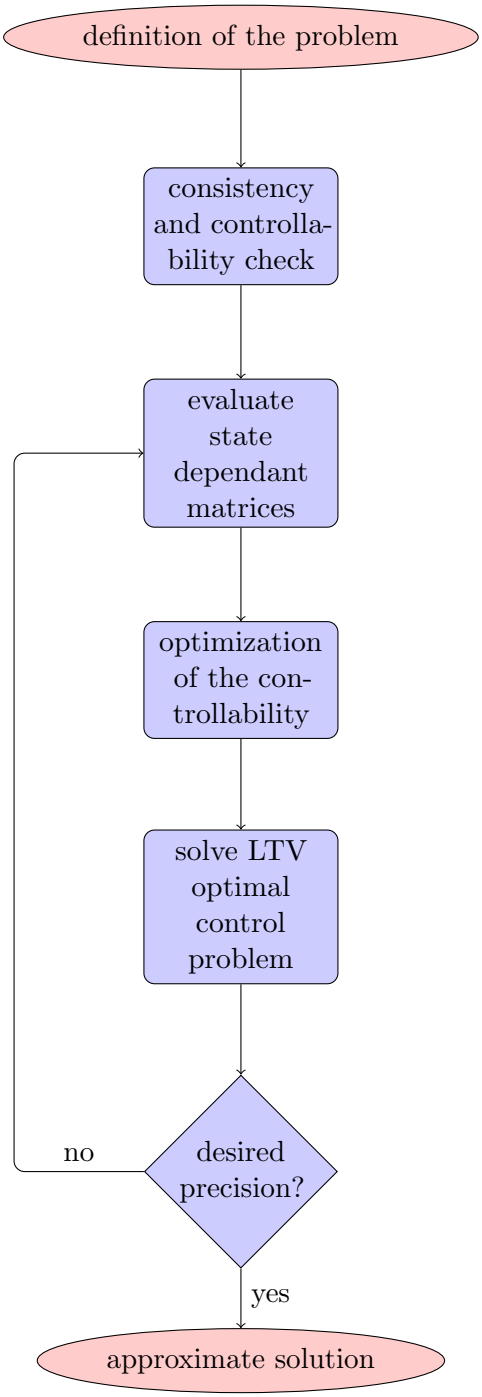


Figure 4.3: Work-flow of the modified ASRE algorithm

4.7 A note on automatic factorization

It would be now interesting to find a way to compute automatically the SDC factorizations of the dynamical system starting from the nonlinear equation (2.1), i.e., to find a matrix $\mathbf{A}$ and a matrix $\mathbf{B}$ such that

$$\mathbf{A}(\mathbf{x},t)\mathbf{x}+\mathbf{B}(\mathbf{x},\mathbf{u},t)\mathbf{u}=\mathbf{f}(\mathbf{x},\mathbf{u},t). \tag{4.60}$$

For the sake of simplicity, it shall be assumed that the dynamical system is control-affine. This means that only one $\mathbf{B}$ matrix exists:

$$\mathbf{A}(\mathbf{x},t)\mathbf{x}+\mathbf{B}(\mathbf{x},t)\mathbf{u}=\mathbf{f}(\mathbf{x},\mathbf{u},t). \tag{4.61}$$

Matrix $\mathbf{A}$ is then the solution of

$$\mathbf{A}(\mathbf{x},t)\mathbf{x}=\mathbf{f}(\mathbf{x},\mathbf{u},t)-\mathbf{B}(\mathbf{x},t)\mathbf{u}=\mathbf{g}(\mathbf{x},t). \tag{4.62}$$

The expression for $\mathbf{g}(\mathbf{x},t)$ is obtained easily, but the resulting system to be solved is of course under-determined, meaning that there are more unknowns (the elements of $\mathbf{A}$) than equations. This gives rise to the infinite possible factorizations, but it is necessary to develop a routine to solve the system as many times as needed.

First, it is necessary to determine if the system can be solved: to do this it is possible to enforce the elegant algorithm devised by R. W. Bass in [17]. Given a vectorial function $\mathbf{g}(\mathbf{x},t)$ such that $\mathbf{g}(\mathbf{0},t)=\mathbf{0}$, the following identity is true:

$$\mathbf{g}(\mathbf{x},t)\equiv\mathbf{g}(\mathbf{x},t)-\mathbf{g}(\mathbf{0},t)\equiv\int_0^1\frac{\partial\mathbf{g}(\lambda\mathbf{x},t)}{\partial\lambda}\mathrm{d}\lambda\equiv\int_0^1\left(\left[\frac{\partial\mathbf{g}(\mathbf{x},t)}{\partial\mathbf{x}}\right]_{\mathbf{x}=\lambda\mathbf{x}}\mathbf{x}\right)\mathrm{d}\lambda, \tag{4.63}$$

so that

$$\mathbf{g}(\mathbf{x},t)\equiv\int_0^1\left[\frac{\partial\mathbf{g}(\mathbf{x},t)}{\partial\mathbf{x}}\right]_{\mathbf{x}=\lambda\mathbf{x}}\mathrm{d}\lambda\;\mathbf{x}=\mathbf{A}_{parent}\mathbf{x} \tag{4.64}$$

$$\Downarrow$$

$$\mathbf{A}_{parent}(\mathbf{x},t)=\int_0^1\left[\frac{\partial\mathbf{g}(\mathbf{x},t)}{\partial\mathbf{x}}\right]_{\mathbf{x}=\lambda\mathbf{x}}\mathrm{d}\lambda. \tag{4.65}$$

It must be noticed that this algorithm works only for those function that are null in the origin. Once a particular solution $\mathbf{A}_{parent}$ is obtained, any other solution can be seen as the sum of the particular solution with a matrix

$\mathbf{A}_{null}$ whose rows belong to the *null space* of the $\mathbf{x}$ vector:

$$(\mathbf{A}_{parent} + \mathbf{A}_{null})\,\mathbf{x} = \mathbf{A}_{parent}\mathbf{x} + \mathbf{A}_{null}\mathbf{x} = \mathbf{g}(\mathbf{x}, t) \tag{4.66}$$

$$\Downarrow$$

$$\mathbf{A}_{null}\mathbf{x} = \mathbf{0} \tag{4.67}$$

$$\Downarrow$$

$$\begin{bmatrix} a_{i1}^{null} & a_{i2}^{null} & \cdots & a_{in}^{null} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = 0, \quad i = 1, 2, \ldots, n. \tag{4.68}$$

Then, once a basis for the null space has been computed, it is possible to combine linearly the elements of that basis to obtain the rows of the $\mathbf{A}_{null}$ matrices. The linear combination can be made using random coefficients, creating as many factorization as needed. Each of these factorizations will then be checked for consistency with initial conditions and global controllability.

4.7.1 Example of automatic factorization

Given a nonlinear, control affine dynamical system

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= x_1 x_2 + u, \end{aligned}$$

the system is then divided into an "uncontrolled-dynamics" part and a "control" part:

$$\begin{Bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{Bmatrix} = \begin{Bmatrix} x_2 \\ x_1 x_2 \end{Bmatrix} + \begin{Bmatrix} 0 \\ u \end{Bmatrix}.$$

Since the linear nature of the "control" part, it easy to factorize it:

$$\begin{Bmatrix} 0 \\ u \end{Bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} u.$$

Then, the non-linear "uncontrolled-dynamics" part is factorized using the algorithm presented above:

$$\mathbf{g}(\mathbf{x}) = \begin{Bmatrix} x_2 \\ x_1 x_2 \end{Bmatrix} = \mathbf{A}\mathbf{x}.$$

First, the *Jacobian* of the vector field $\mathbf{g}(\mathbf{x})$ is computed:

$$\nabla \mathbf{g}(\mathbf{x}) = \begin{bmatrix} 0 & 1 \\ x_2 & x_1 \end{bmatrix}.$$

Subsequently, the Jacobian of the vector field is evaluated at $\mathbf{x} = \lambda\mathbf{x}$:

$$\begin{bmatrix} 0 & 1 \\ x_2 & x_1 \end{bmatrix}_{\mathbf{x}=\lambda\mathbf{x}} = \begin{bmatrix} 0 & 1 \\ \lambda x_2 & \lambda x_1 \end{bmatrix}.$$

Then, the expression is integrated over λ in the interval $[0, 1]$, yielding the "parent" factorization $\mathbf{A}_{parent}$:

$$\int_0^1 \begin{bmatrix} 0 & 1 \\ \lambda x_2 & \lambda x_1 \end{bmatrix} \mathrm{d}\lambda = \begin{bmatrix} \int_0^1 0\mathrm{d}\lambda & \int_0^1 1\mathrm{d}\lambda \\ \int_0^1 \lambda x_2 \mathrm{d}\lambda & \int_0^1 \lambda x_1 \mathrm{d}\lambda \end{bmatrix}$$

$$\Downarrow$$

$$\mathbf{A}_{parent}(\mathbf{x}) = \begin{bmatrix} 0 & 1 \\ \frac{1}{2}x_2 & \frac{1}{2}x_1 \end{bmatrix}.$$

Next, it is necessary to evaluate a basis for the *null space* of the vector $\mathbf{x}$; since this example is set in a 2D space, the null space is the line orthogonal to the vector $\mathbf{x}$:

$$\mathrm{Null}(\mathbf{x}) = \alpha \begin{bmatrix} -\dfrac{x_2}{x_1} & 1 \end{bmatrix} = \alpha \mathbf{x}_{null} \quad \alpha \in \mathbb{R}.$$

At this point, combining elements of the null space, it is possible to build matrices whose right product with the vector $\mathbf{x}$ gives a vector of zeros:

$$\mathbf{A}_{null}\mathbf{x} = \begin{bmatrix} \alpha\mathbf{x}_{null} \\ \beta\mathbf{x}_{null} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} \quad \alpha, \beta \in \mathbb{R}.$$

Summing the parent factorization $\mathbf{A}_{parent}$ with the null matrix $\mathbf{A}_{null}$ gives another factorization of the initial system. Since the coefficients α and β can be chosen randomly, an infinite number of factorizations is readily obtained, for example:

$$\begin{Bmatrix} \alpha \\ \beta \end{Bmatrix} = \begin{Bmatrix} 1 \\ -1 \end{Bmatrix} \Rightarrow \mathbf{A}_{null}^{[1]} = \begin{bmatrix} -\dfrac{x_2}{x_1} & 1 \\ \dfrac{x_2}{x_1} & -1 \end{bmatrix}$$

$$\Downarrow$$

$$\mathbf{A}_1 = \mathbf{A}_{parent} + \mathbf{A}_{null}^{[1]} = \begin{bmatrix} -\dfrac{x_2}{x_1} & 2 \\ \dfrac{x_1 x_2 + 2x_2}{2x_1} & \dfrac{x_1 - 2}{2} \end{bmatrix}.$$

In general, for a problem with n states, the basis of the null space of the vector $\mathbf{x}$ consists of $n-1$ row vectors $\mathbf{v}_j \in \mathbb{R}^n$; a linear combination of these vectors still belongs to the null space, and so the rows of the $\mathbf{A}_{null}$ matrices can be built by linearly combining the basis:

$$\mathbf{A}_{null} = \begin{bmatrix} \alpha_{11}\mathbf{v}_1 + \alpha_{12}\mathbf{v}_2 + \cdots + \alpha_{1n-1}\mathbf{v}_{n-1} \\ \alpha_{21}\mathbf{v}_1 + \alpha_{22}\mathbf{v}_2 + \cdots + \alpha_{2n-1}\mathbf{v}_{n-1} \\ \vdots \\ \alpha_{n1}\mathbf{v}_1 + \alpha_{n2}\mathbf{v}_2 + \cdots + \alpha_{nn-1}\mathbf{v}_{n-1} \end{bmatrix}, \quad \alpha_{ij} \in \mathbb{R}. \tag{4.69}$$

In this chapter it has been studied how the ASRE algorithm can be improved, making it a more useful instrument. Still it would be interesting to evaluate the capabilities of this method by inspecting whether the solution obtained is optimal or rather sub-optimal. To do this it is possible to use the solution from the ASRE as a starting guess of a classical TPBVP solver; this issues will be faced in the next chapter.

Chapter 5

Refinement of the approximate solution

In the previous chapter it has been explained how the ASRE algorithm has been improved and how these improvements can be practically implemented. Starting from the nonlinear optimal control problem and some factorizations, the routine is able to provide a solution; since the ASRE method is an approximate one, its solution will be suboptimal. This approximation is the price paid to avoid guessing the initial Lagrange multiplier function, which is the typical limitation of a classical numerical solver for optimal control Boundary Value Problem (BVP). In a way, it is as if the role played by the choice of an initial guess for a BVP solver is transferred to the choice of the factorization for the ASRE algorithm, and in the previous chapter it has been proposed a way to carry out this choice in an automatized way. Still, it is desirable to obtain an optimal solution, at least in a numerical sense. To do this, the natural continuation of the routine is the refinement of the ASRE solution with the help of a classical BVP numerical solver, which is actually, from another point of view, the same as computing an optimal solution using as starting guess the solution of the ASRE algorithm.

In this chapter it will be first recalled the analytical form of the BVP stemming from the optimal control problem, then a brief overview of the classical numerical approaches to the BVP follows and lastly it is explained how to practically interface the ASRE routine with the BVP numerical solver.

5.1 Optimal control and BVPs

Given the dynamical system

$$\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x}, \mathbf{u}, t\right), \tag{5.1}$$

with proper boundary conditions, the cost function

$$J = \frac{1}{2}\varphi\left(\mathbf{x}_f, t_f\right) + \frac{1}{2}\int_{t_i}^{t_f} L\left(\mathbf{x}, \mathbf{u}, t\right) \mathrm{d}t, \tag{5.2}$$

and defining the Hamiltonian

$$H\left(\mathbf{x}, \mathbf{u}, t\right) = L\left(\mathbf{x}, \mathbf{u}, t\right) + \boldsymbol{\lambda}^T \mathbf{f}\left(\mathbf{x}, \mathbf{u}, t\right), \tag{5.3}$$

first-order necessary conditions for finding the control that minimizes the cost function while respecting the dynamical constraints are

$$\dot{\mathbf{x}} = \frac{\partial H}{\partial \boldsymbol{\lambda}}, \tag{5.4}$$

$$\dot{\boldsymbol{\lambda}} = -\frac{\partial H}{\partial \mathbf{x}}, \tag{5.5}$$

$$0 = \frac{\partial H}{\partial \mathbf{u}}, \tag{5.6}$$

subject to the set of boundary conditions completed by the *transversality condition*

$$\boldsymbol{\lambda}_f^T = \frac{\partial \varphi}{\partial \mathbf{x}_f}. \tag{5.7}$$

The algebraic equation (5.6) can be exploited to express the control $\mathbf{u}(t)$ as function of state and co-state, so that the optimal control problem becomes a boundary value problem of the form

$$\dot{\mathbf{z}} = \mathbf{F}(\mathbf{z}, t), \tag{5.8}$$

where $\mathbf{z}^T = \begin{Bmatrix} \mathbf{x} & \boldsymbol{\lambda} \end{Bmatrix}$ and

$$\mathbf{F} = \begin{Bmatrix} \dfrac{\partial H}{\partial \boldsymbol{\lambda}} \\ -\dfrac{\partial H}{\partial \mathbf{x}} \end{Bmatrix}, \tag{5.9}$$

along with the proper two-point boundary conditions

$$\mathbf{g}\left(\mathbf{z}(t_i), \mathbf{z}(t_f)\right) = 0. \tag{5.10}$$

It must be noticed that, in general, it is not always possible to obtain an explicit expression of the control $\mathbf{u}(t)$ as function of state $\mathbf{x}(t)$ and co-state $\boldsymbol{\lambda}(t)$, but, since it has been assumed that the dynamics are control-affine, it is sufficient to require a cost function that is a quadratic form of the control $\mathbf{u}(t)$, and from now on this assumption will hold.

In general, a given TPBVP can have no solution, a single solution, a finite number of solutions or even infinite solutions, but, as shown in section 2.3, it is always difficult to solve the problem analytically. For this reason, have been developed numerical procedures that solve the BVP; in the next section it will be presented an overview of the different kind of solvers and a more detailed explanation of the numerical solver used in this thesis.

5.2 Numerical approaches to the BVPs

According to [6], BVP numerical solver can be divided into four categories: *simple-shooting* methods, *multiple-shooting* methods, *differences* methods and *variational* methods.

5.2.1 Simple-shooting methods

The BVP is first treated as an Initial Value Problem (IVP), whose unique solution, under mild hypotheses, will depend on an unknown parameter, and then the value of the parameter is computed enforcing the final conditions, i.e., to solve the BVP

$$\begin{aligned} \dot{\mathbf{z}} &= \mathbf{f}\left(\mathbf{z}, t\right), \\ \mathbf{g}\left(\mathbf{z}(a), \mathbf{z}(b)\right) &= 0, \end{aligned} \tag{5.11}$$

first, the corresponding IVP

$$\begin{aligned} \dot{\mathbf{z}} &= \mathbf{f}\left(\mathbf{z}, t\right), \\ \mathbf{z}(a) &= \mathbf{s}, \end{aligned} \tag{5.12}$$

is processed, yielding a solution $\mathbf{z} = \mathbf{z}\left(t; \mathbf{s}\right)$, and then the final condition is enforced, finding the zeros $\bar{\mathbf{s}}_i$ of the algebraic equation

$$\mathbf{g}\left(\mathbf{z}(a; \mathbf{s}), \mathbf{z}(b; \mathbf{s})\right) \equiv \mathbf{g}\left(\mathbf{s}, \mathbf{z}(b; \mathbf{s})\right) = 0, \tag{5.13}$$

so that $\mathbf{z}\left(t; \bar{\mathbf{s}}_i\right)$ are solutions of the BVP.

5.2.2 Multiple-shooting methods

Simple-shooting methods can be not the best kind of solver for problems that are highly nonlinear, because the solution of the IVP will be very sensitive to the choice of the initial conditions, meaning that the integration over the domain will be unstable, especially for higly nonlinear dynamics. For this reason, multiple-shooting methods split the domain into smaller pieces, making it easier to integrate, and then it is enforced the respect of the two-point boundary conditions along with the continuity of the solution over the complete domain.

5.2.3 Difference methods

Difference methods basically consist in replacing the differential quotients with difference quotients and solving the discrete equations that are so obtained. For example, to solve the BVP

$$\begin{aligned} \dot{z} &= f(t), \\ z(a) = \alpha, &\quad z(b) = \beta, \end{aligned} \tag{5.14}$$

first, the domain $[a, b]$ is subdivided into $n + 1$ equal subintervals

$$a = t_0 < t_1 < \cdots < t_n < t_{n+1} = b, \quad t_j = a + jh, \quad h := \frac{b-a}{n+1}, \tag{5.15}$$

then, employing the central difference formula

$$\Delta z_i = \frac{z_{i+1} - z_{i-1}}{2h}, \tag{5.16}$$

the discretized approximation $z_i \approx z(t_i)$ of the BVP solution must satisfy the following equations:

$$\begin{cases} z_0 = \alpha, \\ \dfrac{z_{i+1} - z_{i-1}}{2h} = f(t_i), \\ z_{n+1} = \beta. \end{cases} \tag{5.17}$$

This structure can be rearranged as

$$\mathbf{A}\tilde{\mathbf{z}} = \mathbf{b}, \tag{5.18}$$

where $\mathbf{A}$ is the $n \times n$ tridiagonal matrix

$$\mathbf{A} = \frac{1}{2h} \begin{bmatrix} 0 & 1 & & & 0 \\ -1 & 0 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 0 & 1 \\ 0 & & & -1 & 0 \end{bmatrix}, \tag{5.19}$$

while $\tilde{\mathbf{z}}$ and $\mathbf{b}$ are $\mathbb{R}^n$ vectors

$$\tilde{\mathbf{z}} = \begin{Bmatrix} z_1 \\ z_2 \\ \vdots \\ z_{n-1} \\ z_n \end{Bmatrix}, \qquad \mathbf{b} = \begin{Bmatrix} f(t_1) + \dfrac{\alpha}{2h} \\ f(t_2) \\ \vdots \\ f(t_{n-1}) \\ f(t_n) - \dfrac{\beta}{2h} \end{Bmatrix}. \tag{5.20}$$

Solving the linear system allows to find an approximation of the BVP solution. For those problems where the right-hand side of differential equation (5.14) is not only function of t but also of z, it is usually possible to take into account the additional z_i's by suitably modifying matrix $\mathbf{A}$, using it not only for the terms stemming from the difference quotients but also for the terms coming from function $f(z, t)$, so that the problem it is again reduced to solving a linear system of equations.

5.2.4 Variational methods

Variational methods rely, in general, on approximating the solution of a BVP by finding the function y_s that minimizes a quadratic form $F(y)$; y_s belongs to the subspace of all the functions that are null at the boundaries of the domain $[a, b]$. The minimization occurs by choosing a proper basis of the subspace. *Collocation methods* are a particular set of variational methods: having chosen a proper basis and a set of collocation points, it is enforced the satisfaction of the differential equations on the collocation points.

The solver employed to refine the ASRE solution is the collocation method presented in [29], that exactly satisfies the differential equation

$$\mathbf{y}' = \mathbf{f}\left(x, \mathbf{y}\right), \tag{5.21}$$

at both ends and midpoint of each interval $[x_j, x_{j+1}]$ of the mesh

$$a = x_0 < x_1 < \cdots < x_n < x_{n+1} = b, \tag{5.22}$$

using cubic polynomials as basis, so that the approximate solution is a continuous function $\mathbf{S}(x)$, whose coefficients are found by solving the system of algebraic, nonlinear equations

$$\begin{gathered} \mathbf{S}'(x_j) = \mathbf{f}\left(x_j, \mathbf{S}(x_j)\right), \\ \mathbf{S}'\left(\frac{x_j + x_{j+1}}{2}\right) = \mathbf{f}\left(\frac{x_j + x_{j+1}}{2}, \mathbf{S}\left(\frac{x_j + x_{j+1}}{2}\right)\right), \\ \mathbf{S}'(x_{j+1}) = \mathbf{f}\left(x_{j+1}, \mathbf{S}(x_{j+1})\right), \end{gathered} \tag{5.23}$$

on every interval of the mesh. As said in [29], this method is a fourth-order approximation of an isolated solution $\mathbf{y}(x)$, i.e.,

$$\|\mathbf{y}(x) - \mathbf{S}(x)\| \leq Ch^4, \tag{5.24}$$

where C is a constant and h is the unit step of the mesh. As any other classical BVP solver, this collocation method requires a good starting guess in order to initialize the solution of the algebraic system of equations (5.23), that for the case in exam comes from the solution of the ASRE. In the next section it will be briefly presented how to cast the optimal control problem definition and the solution of the ASRE into suitable forms.

5.3 Interface between ASRE and BVP solver

The approximate solution of the optimal control problem coming from the ASRE routine is expressed as functions evaluated at discrete locations of a time grid that counts $N - 1$ intervals; those functions are state $\mathbf{x}_{ASRE}$,

co-state $\boldsymbol{\lambda}_{ASRE}$ and control $\mathbf{u}_{ASRE}$. The BVP associated to the control problem is

$$\begin{Bmatrix} \dot{\mathbf{x}} \\ \dot{\boldsymbol{\lambda}} \end{Bmatrix} = \begin{Bmatrix} \dfrac{\partial H}{\partial \boldsymbol{\lambda}} \\ -\dfrac{\partial H}{\partial \mathbf{x}} \end{Bmatrix}, \tag{5.25}$$

so that, in order to initialize the BVP solver, first it is necessary to provide the vector field of the problem, then the boundary conditions and lastly the starting guess, expressed as a time-mesh and a vector containing the values of the initial guess at each point of the mesh. The first two tasks are accomplished by resorting to a symbolic processing of the optimal control data, while the provision of an initial guess simply consists in the time grid adopted by the ASRE algorithm and the vector that results from concatenating the ASRE solution of state and co-state

$$\begin{Bmatrix} \dot{\mathbf{x}} \\ \dot{\boldsymbol{\lambda}} \end{Bmatrix} = \begin{Bmatrix} \dfrac{\partial H}{\partial \boldsymbol{\lambda}} \\ -\dfrac{\partial H}{\partial \mathbf{x}} \end{Bmatrix} \quad \Rightarrow \quad \dot{\mathbf{z}} = \mathbf{F}(z), \tag{5.26}$$

$$\begin{cases} \mathbf{x}(t_i) = \mathbf{x}_i \\ \mathbf{x}(t_f) = \mathbf{x}_f \\ \boldsymbol{\lambda}^T(t_f) = \dfrac{\partial \varphi}{\partial \mathbf{x}_f} \end{cases} \quad \Rightarrow \quad \mathbf{g}\left(\mathbf{z}(t_i), \mathbf{z}(t_f)\right) = 0, \tag{5.27}$$

$$\begin{Bmatrix} \mathbf{x}_{ASRE} \\ \boldsymbol{\lambda}_{ASRE} \end{Bmatrix} \quad \Rightarrow \quad \mathbf{z}_{guess}. \tag{5.28}$$

In order to improve the convergence of the BVP solver, it is possible to provide the analytical forms of the Jacobians of both vector field and boundary conditions. To do this, again, it is possible to exploit symbolic manipulations.

5.4 Work-flow of the complete solver

The previous section allows to link the ASRE algorithm with the BVP algorithm, producing a complete solver able to compute the optimal state, co-state and control histories. In Figure 5.1 it is shown the flowchart of this complete solver.

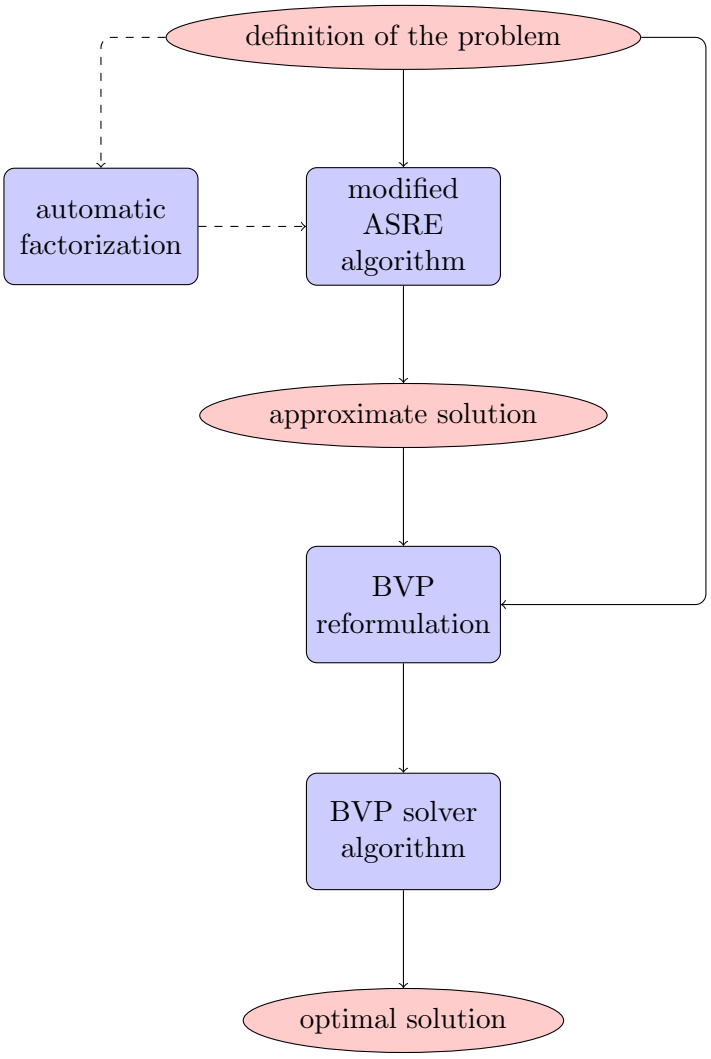


Figure 5.1: Work-flow of the complete solver

Chapter 6

Astrodynamics applications and other examples

In this chapter it will be presented a selection of examples that show how the modified ASRE algorithm along with the BVP solver can deal with a number of problems, yielding satisfying results. First it will be solved a couple of simple benchmark problems, then a few examples of astrodynamics related optimal control problems will be analysed and solved in detail.

6.1 Benchmark problems

The first two problems studied have no specific physical meaning, but are handy to grasp the basics of the solver presented. The first problem has been already introduced in section 4.2.1, while the second example is a slightly more nonlinear problem.

6.1.1 Benchmark problem 1

Given the dynamic system

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= x_1 x_2 + u, \end{aligned} \tag{6.1}$$

with boundary conditions

$$\begin{aligned} \mathbf{x}(t_i) &= \begin{Bmatrix} 2 \\ 1 \end{Bmatrix}, \quad t_i = 0, \\ \mathbf{x}(t_f) &= \begin{Bmatrix} 4 \\ 1 \end{Bmatrix}, \quad t_f = 4, \end{aligned}$$

it is required to find the control $u(t)$ that minimizes the cost function

$$J = \frac{1}{2} \int_{t_i}^{t_f} \left[x_1^2 + x_2^2 + u^2 \right] \mathrm{d}t. \tag{6.2}$$

The factorized form is

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\ \mathbf{x} + \mathbf{B}\ \mathbf{u}, \\ J &= \frac{1}{2}\int_{t_i}^{t_f} \left[\mathbf{x}^T\mathbf{Q}\ \mathbf{x} + \mathbf{u}^T\mathbf{R}\ \mathbf{u}\right] \mathrm{d}t, \end{aligned} \tag{6.3}$$

with

$$\mathbf{B} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \mathbf{Q} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{R} = \begin{bmatrix} 1 \end{bmatrix},$$

and the algorithm has been provided with four different factorizations of the uncontrolled dynamics

$$\begin{aligned} \mathbf{A}_1(\mathbf{x}) &= \begin{bmatrix} 0 & 1 \\ 0 & x_1 \end{bmatrix}, \quad \mathbf{A}_2(\mathbf{x}) = \begin{bmatrix} 0 & 1 \\ x_2 & 0 \end{bmatrix}, \\ \mathbf{A}_3(\mathbf{x}) &= \begin{bmatrix} x_2 & 1-x_1 \\ 0 & x_1 \end{bmatrix}, \quad \mathbf{A}_4(\mathbf{x}) = \begin{bmatrix} x_2 & 1-x_1 \\ x_2 & 0 \end{bmatrix}. \end{aligned}$$

It must be noticed that all the four factorizations are consistent with the initial boundary conditions

$$\begin{aligned} \mathbf{A}_1(\mathbf{x}_i) &= \begin{bmatrix} 0 & 1 \\ 0 & 2 \end{bmatrix}, \quad \mathbf{A}_2(\mathbf{x}_i) = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \\ \mathbf{A}_3(\mathbf{x}_i) &= \begin{bmatrix} 1 & -1 \\ 0 & 2 \end{bmatrix}, \quad \mathbf{A}_4(\mathbf{x}_i) = \begin{bmatrix} 1 & -1 \\ 1 & 0 \end{bmatrix}, \end{aligned}$$

but this is not true for what concerns controllability. In fact, evaluating the controllability matrices of the four factorizations

$$\begin{aligned} \mathbf{K}_{c1}(\mathbf{x}) &= \begin{bmatrix} 0 & 1 \\ 1 & x_1 \end{bmatrix}, \quad \mathbf{K}_{c2}(\mathbf{x}) = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \\ \mathbf{K}_{c3}(\mathbf{x}) &= \begin{bmatrix} 0 & 1-x_1 \\ 1 & x_1 \end{bmatrix}, \quad \mathbf{K}_{c4}(\mathbf{x}) = \begin{bmatrix} 0 & 1-x_1 \\ 1 & 0 \end{bmatrix}, \end{aligned}$$

it is clear that the ranks of $\mathbf{K}_{c1}(\mathbf{x})$ and $\mathbf{K}_{c2}(\mathbf{x})$ are always equal to the number of states

$$\begin{aligned} \det\left[\mathbf{K}_{c1}(\mathbf{x})\right] &= -1 \quad \forall \mathbf{x}, \\ \det\left[\mathbf{K}_{c2}(\mathbf{x})\right] &= -1 \quad \forall \mathbf{x}, \end{aligned}$$

while both $\mathbf{K}_{c3}(\mathbf{x})$ and $\mathbf{K}_{c4}(\mathbf{x})$ become singular when $x_1 = 1$

$$\begin{aligned} \det\left[\mathbf{K}_{c3}(\mathbf{x})\right] &= x_1 - 1, \\ \det\left[\mathbf{K}_{c4}(\mathbf{x})\right] &= x_1 - 1. \end{aligned}$$

It is interesting to notice that the factorizations that are not always controllable, $\mathbf{A}_3(\mathbf{x})$ and $\mathbf{A}_4(\mathbf{x})$, are not a "natural" representation of the uncontrolled

Table 6.1: Iterations of the MASRE algorithm for the first benchmark problem, computational time 40.0 s with an Intel Core2 Quad CPU 2.50 GHz

	Error	σ_{min}	J	Matrix
1	2.000	2.082	2.436×10^{1}	$\mathbf{A}_1$
2	7.173×10^{-1}	1.578×10^{6}	2.505×10^{1}	$\mathbf{A}_1$
3	1.412×10^{-1}	1.130×10^{7}	2.606×10^{1}	$\mathbf{A}_1$
4	1.294×10^{-2}	1.960×10^{7}	2.613×10^{1}	$\mathbf{A}_1$
5	1.648×10^{-3}	2.009×10^{7}	2.612×10^{1}	$\mathbf{A}_1$
6	1.767×10^{-4}	1.997×10^{7}	2.612×10^{1}	$\mathbf{A}_1$
7	1.893×10^{-5}	1.996×10^{7}	2.612×10^{1}	$\mathbf{A}_1$
8	2.259×10^{-6}	1.996×10^{7}	2.612×10^{1}	$\mathbf{A}_1$
9	2.083×10^{-7}	1.996×10^{7}	2.612×10^{1}	$\mathbf{A}_1$

dynamics, i.e., while the nonlinear system is obtained by simply multiplying $\mathbf{A}_1(\mathbf{x})$ and $\mathbf{A}_2(\mathbf{x})$ with the state vector $\mathbf{x}$, $\mathbf{A}_3(\mathbf{x})$ and $\mathbf{A}_4(\mathbf{x})$ rely on some sort of addition and subtraction of the same term. The algorithm is then initialized, having set the value of the termination tolerance to 10^{-6}.

The algorithm is able to detect the non-controllability of factorizations $\mathbf{A}_3(\mathbf{x})$ and $\mathbf{A}_4(\mathbf{x})$, and so it discards them. Then, the MASRE routine starts and its sequence of iterations it is shown in Table 6.1, while the final optimal cost function is

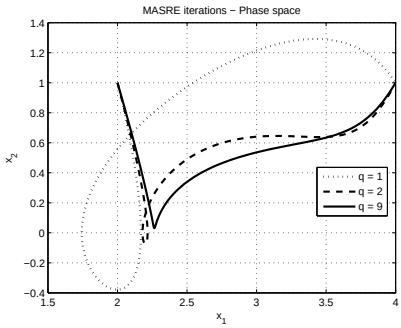
$$J_{opt} = 2.606 \times 10^{1}.$$

In Figure 6.1a it is shown the sequence of iterations made by the MASRE algorithm: the optimal matrix turns out to be $\mathbf{A}_1(\mathbf{x})$, and this, remembering the results obtained in section 4.2.1 by solving the problem with the original ASRE routine, confirms that the algorithm is able to detect the best factorization in terms of controllability and thus cost function. The same pattern can obviously be observed, in Figure 6.1b, by looking at the control input stemming from the approximate solutions of the MASRE algorithm. Lastly, it is important to underline that the final solution of the MASRE algorithm is very close to the actual optimal solution found with the BVP solver, meaning that only a refinement occurs, rather than a completely different solution; still, the refinement makes clear that the approximate solution is not optimal, but rather sub-optimal. Figures 6.1c and 6.1d clearly illustrate this refinement in terms of both phase-space solution and control.

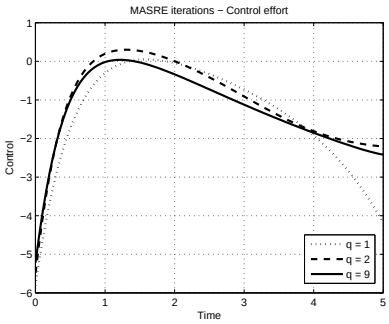
6.1.2 Benchmark problem 2

Given the dynamic system

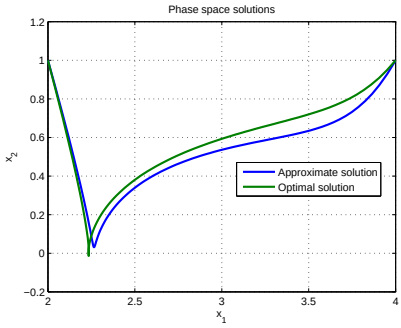
$$\begin{aligned} \dot{x}_1 &= x_1^2 x_2 + x_2, \\ \dot{x}_2 &= x_1 x_2 + u, \end{aligned} \tag{6.4}$$



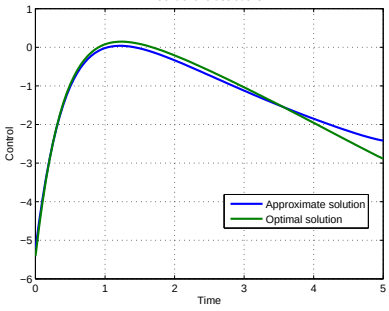
(a) MASRE phase-space iterations



(b) MASRE control iterations



(c) Phase-space solutions



(d) Solutions for the control

Figure 6.1: Solution of the first benchmark problem

with boundary conditions

$$\mathbf{x}(t_i) = \begin{Bmatrix} 2 \\ 1 \end{Bmatrix}, \quad t_i = 0,$$
$$\mathbf{x}(t_f) = \begin{Bmatrix} 4 \\ 1 \end{Bmatrix}, \quad t_f = 5,$$

it is required to find the control $u(t)$ that minimizes the cost function

$$J = \frac{1}{2} \int_{t_i}^{t_f} \left[x_1^2 + x_2^2 + u^2 \right] \mathrm{d}t. \tag{6.5}$$

The factorized form is again that of equation (6.3) with

$$\mathbf{B} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \mathbf{Q} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{R} = \begin{bmatrix} 1 \end{bmatrix},$$

and the algorithm has been provided with four different factorizations of the uncontrolled dynamics

$$\mathbf{A}_1(\mathbf{x}) = \begin{bmatrix} x_1 x_2 & 1 \\ 0 & x_1 \end{bmatrix}, \quad \mathbf{A}_2(\mathbf{x}) = \begin{bmatrix} 0 & x_1^2 + 1 \\ 0 & x_1 \end{bmatrix},$$
$$\mathbf{A}_3(\mathbf{x}) = \begin{bmatrix} x_1 x_2 & 1 \\ x_2 & 0 \end{bmatrix}, \quad \mathbf{A}_4(\mathbf{x}) = \begin{bmatrix} 0 & x_1^2 + 1 \\ x_2 & 0 \end{bmatrix}.$$

All of the four factorization are both consistent with the initial boundary conditions

$$\mathbf{A}_1(\mathbf{x}_i) = \begin{bmatrix} 2 & 1 \\ 0 & 2 \end{bmatrix}, \quad \mathbf{A}_2(\mathbf{x}_i) = \begin{bmatrix} 0 & 5 \\ 0 & 2 \end{bmatrix},$$
$$\mathbf{A}_3(\mathbf{x}_i) = \begin{bmatrix} 2 & 1 \\ 1 & 0 \end{bmatrix}, \quad \mathbf{A}_4(\mathbf{x}_i) = \begin{bmatrix} 0 & 5 \\ 1 & 0 \end{bmatrix},$$

and always controllable

$$\mathbf{K}_{c1}(\mathbf{x}) = \begin{bmatrix} 0 & 1 \\ 1 & x_1 \end{bmatrix} \quad \Rightarrow \quad \det\left[\mathbf{K}_{c1}(\mathbf{x})\right] = -1 \quad \forall \mathbf{x},$$
$$\mathbf{K}_{c2}(\mathbf{x}) = \begin{bmatrix} 0 & x_1^2 + 1 \\ 1 & x_1 \end{bmatrix} \quad \Rightarrow \quad \det\left[\mathbf{K}_{c2}(\mathbf{x})\right] = -x_1^2 - 1 \leq -1 \quad \forall \mathbf{x},$$
$$\mathbf{K}_{c3}(\mathbf{x}) = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad \Rightarrow \quad \det\left[\mathbf{K}_{c3}(\mathbf{x})\right] = -1 \quad \forall \mathbf{x},$$
$$\mathbf{K}_{c4}(\mathbf{x}) = \begin{bmatrix} 0 & x_1^2 + 1 \\ 1 & 0 \end{bmatrix} \quad \Rightarrow \quad \det\left[\mathbf{K}_{c4}(\mathbf{x})\right] = -x_1^2 - 1 \leq -1 \quad \forall \mathbf{x}.$$

Differently from the previous problem, where two factorizations were evidently not "natural" representations of the uncontrolled dynamics, this problem is

Table 6.2: Iterations of the MASRE algorithm for the second benchmark problem, CPU time 231.9 s

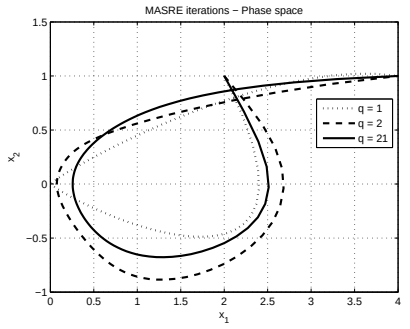
	Error	σ_{min}	J	Matrix
1	2.000	1.739×10^{7}	1.444×10^{1}	$\mathbf{A}_4$
2	1.376	2.978×10^{3}	7.876	$\mathbf{A}_4$
3	1.121	1.355×10^{4}	1.086×10^{1}	$\mathbf{A}_2$
4	7.010×10^{-1}	1.615×10^{4}	9.327	$\mathbf{A}_2$
5	2.423×10^{-1}	1.313×10^{4}	9.798	$\mathbf{A}_2$
6	9.391×10^{-2}	1.480×10^{4}	9.767	$\mathbf{A}_2$
7	6.236×10^{-2}	1.495×10^{4}	9.643	$\mathbf{A}_2$
8	2.657×10^{-2}	1.434×10^{4}	9.708	$\mathbf{A}_2$
9	9.479×10^{-3}	1.443×10^{4}	9.710	$\mathbf{A}_2$
10	6.504×10^{-3}	1.452×10^{4}	9.696	$\mathbf{A}_2$
11	2.838×10^{-3}	1.449×10^{4}	9.701	$\mathbf{A}_2$
12	9.442×10^{-4}	1.449×10^{4}	9.701	$\mathbf{A}_2$
13	6.624×10^{-4}	1.449×10^{4}	9.701	$\mathbf{A}_2$
14	3.024×10^{-4}	1.449×10^{4}	9.701	$\mathbf{A}_2$
15	9.275×10^{-5}	1.449×10^{4}	9.701	$\mathbf{A}_2$
16	6.779×10^{-5}	1.449×10^{4}	9.701	$\mathbf{A}_2$
17	3.273×10^{-5}	1.449×10^{4}	9.701	$\mathbf{A}_2$
18	9.125×10^{-6}	1.449×10^{4}	9.701	$\mathbf{A}_2$
19	6.971×10^{-6}	1.449×10^{4}	9.701	$\mathbf{A}_2$
20	3.465×10^{-6}	1.449×10^{4}	9.701	$\mathbf{A}_2$
21	9.011×10^{-7}	1.449×10^{4}	9.701	$\mathbf{A}_2$

expressed naturally with four factorizations, so that one would not know *a priori* which factorizations should be used with the original ASRE algorithm. As before, the algorithm is initialized having set the termination tolerance to 10^{-6}.

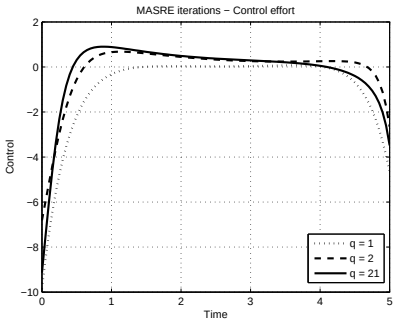
The iterations of the MASRE algorithm are shown in Table 6.2, and after the refinement operated with the BVP solver, the optimal value of the cost function is

$$J_{opt} = 9.413.$$

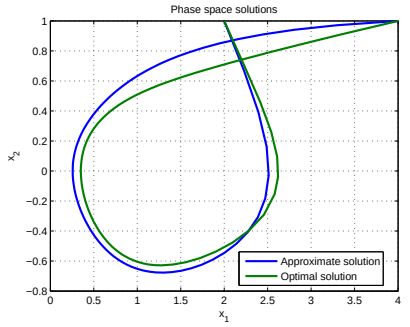
The graphical representations of the approximate and optimal solutions are presented as before in Figure 6.2; as before, the consistent refinement shows that the approximate solution is not optimal but only sub-optimal. In order to test the quality of this solution, the problem has been solved with the classical ASRE algorithm, employing the four different factorizations separately, and it has been confirmed that the the best solution is the one related to factorization $\mathbf{A}_2(\mathbf{x})$; not only, because, even though all of the four factorizations are consistent with the initial boundary conditions and always



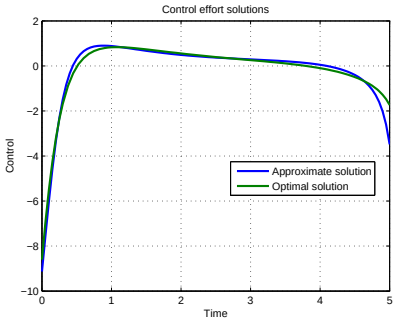
(a) MASRE phase-space iterations



(b) MASRE control iterations



(c) Phase-space solutions



(d) Solutions for the control

Figure 6.2: Solution of the second benchmark problem

controllable, the classical ASRE solver is not able to provide a solution using $\mathbf{A}_1(\mathbf{x})$ or $\mathbf{A}_3(\mathbf{x})$: the former factorization yields a sequence of iterations whose error stably oscillates, without converging to any solution, while the latter is not even able to continue the routine since transition matrix becomes singular after a certain number of iterations. First of all, those results imply that controllability and consistency are at most *necessary conditions* for a converging factorization, but the above discussion also stresses the fact that the optimization of the controllability seems to allow to dodge those factorizations that would not lead to any solution.

After having tested the MASRE algorithm with the previous benchmark problem, in the next sections it will be shown how the MASRE routine can be effective when dealing with physical control problems, in particular those that refer to the astrodynamics issues.

6.2 Rendez-vous

A space rendez-vous is an orbital manoeuvre that aims at having two objects, i.e., spacecraft, namely a "target" and a "chaser", at the same location, at the same time and with the same velocity. Historically, the first documented attempts of putting two objects on the same orbit were made by Russians, who launched pairs of spacecraft from the same launchpad one or two days apart (*Vostok 3* and *4* in 1962 and *Vostok 5* and *6* in 1963), but the objects could not achieve a rendez-vous, since the orbit adjustments were very rough. By 1965, US astronaut Jim McDivitt attempted to manoeuvre his *Gemini 4* craft to meet the spent *Titan II* upper stage, but, even though the capsule was equipped with the necessary actuators, the target was not approached since the underling orbital mechanics' principles were not clear enough. Only a few months later, US astronaut Wally Schirra was able to manoeuvre his *Gemini 6* within 30 cm of its sister craft *Gemini 7*.

The main difficulties related to rendez-vous are the strong nonlinearities of the dynamics, along with the strict requirements in terms of precision. For this reason, it is interesting to study whether the MASRE algorithm can deal with such a control problem.

6.2.1 Derivation of the dynamics

As stated by Newton in his *Philosophiæ Naturalis Principia Mathematica*, every point mass attracts any other point mass with a force that is proportional to the product of the two masses and inversely proportional to the square of the distance between the two masses; this force acts along the direction that connects the two masses, e.g., being the two masses m_1 and m_2 and their positions inside an inertial frame $\mathbf{R}_1$ and $\mathbf{R}_2$, the position vector $\mathbf{r}$ of m_2 relative to m_1 will be

$$\mathbf{r} = \mathbf{R}_2 - \mathbf{R}_1, \tag{6.6}$$

that can be rewritten as the product of the magnitude $r = \|\mathbf{r}\|$ with the unit vector pointing from m_1 towards m_2

$$\hat{\mathbf{i}}_r = \frac{\mathbf{r}}{r}. \tag{6.7}$$

Then, the force exerted on m_2 by m_1 is

$$\mathbf{F}_{21} = \frac{Gm_1m_2}{r^2}\left(-\hat{\mathbf{i}}_r\right) = -\frac{Gm_1m_2}{r^2}\hat{\mathbf{i}}_r, \tag{6.8}$$

and subsequently, remembering Newton's third law of motion,

$$\mathbf{F}_{12} = -\mathbf{F}_{21} = \frac{Gm_1m_2}{r^2}\hat{\mathbf{i}}_r. \tag{6.9}$$

Finally, introducing Newton's second law of motion, it is possible to write the equations of motion for the two masses

$$m_1\ddot{\mathbf{R}}_1 = \mathbf{F}_{12} = \frac{Gm_1m_2}{r^2}\hat{\mathbf{i}}_r, \tag{6.10}$$

$$m_2\ddot{\mathbf{R}}_2 = \mathbf{F}_{21} = -\frac{Gm_1m_2}{r^2}\hat{\mathbf{i}}_r. \tag{6.11}$$

Let us now multiply both sides of equation (6.10) by m_2 and both sides of equation (6.11) by m_1

$$m_1m_2\ddot{\mathbf{R}}_1 = \frac{Gm_1m_2^2}{r^2}\hat{\mathbf{i}}_r, \tag{6.12}$$

$$m_1m_2\ddot{\mathbf{R}}_2 = -\frac{Gm_1^2m_2}{r^2}\hat{\mathbf{i}}_r, \tag{6.13}$$

and subtract equation (6.12) from equation (6.13)

$$m_1m_2\left(\ddot{\mathbf{R}}_2 - \ddot{\mathbf{R}}_1\right) = -\frac{Gm_1m_2\left(m_1+m_2\right)}{r^2}\hat{\mathbf{i}}_r. \tag{6.14}$$

Dividing by m_1m_2 and enforcing equation (6.6) and equation (6.7) yields

$$\ddot{\mathbf{r}} = -\frac{\mu}{r^3}\mathbf{r}, \tag{6.15}$$

where μ is the gravitational parameter defined as

$$\mu = G\left(m_1+m_2\right). \tag{6.16}$$

Equation (6.15) governs the relative motion of two bodies only acted upon by the mutual gravitational attraction. By introducing a control acceleration, the dynamical system is complete

$$\ddot{\mathbf{r}} = -\frac{\mu}{r^3}\mathbf{r} + \mathbf{u}. \tag{6.17}$$

In order to solve a rendez-vous related optimal control problem, which typically involves very small variations of the position vector expressed in a Sun-centered polar reference, it is advisable to cast equation (6.17) in a different form, shifting to a non-inertial reference frame that is rotating at constant speed on a circular orbit around the origin of the inertial reference frame. The origin of this new reference frame represents the "target" spacecraft, to which the "chasing" spacecraft is approaching. To do this, let us assume that the new reference frame's origin is located at a distance $\mathbf{R}$ from the origin of the inertial reference frame, and it is rotating at a constant angular velocity $\boldsymbol{\omega}$. The new reference frame's unit vectors are defined as follows:

$\hat{\mathbf{i}}$ is directed along the radial direction;

$\hat{\mathbf{j}}$ is directed along the transversal direction;

$\hat{\mathbf{k}}$ is directed along the normal to the orbital plane.

According to this new reference frame, its position vector from the origin of the inertial reference frame will be $\mathbf{R} = R\hat{\mathbf{i}}$, its angular velocity will be $\boldsymbol{\omega} = \omega\hat{\mathbf{k}}$ and the position of the spacecraft from the origin of the rotating reference frame will be $\delta\mathbf{r} = [x\ y\ z]^T$. It follows that the distance of the spacecraft from the origin of the inertial reference frame is expressed as

$$\begin{aligned}\mathbf{r} &= \mathbf{R} + \delta\mathbf{r} \\ &= R\hat{\mathbf{i}} + x\hat{\mathbf{i}} + y\hat{\mathbf{j}} + z\hat{\mathbf{k}}, \\ &= (R + x)\,\hat{\mathbf{i}} + y\hat{\mathbf{j}} + z\hat{\mathbf{k}}\end{aligned} \tag{6.18}$$

and the derivatives of this position vector are

$$\dot{\mathbf{r}} = \omega R\hat{\mathbf{j}} + \dot{x}\hat{\mathbf{i}} + \omega x\hat{\mathbf{j}} + \dot{y}\hat{\mathbf{j}} - \omega y\hat{\mathbf{i}} + \dot{z}\hat{\mathbf{k}}, \tag{6.19}$$

$$\begin{aligned}\ddot{\mathbf{r}} &= -\omega^2 R\hat{\mathbf{i}} + \ddot{x}\hat{\mathbf{i}} + \omega\dot{x}\hat{\mathbf{j}} + \omega\dot{x}\hat{\mathbf{j}} - \omega^2 x\hat{\mathbf{i}} + \ddot{y}\hat{\mathbf{j}} - \omega\dot{y}\hat{\mathbf{i}} - \omega\dot{y}\hat{\mathbf{i}} - \omega^2 y\hat{\mathbf{j}} + \ddot{z}\hat{\mathbf{k}}, \\ &= \left[\ddot{x} - 2\omega\dot{y} - \omega^2\,(R + x)\right]\hat{\mathbf{i}} + \left[\ddot{y} + 2\omega\dot{x} - \omega^2 y\right]\hat{\mathbf{j}} + \ddot{z}\hat{\mathbf{k}}.\end{aligned} \tag{6.20}$$

Substituting equations (6.18) and (6.20) inside equation (6.15), it is possible to write the following expression

$$\ddot{x} - 2\omega\dot{y} - \omega^2\,(R + x) = -\frac{\mu}{r^3}\,(R + x) + u_x, \tag{6.21}$$

$$\ddot{y} + 2\omega\dot{x} - \omega^2 y = -\frac{\mu}{r^3}y + u_y, \tag{6.22}$$

$$\ddot{z} = -\frac{\mu}{r^3}z + u_z, \tag{6.23}$$

where r is the scalar distance of the spacecraft from the origin of the inertial reference frame, defined as

$$r = \sqrt{(R + x)^2 + y^2 + z^2}, \tag{6.24}$$

and u_x, u_y and u_z are the components of the control acceleration along the three directions. The differential system (6.21)-(6.22)-(6.23) describes the deviations of the spacecraft from the origin of the rotating reference frame, and it is a natural way of expressing the underlying dynamics of a rendez-vous. In the following it will be neglected the component of the dynamics transversal to the orbital plane represented by (6.23), by assuming that the initial position and velocity have no transversal component, so that, defining $x_1 = x$, $x_2 = y$, $x_3 = \dot{x}$, $x_4 = \dot{y}$, $u_1 = u_x$ and $u_2 = u_y$ the resulting

state-space form of the system will be

$$\begin{cases} \dot{x}_1 = x_3, \\ \dot{x}_2 = x_4, \\ \dot{x}_3 = 2\omega x_4 + \left(\omega^2 - \dfrac{\mu}{r^3}\right)(R + x_1) + u_1, \\ \dot{x}_4 = -2\omega x_3 + \left(\omega^2 - \dfrac{\mu}{r^3}\right) x_2 + u_2, \end{cases} \tag{6.25}$$

and also taking into account the algebraic equation (6.24), modified to take into account the lack of transversal component

$$r = \sqrt{(R + x_1)^2 + x_2^2}. \tag{6.26}$$

6.2.2 Problem set-up

The following problem is taken from [28], where it has been solved in its HCP formulation relying on generating functions, and the results obtained there will be used to verify the effectiveness of the MASRE solution. The problem considered can be simplified by employing as unit length the radius R, as unit time $1/\omega$ and a unit mass that makes $\mu = 1$, so that the system can be rewritten as

$$\begin{cases} \dot{x}_1 = x_3, \\ \dot{x}_2 = x_4, \\ \dot{x}_3 = 2x_4 + \left(1 - \dfrac{1}{r^3}\right)(1 + x_1) + u_1, \\ \dot{x}_4 = -2x_3 + \left(1 - \dfrac{1}{r^3}\right) x_2 + u_2, \end{cases} \tag{6.27}$$

The cost function to be minimized is

$$J = \frac{1}{2} \int_{t_i}^{t_f} \mathbf{u}^T \mathbf{u} \mathrm{d}t, \tag{6.28}$$

and the boundary conditions applied are

$$\left.\begin{array}{l} x_1 = 0.2, \\ x_2 = 0.2, \\ x_3 = 0.1, \\ x_4 = 0.1, \end{array}\right\}_{t=0} \qquad \left.\begin{array}{l} x_1 = 0, \\ x_2 = 0, \\ x_3 = 0, \\ x_4 = 0, \end{array}\right\}_{t=1}$$

that define a HCP; leaving the final conditions unspecified and defining a weighting matrix $\mathbf{S}(\mathbf{x}_f)$ of the final state

$$\mathbf{S}(\mathbf{x}_f) = \begin{bmatrix} 25 & 0 & 0 & 0 \\ 0 & 15 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix}, \tag{6.29}$$

on the other hand, yields a SCP. The differential equations (6.27) have been first factorized in the most natural form

$$\begin{Bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{Bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \left(1-\frac{1}{r^3}\right)\left(1+\frac{1}{x_1}\right) & 0 & 0 & 2 \\ 0 & \left(1-\frac{1}{r^3}\right) & -2 & 0 \end{bmatrix}}_{\mathbf{A}_1(\mathbf{x})} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{\mathbf{B}} \begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix}, \tag{6.30}$$

while the cost function has been rewritten in factorized form by setting $\mathbf{Q} = \mathbf{0}_{4\times4}$ and $\mathbf{R} = \mathbf{I}_{2\times2}$. In order to test the ability of the MASRE algorithm of selecting the best factorization, the uncontrolled dynamics have been manipulated to obtain more factorizations. The term

$$\left(1-\frac{1}{r^3}\right)(1+x_1) \tag{6.31}$$

can be rewritten as

$$\left(1-\frac{1}{r^3}\right)(1+x_1) = \left(1-\frac{1}{r^3}\right)x_1 + \left(1-\frac{1}{r^3}\right), \tag{6.32}$$

and the first term of the RHS is easily expressed as

$$\left(1-\frac{1}{r^3}\right)x_1 = \left\{1-\frac{1}{r^3} \quad 0 \quad 0 \quad 0\right\} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix}. \tag{6.33}$$

The second term of the RHS can be decomposed applying the equality

$$\left(a^3-b^3\right) = (a-b)\left(a^2+ab+b^2\right), \tag{6.34}$$

so that

$$1-\frac{1}{r^3} = \frac{r^3-1}{r^3} = \frac{r-1}{r^3}\left(r^2+r+1\right). \tag{6.35}$$

Recalling equation (6.26)

$$\begin{aligned} \frac{r-1}{r^3}\left(r^2+r+1\right) &= \frac{r-1}{r^3}\left[(1+x_1)^2+x_2^2+r+1\right] \\ &= \frac{r-1}{r^3}\left[x_1^2+2x_1+x_2^2+r+2\right], \end{aligned} \tag{6.36}$$

so that it is possible to factorize the uncontrolled dynamics as

$$\mathbf{A}_2(\mathbf{x}) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ c_1 & \dfrac{r-1}{r^3}x_2 & 0 & 2 \\ 0 & \dfrac{r^3-1}{r^3} & -2 & 0 \end{bmatrix},$$

where

$$c_1 = \frac{r-1}{r^3}\left[x_1 + 2 + \frac{1}{x_1}(r+2)\right] + \frac{r^3-1}{r^3},$$

or as

$$\mathbf{A}_3(\mathbf{x}) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ c_2 & c_3 & 0 & 2 \\ 0 & \dfrac{r^3-1}{r^3} & -2 & 0 \end{bmatrix},$$

where

$$c_2 = \frac{r-1}{r^3}(x_1+2) + \frac{r^3-1}{r^3}, \qquad c_3 = \frac{r-1}{r^3}\left[x_2 + \frac{1}{x_2}(r+2)\right].$$

The MASRE algorithm has been initialized with a termination tolerance of 10^{-6}.

6.2.3 Results

The MASRE algorithm is not able to solve the HCP, and this failure is due to the inconsistency of the factorizations with final conditions: factorizations $\mathbf{A}_1(\mathbf{x})$ and $\mathbf{A}_2(\mathbf{x})$ are not defined for $x_1 = 0$, while $\mathbf{A}_3(\mathbf{x})$ is not defined for $x_2 = 0$; actually, factorization $\mathbf{A}_1(\mathbf{x})$ has been successfully used in [13], with the same set of boundary conditions, so that this is a good proof of how the inconsistency with final conditions is not a necessary condition, but still a critical point. By modifying the final boundary conditions to

$$\left.\begin{aligned} x_1 &= 10^{-7}, \\ x_2 &= 10^{-7}, \\ x_3 &= 10^{-7}, \\ x_4 &= 10^{-7}, \end{aligned}\right\}_{t=1} \tag{6.37}$$

the MASRE algorithm solves the HCP within five iterations, that are presented in Table 6.3. In this case, all of the three factorizations are both always controllable and consistent with the boundary conditions, and the optimal one turns out to be $\mathbf{A}_1(\mathbf{x})$. This is already a good proof that, when the consistency with boundary conditions is respected, the optimization process selects the best factorization, since it has been shown in [13] that the ASRE solution associated to the factorization $(\mathbf{A}_1(\mathbf{x}), \mathbf{B})$ yields the same results that are obtained in [28]. Going further, the stand-alone solutions obtained with the factorizations $\mathbf{A}_2(\mathbf{x})$ and $\mathbf{A}_3(\mathbf{x})$ show that actually the best solution in terms of cost function is the one related to $\mathbf{A}_2(\mathbf{x})$, but the difference with the solution obtained with the MASRE algorithm is minimal and also the controllability figures are very similar. The representations of

Table 6.3: Iterations of the MASRE algorithm for the rendez-vous HCP, computational time 26.2 s

	Error	σ_{min}	J	Matrix
1	4.790×10^{-1}	4.140×10^{-2}	9.582×10^{-1}	$\mathbf{A}_3$
2	7.179×10^{-3}	3.721×10^{-2}	9.586×10^{-1}	$\mathbf{A}_1$
3	7.029×10^{-5}	3.723×10^{-2}	9.587×10^{-1}	$\mathbf{A}_1$
4	6.290×10^{-6}	3.723×10^{-2}	9.587×10^{-1}	$\mathbf{A}_1$
5	5.408×10^{-7}	3.723×10^{-2}	9.587×10^{-1}	$\mathbf{A}_1$

Table 6.4: Iterations of the MASRE algorithm for the rendez-vous SCP, computational time 9.8 s

	Error	σ_{min}	J	Matrix
1	3.420×10^{-1}	4.140×10^{-2}	5.599×10^{-1}	$\mathbf{A}_3$
2	7.106×10^{-3}	3.923×10^{-2}	5.662×10^{-1}	$\mathbf{A}_1$
3	9.771×10^{-5}	3.925×10^{-2}	5.661×10^{-1}	$\mathbf{A}_1$
4	3.433×10^{-6}	3.925×10^{-2}	5.661×10^{-1}	$\mathbf{A}_1$
5	5.844×10^{-8}	3.925×10^{-2}	5.661×10^{-1}	$\mathbf{A}_1$

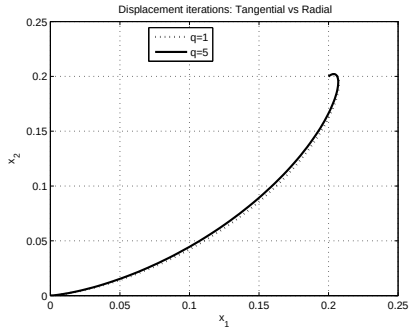
the state-space solutions and of the control are shown in Figure 6.3, while the optimal cost function after the refinement is

$$J_{opt}^{HCP} = 9.594 \times 10^{-1},$$

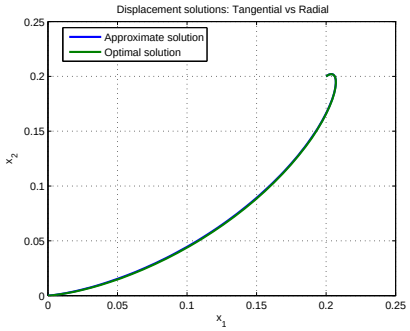
making clear that not always the optimal refinement leads to a decrease of the cost function. This particular example highlights the criticality of using factorizations that are not consistent with final conditions, even though in some cases it is still possible to use those factorization to solve the optimal control problem.

Regarding the SCP, which is more representative of a station-keeping procedure rather than a rendez-vous, the approximate solution is again obtained within five iterations, that are shown in Table 6.4, but this time the refinement is very marked, with a relative variation of the cost function of more than 6%. This strong modification is also noticeable in Figures 6.4d, 6.4f and 6.4b. On the other hand, the relaxation associated with a SCP is apparent in comparing its optimal cost function with that of the HCP

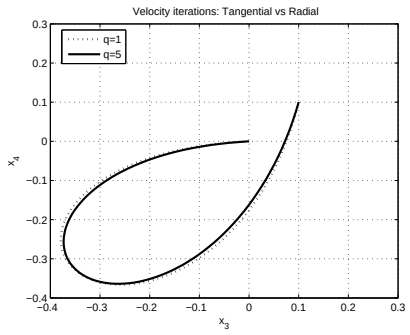
$$J_{opt}^{SCP} = 6.017 \times 10^{-1},$$



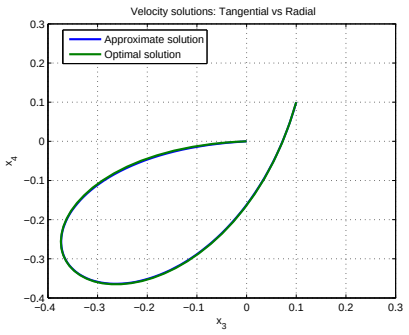
(a) MASRE phase-space iterations



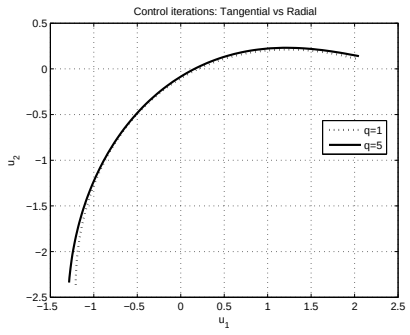
(b) Phase-space solutions



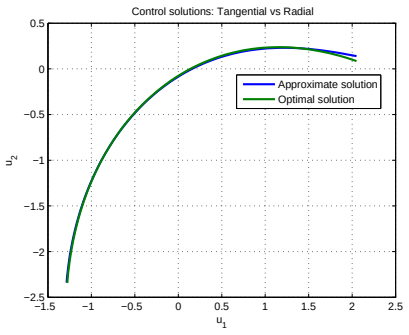
(c) MASRE phase-space iterations



(d) Phase-space solutions

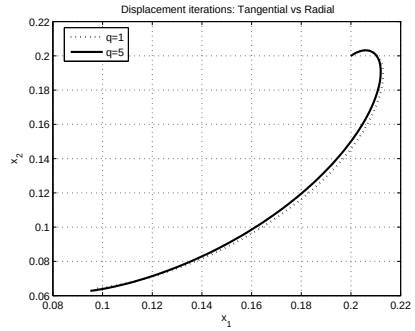


(e) MASRE control iterations - u_1 vs u_2

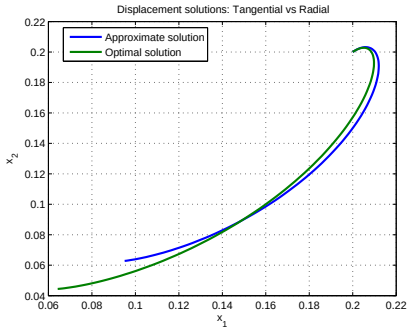


(f) Solutions for the control - u_1 vs u_2

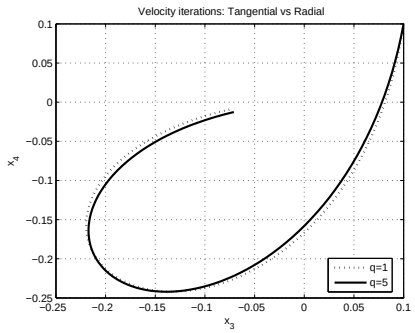
Figure 6.3: Solution of the rendez-vous HCP



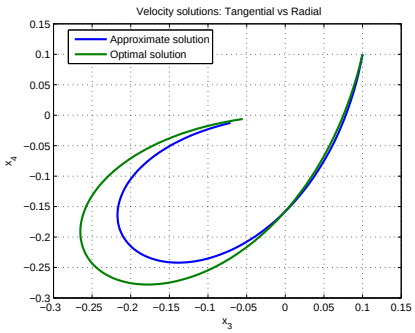
(a) MASRE phase-space iterations



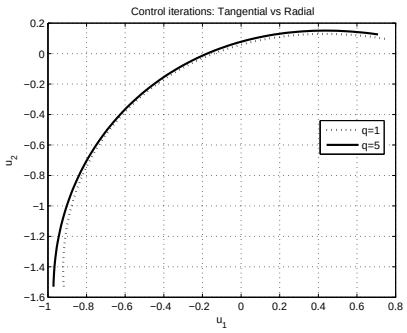
(b) Phase-space solutions



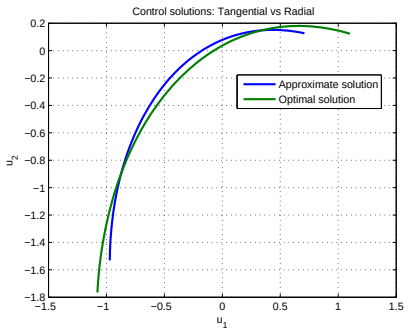
(c) MASRE phase-space iterations



(d) Phase-space solutions



(e) MASRE control iterations - u_1 vs u_2



(f) Solutions for the control - u_1 vs u_2

Figure 6.4: Solution of the rendez-vous SCP

6.3 Orbital transfer

One of the classical problems of astrodynamics is that of determining how to transfer a spacecraft from a given orbit to another. A simple way to deal with this kind of problem makes use of *impulsive manoeuvres*, i.e., the problem is solved by resorting to a number of discrete velocity impulses $\Delta\mathbf{V}$ that change the transfer trajectory. For example, considering a simple two-body problem, it is possible to bring a spacecraft from a circular orbit to a larger circular orbit that shares its centre with a couple of tangential impulses, one applied during the coast on the initial orbit and one applied when the spacecraft reaches the final orbit: this is known as the energy-efficient *Hohmann transfer*, depicted in Figure 6.5. When using this kind of manoeuvres, it is actually made the assumption that the velocity impulse is instantaneous, or at least that the distance covered by the spacecraft during the firing is very small compared to the complete transfer. Since the variation of linear momentum $m\Delta\mathbf{V}$ is the product of a thrust $\mathbf{T}$ and the duration time of the firing

$$m\Delta\mathbf{V} = \mathbf{T}\Delta t, \tag{6.38}$$

it is necessary for the thrust to be very intense. If not, the efficiency would decrease because part of the impulse $\Delta\mathbf{V}$ would be radial rather than tangential. This is why the typical apogee thrusters are solid-fuel motors, that possess the proper characteristics (see [7]). One of the problems of impulsive manoeuvres is that they require a great amount of fuel, i.e., the thrust per unit mass of expellant is quite low. To overcame this limitation, that strongly bounds the possible applications of impulsive manoeuvres, it has been resorted to *low-thrust propulsion*; since the total $\Delta\mathbf{V}$ depends mostly on the initial and final orbit, instead of employing intense thrusting for a limited time, it is possible to apply a small thrust, as small as of few mN, for a prolonged time, ideally even for the whole duration of the transfer. In this case, the thrusters are typically based on electric and magnetic propulsion, that ensure a good specific thrust, and so allow for long firings without the need of a critical amount of expellant. This kind of propulsion has very different technological bases from those of the well developed thermochemical propulsion, and it is also very interesting from the control point of view: in general, it is required a continuous regulation of the magnitude and direction of the thrust, and the dynamics involved are typically nonlinear. In the following sections it will be used the MASRE algorithm to solve the nonlinear optimal control problem that stems from low-thrust transfers.

6.3.1 Two-body problem

The first kind of orbital transfer that will be analysed is the one that takes into account only two bodies in relative motion under the action of their mutual gravitational attraction and of the control action. First, starting from

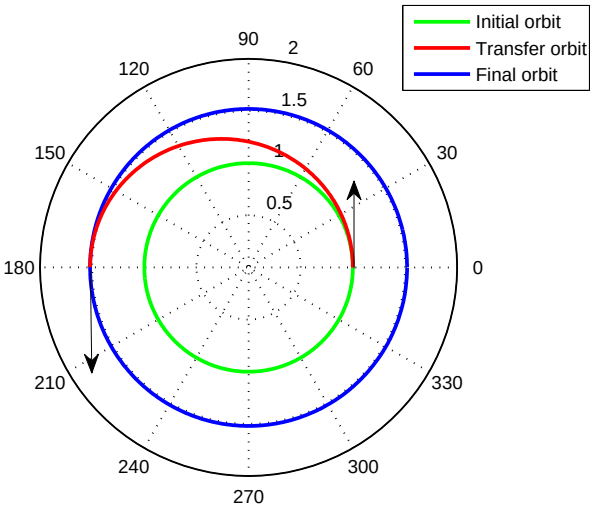


Figure 6.5: Hohmann transfer

equation (6.17), it will be written the system of equations representing the controlled dynamics, then the resulting nonlinear problem will be factorized, solved and lastly will follow a presentation of the results.

Derivation of the dynamics

The equation representing the controlled motion of a spacecraft orbiting around a certain celestial body is

$$\ddot{\mathbf{r}} = -\frac{\mu}{r^3}\mathbf{r} + \mathbf{u}. \tag{6.39}$$

The dynamics of this system are vectorial: a straightforward way to deal with this problem can rely on simply decomposing the system in its cartesian components

$$\mathbf{r} = \begin{Bmatrix} r\cos\theta \\ r\sin\theta \end{Bmatrix}, \tag{6.40}$$

but actually it is more sensible to stick to the natural polar representation of the problem, using as states not the projections of the position vector and their derivatives, but rather the magnitude of the relative distance r and the angle θ, formed with an inertial reference. According to this representation, the position vector is expressed as

$$\mathbf{r} = r\hat{\mathbf{i}}_r, \tag{6.41}$$

where $\hat{\mathbf{i}}_r$ is the unit vector associated with $\mathbf{r}$; as a consequence, the derivatives of the position vector are

$$\dot{\mathbf{r}} = \dot{r}\hat{\mathbf{i}}_r + r\dot{\theta}\hat{\mathbf{i}}_\theta, \tag{6.42}$$

$$\begin{aligned}\ddot{\mathbf{r}} &= \ddot{r}\hat{\mathbf{i}}_r + \dot{r}\dot{\theta}\hat{\mathbf{i}}_\theta + \dot{r}\dot{\theta}\hat{\mathbf{i}}_\theta + r\ddot{\theta}\hat{\mathbf{i}}_\theta - r\dot{\theta}^2\hat{\mathbf{i}}_r \\ &= \ddot{r}\hat{\mathbf{i}}_r + 2\dot{r}\dot{\theta}\hat{\mathbf{i}}_\theta + r\ddot{\theta}\hat{\mathbf{i}}_\theta - r\dot{\theta}^2\hat{\mathbf{i}}_r \\ &= \left(\ddot{r} - r\dot{\theta}^2\right)\hat{\mathbf{i}}_r + \left(2\dot{r}\dot{\theta} + r\ddot{\theta}\right)\hat{\mathbf{i}}_\theta. \end{aligned} \tag{6.43}$$

Substituting equation (6.41) and equation (6.43) in equation (6.39) and decomposing the control acceleration into a radial acceleration $u_r\hat{\mathbf{i}}_r$ and a transverse acceleration $u_\theta\hat{\mathbf{i}}_\theta$ yields

$$\left(\ddot{r} - r\dot{\theta}^2\right)\hat{\mathbf{i}}_r + \left(2\dot{r}\dot{\theta} + r\ddot{\theta}\right)\hat{\mathbf{i}}_\theta = -\frac{\mu}{r^2}\hat{\mathbf{i}}_r + u_r\hat{\mathbf{i}}_r + u_\theta\hat{\mathbf{i}}_\theta, \tag{6.44}$$

that can be easily split into radial and transverse dynamics

$$\ddot{r} - r\dot{\theta}^2 = -\frac{\mu}{r^2} + u_r, \tag{6.45}$$

$$2\dot{r}\dot{\theta} + r\ddot{\theta} = u_\theta. \tag{6.46}$$

By defining $x_1 = r$, $x_2 = \theta$, $x_3 = \dot{r}$ and $x_4 = \dot{\theta}$, it is possible to cast the problem into the classical state-space form

$$\begin{cases} \dot{x}_1 = x_3 \\ \dot{x}_2 = x_4 \\ \dot{x}_3 = x_1 x_4^2 - \dfrac{\mu}{x_1^2} + u_r \\ \dot{x}_4 = -2\dfrac{x_3 x_4}{x_1} + \dfrac{u_\theta}{x_1} \end{cases} \tag{6.47}$$

Problem set-up

System (6.47) is able to represent any two-body controlled problem. In our specific case, it will be studied a transfer between Earth orbit and Mars orbit in a Sun-centered polar reference; for the sake of simplicity, the problem will be written with scaled coordinates, where the unit length is one astronomical unit (1 ua), the time unit is such that the Earth period is 2π and the gravitational parameter of the Sun μ_{Sun} is equal to 1. The orbits of the planets are assumed to be circular. Initially, the spacecraft is coasting on the Earth orbit, so that $r(t_i) = 1$ and $\dot{r}(t_i) = 0$, while, recalling that the velocity of circular orbits is

$$v_\theta = r\dot{\theta} = \sqrt{\frac{\mu}{r}}, \tag{6.48}$$

the initial value of the angular velocity is $\dot{\theta}(t_i) = 1$; the initial angle is arbitrarily fixed to zero. Regarding the final conditions, the distance of Mars from the Sun is $r(t_f) = 1.52$, the radial component of velocity must be again null ($\dot{r}(t_f) = 0$) and the angular velocity, making use of equation (6.48), is $\dot{\theta}(t_f) = 1.52^{-\frac{3}{2}}$; the final angle can be chosen freely, or it can be even left as a free parameter, to be select by the optimal control. The cost function to be minimized is

$$J = \frac{1}{2}\int_{t_i}^{t_f} \mathbf{u}^T\mathbf{u}\mathrm{d}t, \tag{6.49}$$

where $\mathbf{u}^T = \begin{bmatrix} u_r & u_\theta \end{bmatrix}$.

The uncontrolled dynamics have been factorized in three different ways

$$\mathbf{A}_1 = \mathbf{A}_1(\mathbf{x}) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\dfrac{1}{x_1^3} & 0 & 0 & x_1x_4 \\ 0 & 0 & -2\dfrac{x_4}{x_1} & 0 \end{bmatrix},$$

$$\mathbf{A}_2 = \mathbf{A}_2(\mathbf{x}) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ x_4^2 - \dfrac{1}{x_1^3} & 0 & 0 & 0 \\ 0 & 0 & -2\dfrac{x_4}{x_1} & 0 \end{bmatrix},$$

$$\mathbf{A}_3 = \mathbf{A}_3(\mathbf{x}) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & x_4x_1 - \dfrac{1}{x_4x_1^2} \\ 0 & 0 & -2\dfrac{x_4}{x_1} & 0 \end{bmatrix},$$

while the rest of the matrices is

$$\mathbf{B} = \mathbf{B}(\mathbf{x}) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & \dfrac{1}{x_1} \end{bmatrix}, \quad \mathbf{Q} = \mathbf{0}_{4\times4}, \quad \mathbf{R} = \mathbf{I}_{2\times2}.$$

The problem has been solved for two different sets of boundary conditions: the first defines an *hard constrained* problem

$$\left.\begin{aligned} x_1 &= 1, \\ x_2 &= 0, \\ x_3 &= 0, \\ x_4 &= 1, \end{aligned}\right\}_{t=0} \qquad \left.\begin{aligned} x_1 &= 1.52, \\ x_2 &= \pi, \\ x_3 &= 0, \\ x_4 &= 1.52^{-\frac{3}{2}}, \end{aligned}\right\}_{t=\pi} \tag{6.50}$$

where all the states are specified both at initial and final time, while the second set is that of a *mixed constrained* problem, with the final angle left unspecified

$$\left.\begin{array}{l} x_1 = 1, \\ x_2 = 0, \\ x_3 = 0, \\ x_4 = 1, \end{array}\right\}_{t=0} \qquad \left.\begin{array}{r} x_1 = 1.52, \\ x_3 = 0, \\ x_4 = 1.52^{-\frac{3}{2}}. \end{array}\right\}_{t=\pi} \tag{6.51}$$

The mixed constrained problem requires a modification of the cost function in order to take into account a possible weighting of the final condition, introduced with a matrix $\mathbf{S}(\mathbf{x}_f)$

$$J = \frac{1}{2}\mathbf{x}_f^T\mathbf{S}(\mathbf{x}_f)\mathbf{x}_f + \frac{1}{2}\int_{t_i}^{t_f} \mathbf{u}^T\mathbf{u}\mathrm{d}t. \tag{6.52}$$

Since the only unspecified final state is x_2, this will be the only state to be weighted, accordingly, by a scalar $S(\mathbf{x}_f)$; actually, the final angle is not a term that directly influences the quality of the overall solution, so it will be excluded from the cost function by choosing

$$S = 0. \tag{6.53}$$

In both cases, the algorithm has been initialised by setting the termination tolerance to 10^{-6}.

Results

The iterations relative to the HCP are shown in Table 6.5, and the corresponding solution are shown in Figure 6.6a, along with the approximate control profiles in Figures 6.6c and 6.6e. The optimal cost function is

$$J_{opt} = 1.830 \times 10^{-1},$$

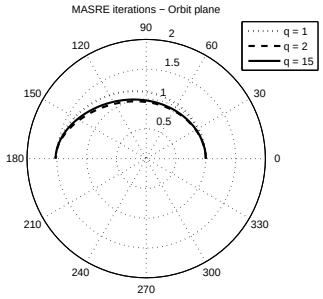
showing a consistent decrement if compared to the approximate solution; this is confirmed by Figures 6.6d and 6.6f, where the optimal controls are quite different from the approximate ones. It is interesting to notice that the factorization associated to matrix $\mathbf{A}_1(\mathbf{x})$ it is discarded because of its non-controllability, but it has been verified that if matrix $\mathbf{A}_1(\mathbf{x})$ is fed to the original ASRE, the algorithm is able to converge to an approximate solution whose cost function is

$$J_{approx} = 2.602 \times 10^{-1}.$$

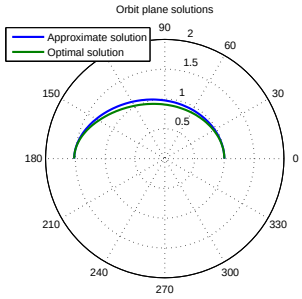
This implies that a factorization is not required to be always controllable in order to produce a converging sequence of approximation, i.e., controllability is not a necessary condition for the ASRE algorithm, so that the exclusion of those factorizations that are not always controllable is a very conservative

Table 6.5: Iterations of the MASRE algorithm for the two-body hard constrained optimal control problem, CPU time 60.1 s. The first iteration shows no convergence to any of the two matrices $\mathbf{A}_1$ or $\mathbf{A}_2$. Notice that the cost function's value of the last iteration is lower than the one relative to the non-controllable matrix

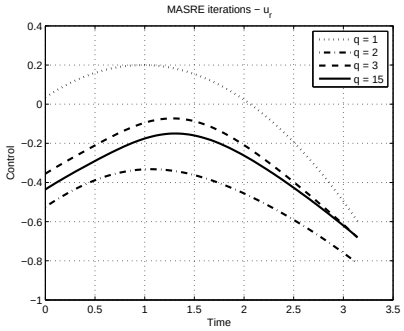
	Error	σ_{min}	J	Matrix
1	3.142	4.058×10^{-1}	1.674×10^{-1}	NO
2	1.760×10^{-1}	3.519×10^{-1}	3.956×10^{-1}	$\mathbf{A}_2$
3	5.618×10^{-2}	3.095×10^{-1}	1.940×10^{-1}	$\mathbf{A}_2$
4	3.565×10^{-2}	3.082×10^{-1}	2.423×10^{-1}	$\mathbf{A}_2$
5	1.179×10^{-2}	3.040×10^{-1}	2.184×10^{-1}	$\mathbf{A}_2$
6	5.335×10^{-3}	3.049×10^{-1}	2.279×10^{-1}	$\mathbf{A}_2$
7	1.939×10^{-3}	3.046×10^{-1}	2.244×10^{-1}	$\mathbf{A}_2$
8	6.792×10^{-4}	3.048×10^{-1}	2.258×10^{-1}	$\mathbf{A}_2$
9	2.533×10^{-4}	3.047×10^{-1}	2.253×10^{-1}	$\mathbf{A}_2$
10	8.548×10^{-5}	3.047×10^{-1}	2.255×10^{-1}	$\mathbf{A}_2$
11	3.057×10^{-5}	3.047×10^{-1}	2.254×10^{-1}	$\mathbf{A}_2$
12	1.074×10^{-5}	3.047×10^{-1}	2.254×10^{-1}	$\mathbf{A}_2$
13	3.699×10^{-6}	3.047×10^{-1}	2.254×10^{-1}	$\mathbf{A}_2$
14	1.320×10^{-6}	3.047×10^{-1}	2.254×10^{-1}	$\mathbf{A}_2$
15	4.567×10^{-7}	3.047×10^{-1}	2.254×10^{-1}	$\mathbf{A}_2$



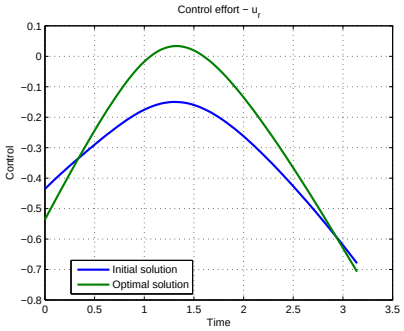
(a) MASRE orbit plane iterations



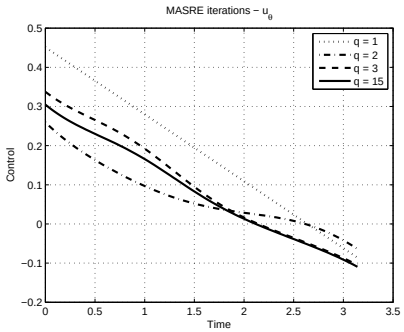
(b) Orbit plane solutions



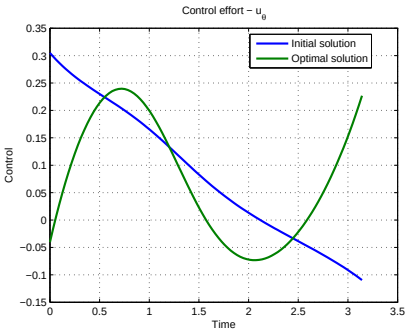
(c) MASRE control iterations - u_r



(d) Solutions for the control - u_r



(e) MASRE control iterations - u_θ



(f) Solutions for the control - u_θ

Figure 6.6: Solution of the two-body HCP

Table 6.6: Iterations of the MASRE algorithm for the two-body mixed constrained optimal control problem, CPU time 19.9 s. The first iteration shows no convergence to any of the two matrices $\mathbf{A}_1$ or $\mathbf{A}_2$

	Error	σ_{min}	J	Matrix
1	2.409	4.058×10^{-1}	1.047×10^{-1}	NO
2	5.237×10^{-2}	3.434×10^{-1}	3.297×10^{-2}	$\mathbf{A}_2$
3	4.522×10^{-2}	3.146×10^{-1}	4.924×10^{-2}	$\mathbf{A}_2$
4	8.627×10^{-3}	3.126×10^{-1}	4.652×10^{-2}	$\mathbf{A}_2$
5	2.728×10^{-3}	3.127×10^{-1}	4.801×10^{-2}	$\mathbf{A}_2$
6	5.133×10^{-4}	3.129×10^{-1}	4.839×10^{-2}	$\mathbf{A}_2$
7	3.166×10^{-4}	3.131×10^{-1}	4.845×10^{-2}	$\mathbf{A}_2$
8	1.568×10^{-4}	3.132×10^{-1}	4.846×10^{-2}	$\mathbf{A}_2$
9	5.434×10^{-5}	3.132×10^{-1}	4.845×10^{-2}	$\mathbf{A}_2$
10	1.475×10^{-5}	3.132×10^{-1}	4.845×10^{-2}	$\mathbf{A}_2$
11	3.045×10^{-6}	3.132×10^{-1}	4.845×10^{-2}	$\mathbf{A}_2$
12	1.325×10^{-6}	3.132×10^{-1}	4.845×10^{-2}	$\mathbf{A}_2$
13	7.700×10^{-7}	3.132×10^{-1}	4.845×10^{-2}	$\mathbf{A}_2$

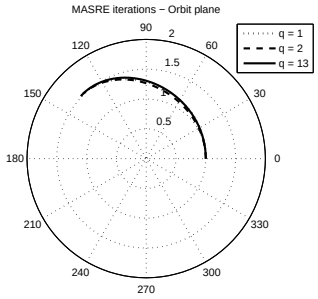
choice, but, comparing the cost function's values of the two approximate solutions, it is apparent that the factorization that is always controllable yields better results.

The problem has also been solved enforcing the set of boundary conditions (6.51), that defines a MCP. The sequence of iterations of the MASRE algorithm is shown in Table 6.6, and Figure 6.7 shows the various solutions in the orbit plane and the control profiles. The optimal cost function found is

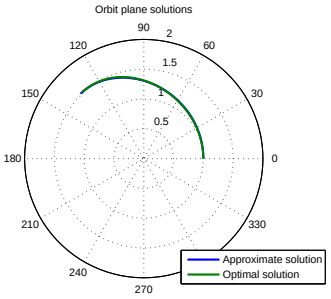
$$J_{opt} = 3.695 \times 10^{-2}.$$

As it would have been expected, by letting the algorithm determine by itself the final angle, the MCP's optimal cost function is lower than the optimal cost function of the HCP's solution, but also the approximate cost function is decreased, and even the convergence is faster, since the algorithm can exploit the additional degree of freedom provided by the unspecified value of the final angular position. The solution can be confronted with the one found in [1], section 2.5, where it is solved a similar problem, i.e., to determine the thrust-direction history for a minimum-time orbital transfer between Earth's and Mars' orbits. Albeit with proper differences, the thrust profile is similar, with the radial component monotonically decreasing from a positive to a negative value, and the transversal component switching from positive to negative and then again positive.

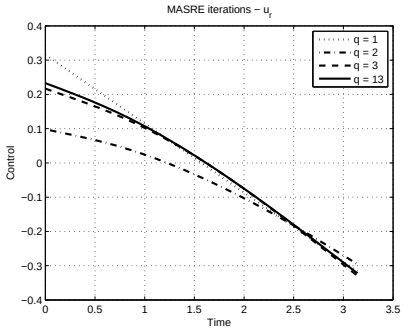
In conclusion, this example shows how the automatic selection of the proper factorization leads to better factorization; otherwise, it would be



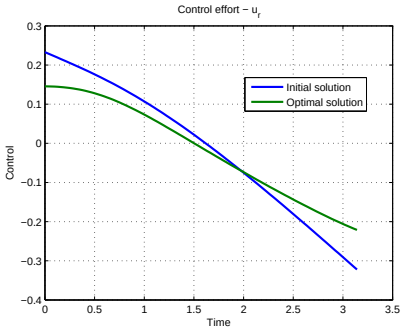
(a) MASRE orbit plane iterations



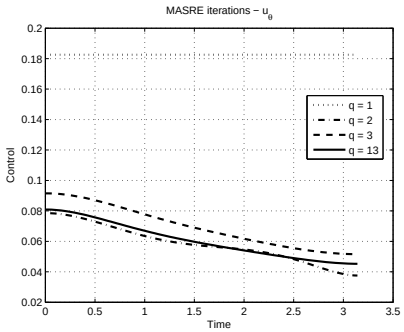
(b) Orbit plane solutions



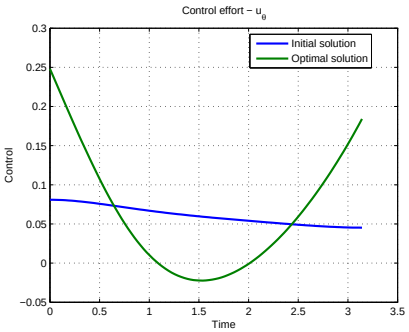
(c) MASRE control iterations - u_r



(d) Solutions for the control - u_r



(e) MASRE control iterations - u_θ



(f) Solutions for the control - u_θ

Figure 6.7: Solution of the two-body MCP

necessary to verify one by one which factorization yields the best results in terms of cost function.

6.3.2 Restricted three-body problem

The two-body problem is a simplified model that can be satisfactory when dealing with orbits that are very close to a single attractor, or when the solution that is looked for does not need to be refined, as it happens during the preliminary phases of a project; on the other hand, if the problem considered is actually referring to a more complex situation that needs to be analysed in detail, it is sensible to resort to a more realistic model, that takes into account the effects of more than one attractor. In the following, it will be presented the so called Circular Restricted Three-Body Problem (CRTBP), that deals with a system of three bodies, two of which are massive attractors (*primaries*) moving under the mutual force of gravity on circular orbits, while the third has negligible mass, not influencing the motion of the attractors; the CRTBP consists in describing the motion of this third body.

The solution of this problem leads to the identification of five *equilibrium points* or *lagrangian points*; three of them, called *eulerian* or *collinear*, are aligned with the primaries, while the other two, called *triangular* or *lagrangian* in a strict sense, are located on the vertices of the equilateral triangles formed by the primaries. Those equilibrium points can be exploited to linearise the dynamical system, thus obtaining informations about the solutions, i.e., orbits, around those equilibrium points: the collinear points are *unstable*, while the triangular ones are *stable*. Actually, the orbits around collinear points can be made periodic by properly selecting the initial conditions, and this, taking into account the fact that the stable equilibrium points require an higher level of energy to be reached, makes the orbits around collinear points very interesting. In order to reach those periodic orbits, it is possible to exploit the *stable manifolds* (see [14]) associated, i.e., trajectories that asymptotically reach a periodic orbit around a collinear point. The next section will focus on selecting the optimal control that allows to reach a stable manifold associated to a periodic orbit around the L1 point of the Earth-Moon system.

Derivation of the dynamics

The primaries and the spacecraft are modelled as point masses, respectively $m_1 > m_2 \gg m_3$, while the distances of the primaries from the centre of mass are respectively a and b, so that the distance between the primaries is $l = a + b$; the equality of centrifugal and gravity forces acting on the

primaries yields

$$G\frac{m_1 m_2}{l^2} = m_1 a n^2, \tag{6.54}$$

$$G\frac{m_1 m_2}{l^2} = m_2 b n^2, \tag{6.55}$$

where G is the gravitational constant and n is the angular velocity around the center of mass; by multiplying both sides of equation (6.54) by m_2 and both sides of equation (6.55) by m_1 and adding the resulting equalities, it is possible to write

$$G\frac{m_1 m_2^2}{l^2} + G\frac{m_1^2 m_2}{l^2} = m_1 m_2 a n^2 + m_1 m_2 b n^2,$$

$$\Downarrow$$

$$Gm_1 m_2 \frac{m_1 + m_2}{l^2} = m_1 m_2 \left(a + b\right) n^2 = m_1 m_2 l n^2,$$

$$\Downarrow$$

$$n^2 = G\frac{m_1 + m_2}{l^3}, \tag{6.56}$$

so that, substituting expression (6.56) inside equations (6.54) and (6.55), the distances of the primaries from the centre of mass can be expressed as

$$a = \frac{m_2 l}{m_1 + m_2}, \tag{6.57}$$

$$b = \frac{m_1 l}{m_1 + m_2}. \tag{6.58}$$

In order to avoid the explicit time dependence due to the motion of the primaries, it is possible to write the dynamics in a *synodic* reference frame, i.e., rotating with the primaries, whose origin lies in the centre of mass of the system, and whose abscissa is aligned with the primaries. In this reference frame, the coordinates of the spacecraft are $\overline{x}$, $\overline{y}$ and $\overline{z}$, so that its position is

$$\mathbf{r} = \overline{x}\hat{\mathbf{i}} + \overline{y}\hat{\mathbf{j}} + \overline{z}\hat{\mathbf{k}}, \tag{6.59}$$

and the corresponding derivatives are

$$\dot{\mathbf{r}} = \dot{\overline{x}}\hat{\mathbf{i}} + \overline{x}n\hat{\mathbf{j}} + \dot{\overline{y}}\hat{\mathbf{j}} - \overline{y}n\hat{\mathbf{j}} + \dot{\overline{z}}\hat{\mathbf{k}}, \tag{6.60}$$

$$\begin{aligned}\ddot{\mathbf{r}} &= \ddot{\overline{x}}\hat{\mathbf{i}} + \dot{\overline{x}}n\hat{\mathbf{j}} + \dot{\overline{x}}n\hat{\mathbf{j}} - \overline{x}n^2\hat{\mathbf{i}} + \ddot{\overline{y}}\hat{\mathbf{j}} - \dot{\overline{y}}n\hat{\mathbf{i}} - \dot{\overline{y}}n\hat{\mathbf{i}} - \overline{y}n^2\hat{\mathbf{j}} + \ddot{\overline{z}}\hat{\mathbf{k}} \\ &= \left(\ddot{\overline{x}} - 2n\dot{\overline{y}} - n^2\overline{x}\right)\hat{\mathbf{i}} + \left(\ddot{\overline{y}} + 2n\dot{\overline{x}} - n^2\overline{y}\right)\hat{\mathbf{j}} + \ddot{\overline{z}}\hat{\mathbf{k}}.\end{aligned} \tag{6.61}$$

The distances of the spacecraft from the primaries are, respectively,

$$\overline{r}_1 = \sqrt{\left(\overline{x} + a\right)^2 + \overline{y}^2 + \overline{z}^2}, \tag{6.62}$$

$$\overline{r}_2 = \sqrt{\left(\overline{x} - b\right)^2 + \overline{y}^2 + \overline{z}^2}, \tag{6.63}$$

Decomposing the dynamics on the synodic reference frame yields

$$\ddot{\overline{x}} - 2n\dot{\overline{y}} - n^2\overline{x} = -G\left[\frac{m_1\left(\overline{x}+a\right)}{\overline{r}_1^3} + \frac{m_2\left(\overline{x}-b\right)}{\overline{r}_2^3}\right], \tag{6.64}$$

$$\ddot{\overline{y}} + 2n\dot{\overline{x}} - n^2\overline{y} = -G\left[\frac{m_1\overline{y}}{\overline{r}_1^3} + \frac{m_2\overline{y}}{\overline{r}_2^3}\right], \tag{6.65}$$

$$\ddot{\overline{z}} = -G\left[\frac{m_1\overline{z}}{\overline{r}_1^3} + \frac{m_2\overline{z}}{\overline{r}_2^3}\right]. \tag{6.66}$$

Switching to a set of non-dimensional variables

$$\begin{aligned} x &= \frac{\overline{x}}{l}, \qquad y = \frac{\overline{y}}{l}, \qquad z = \frac{\overline{z}}{l}, \\ t &= nt^\star, \quad r_1 = \frac{\overline{r}_1}{l}, \quad r_2 = \frac{\overline{r}_2}{l}, \end{aligned} \tag{6.67}$$

and defining a *mass parameter*

$$\mu = \frac{m_2}{m_1+m_2}, \tag{6.68}$$

so that

$$\begin{aligned} a &= \frac{m_2 l}{m_1+m_2} = \mu l, \\ b &= \frac{m_1 l}{m_1+m_2} = (1-\mu)\, l, \end{aligned}$$

allows a simplification of the dynamics

$$\begin{cases} n^2 l\ddot{x} - 2n^2 l\dot{y} - n^2 lx = -G\left[\dfrac{m_1\left(xl+\mu l\right)}{r_1^3 l^3} + \dfrac{m_2\left(xl-(1-\mu)\, l\right)}{r_2^3 l^3}\right], \\ n^2 l\ddot{y} + 2n^2 l\dot{x} - n^2 ly = -G\left[\dfrac{m_1 yl}{r_1^3 l^3} + \dfrac{m_2 yl}{r_2^3 l^3}\right], \\ n^2 l\ddot{z} = -G\left[\dfrac{m_1 zl}{r_1^3 l^3} + \dfrac{m_2 zl}{r_2^3 l^3}\right], \end{cases}$$

$$\Downarrow$$

$$\begin{cases} \ddot{x} - 2\dot{y} - x = -\dfrac{G}{n^2 l^3}\left[\dfrac{m_1\left(x+\mu\right)}{r_1^3} + \dfrac{m_2\left(x-1+\mu\right)}{r_2^3}\right], \\ \ddot{y} + 2\dot{x} - y = -\dfrac{G}{n^2 l^3}\left[\dfrac{m_1 y}{r_1^3} + \dfrac{m_2 y}{r_2^3}\right], \\ \ddot{z} = -\dfrac{G}{n^2 l^3}\left[\dfrac{m_1 z}{r_1^3} + \dfrac{m_2 z}{r_2^3}\right]. \end{cases}$$

Recalling equation (6.56)

$$\begin{cases} \ddot{x} - 2\dot{y} - x = -\dfrac{1}{m_1 + m_2}\left[\dfrac{m_1\,(x+\mu)}{r_1^3} + \dfrac{m_2\,(x-1+\mu)}{r_2^3}\right], \\ \ddot{y} + 2\dot{x} - y = -\dfrac{1}{m_1 + m_2}\left[\dfrac{m_1 y}{r_1^3} + \dfrac{m_2 y}{r_2^3}\right], \\ \ddot{z} = -\dfrac{1}{m_1 + m_2}\left[\dfrac{m_1 z}{r_1^3} + \dfrac{m_2 z}{r_2^3}\right], \end{cases}$$

$$\Downarrow$$

$$\begin{cases} \ddot{x} - 2\dot{y} - x = -\dfrac{m_1}{m_1 + m_2}\dfrac{x+\mu}{r_1^3} - \dfrac{m_2}{m_1 + m_2}\dfrac{x-1+\mu}{r_2^3}, \\ \ddot{y} + 2\dot{x} - y = -\dfrac{m_1}{m_1 + m_2}\dfrac{y}{r_1^3} - \dfrac{m_2}{m_1 + m_2}\dfrac{y}{r_2^3}, \\ \ddot{z} = -\dfrac{m_1}{m_1 + m_2}\dfrac{z}{r_1^3} - \dfrac{m_2}{m_1 + m_2}\dfrac{z}{r_2^3}, \end{cases}$$

$$\Downarrow$$

$$\begin{cases} \ddot{x} - 2\dot{y} - x = -\dfrac{(1-\mu)\,(x+\mu)}{r_1^3} - \dfrac{\mu\,(x-1+\mu)}{r_2^3}, \\ \ddot{y} + 2\dot{x} - y = -\dfrac{(1-\mu)\,y}{r_1^3} - \dfrac{\mu y}{r_2^3}, \\ \ddot{z} = -\dfrac{(1-\mu)\,z}{r_1^3} - \dfrac{\mu z}{r_2^3}. \end{cases} \tag{6.69}$$

The introduction of a potential

$$\Omega = \frac{1}{2}\left(x^2 + y^2\right) + \frac{1-\mu}{r_1} + \frac{\mu}{r_2} + \frac{1}{2}\mu\,(1-\mu)\,, \tag{6.70}$$

where the last term is a constant, further simplifies the dynamics; the partial derivatives of Ω with respect to x, y and z are

$$\Omega_x = \frac{\partial\Omega}{\partial x} = x - \frac{(1-\mu)\,(x+\mu)}{r_1^3} - \frac{\mu\,(x-1+\mu)}{r_2^3},$$

$$\Omega_y = \frac{\partial\Omega}{\partial x} = y - \frac{(1-\mu)\,y}{r_1^3} - \frac{\mu y}{r_2^3},$$

$$\Omega_z = \frac{\partial\Omega}{\partial z} = -\frac{(1-\mu)\,z}{r_1^3} - \frac{\mu z}{r_2^3},$$

so that the uncontrolled dynamics can be expressed as

$$\begin{cases} \ddot{x} - 2\dot{y} = \Omega_x, \\ \ddot{y} + 2\dot{x} = \Omega_y, \\ \ddot{z} = \Omega_z. \end{cases} \tag{6.71}$$

The synodic reference frame allows a formulation that can be used to describe any CRTBP by simply selecting the proper mass parameter and boundary conditions.

Problem set-up

In this example, that follows from [9], it is considered the three-body system Earth-Moon-Spacecraft, and the analysis will be restricted by assuming that the spacecraft is bounded to move in the plane of the primaries; the mass parameter μ is

$$\mu = \frac{m_{Moon}}{m_{Earth} + m_{Moon}} = 0.012,$$

while, regarding the boundary conditions, it is necessary to define what is the orbital transfer: the transfer to be designed starts from a circular orbit located at an altitude $h = 75\,000$ km, in particular from the point lying between the primaries, and its target is to reach a stable manifold of a periodic orbit around L1 that has semi-amplitudes of 25 400 km and 60 000 km along $\overline{x}$ and $\overline{y}$, respectively. It can be shown that a generic point of the stable manifold $\mathbf{x}_s = (x_s, y_s, \dot{x}_s, \dot{y}_s)$ is uniquely identified by using two scalars, i.e., $\mathbf{x}_s(\tau_1, \tau_2)$, where τ_1 and τ_2 are two time variables. For this reason, in order to belong to the stable manifold, the final transfer state $\mathbf{x}_f = (x_f, y_f, \dot{x}_f, \dot{y}_f)$ must satisfy the final boundary conditions

$$\left.\begin{aligned} x_f &= x_s, \\ y_f &= y_s, \\ \dot{x}_f &= \dot{x}_s, \\ \dot{y}_f &= \dot{y}_s. \end{aligned}\right\} \tag{6.72}$$

In Figure 6.8 it is depicted the stable manifold orbit around L1 in the synodic reference frame, along with the circular starting orbit and the extremes of the transfer orbit; the planar dynamics, with the addition of the acceleration controls u_1 and u_2,

$$\begin{cases} \ddot{x} - 2\dot{y} - x = -\dfrac{(1-\mu)(x+\mu)}{r_1^3} - \dfrac{\mu(x-1+\mu)}{r_2^3} + u_1, \\ \ddot{y} + 2\dot{x} - y = -\dfrac{(1-\mu)y}{r_1^3} - \dfrac{\mu y}{r_2^3} + u_2, \end{cases} \tag{6.73}$$

have been first expressed in the canonical state-space form, with $x_1 = x$, $x_2 = y$, $x_3 = \dot{x}$ and $x_4 = \dot{y}$,

$$\begin{cases} \dot{x}_1 = x_3, \\ \dot{x}_2 = x_4, \\ \dot{x}_3 = x_1 + 2x_4 - \dfrac{(1-\mu)(x_1+\mu)}{r_1^3} - \dfrac{\mu(x_1-1+\mu)}{r_2^3} + u_1, \\ \dot{x}_4 = x_2 - 2x_3 - \dfrac{(1-\mu)x_2}{r_1^3} - \dfrac{\mu x_2}{r_2^3} + u_2, \end{cases} \tag{6.74}$$

and then the system has been factorized taking into account that the initial conditions prescribe a zero-valued x_2 and x_3, so that any factorization that

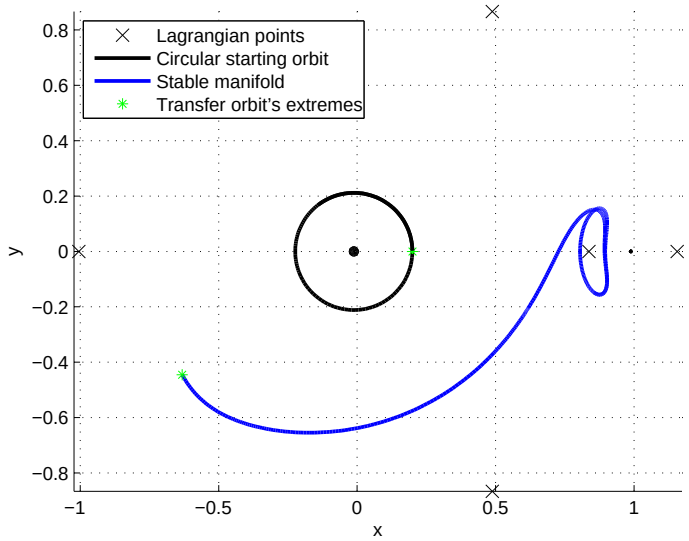


Figure 6.8: Synodic reference frame with Earth, Moon, lagrangian points, the circular starting orbit, the stable manifold and the extremes of the transfer orbit

would have been not consistent has been discarded; the matrices associated

to the uncontrolled dynamics are

$$\mathbf{A}_1(\mathbf{x}) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 - \frac{c_1}{x_1} - \frac{c_2}{x_1} & 0 & 0 & 2 \\ 0 & c & -2 & 0 \end{bmatrix}, \tag{6.75}$$

$$\mathbf{A}_2(\mathbf{x}) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 - \frac{c_1}{x_1} & 0 & 0 & 2 - \frac{c_2}{x_2} \\ 0 & c & -2 & 0 \end{bmatrix}, \tag{6.76}$$

$$\mathbf{A}_3(\mathbf{x}) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 - \frac{c_2}{x_1} & 0 & 0 & 2 - \frac{c_1}{x_2} \\ 0 & c & -2 & 0 \end{bmatrix}, \tag{6.77}$$

$$\mathbf{A}_4(\mathbf{x}) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 2 - \frac{c_1}{x_2} - \frac{c_2}{x_2} \\ 0 & c & -2 & 0 \end{bmatrix}, \tag{6.78}$$

$$\mathbf{A}_5(\mathbf{x}) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 - c_1\frac{x_1}{x_1^2+x_2^2} - c_2\frac{x_1}{x_1^2+x_2^2} & -c_1\frac{x_2}{x_1^2+x_2^2} - c_2\frac{x_2}{x_1^2+x_2^2} & 0 & 2 \\ 0 & c & -2 & 0 \end{bmatrix}, \tag{6.79}$$

where

$$c_1 = \frac{(1-\mu)(x_1+\mu)}{r_1^3}, \tag{6.80}$$

$$c_2 = \frac{\mu(x_1-1+\mu)}{r_2^3}, \tag{6.81}$$

$$c = 1 - \frac{(1-\mu)x_2}{r_1^3} - \frac{\mu x_2}{r_2^3}. \tag{6.82}$$

Regarding the control and the cost function, the associated matrices will be

$$\mathbf{B} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{Q} = \mathbf{0}_{4\times4}, \quad \mathbf{R} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{6.83}$$

Initial and final time are set to $t_i = 0$ and $t_f = 0.5$, the termination tolerance is 10^{-6}.

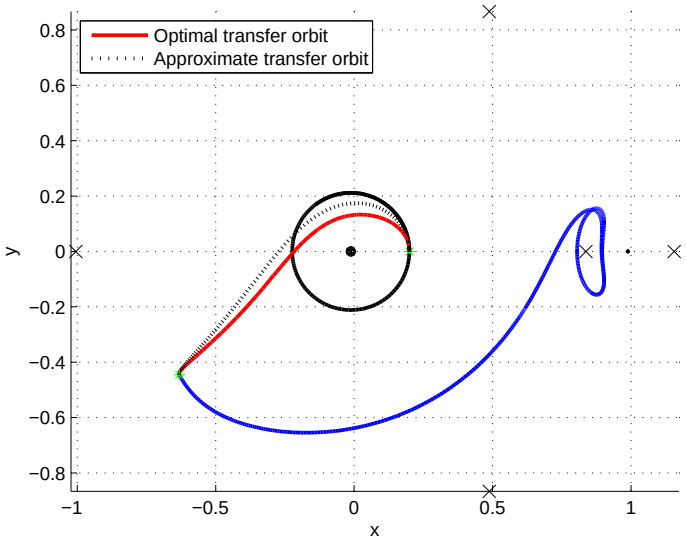


Figure 6.9: Approximate and refined solutions obtained with matrix $\mathbf{A}_5$, CPU time 63.5 s

Results

The algorithm is not able to converge to any solution; this is caused by the singularities that arise during the first iteration: by solving the problem with matrix $\mathbf{A}_5$, it is easy to notice that both components of the position and velocity vectors change sign during the transfer; this means that the matrices $\mathbf{A}_1$, $\mathbf{A}_2$, $\mathbf{A}_3$ and $\mathbf{A}_4$ become not-definite, preventing the algorithm from integrating the STM. In Figure 6.9 it is shown the refined solution obtained by solving the problem using only matrix $\mathbf{A}_5$; this example makes clear that singularities can arise without being able to detect them *a priori*, so that the factorizations must be computed bearing this in mind, e.g., matrix $\mathbf{A}_5$ becomes not defined only in the origin of the synodic reference frame, so that it is safe to assume that it will not be critical. Furthermore, it must be made an annotation about the quality of the solution found: the problem has been originally solved using direct transcription (see [9]), and for that case the solution was a spiral, associated to a cost function much lower than the one found with the MASRE algorithm; this is due to the intrinsic necessity of direct transcription of a starting guess (that can be exploited to select a desired solution), while the ASRE method has no flexibility from this side, providing a solution that is maybe the most "simple", but not the optimal one.

6.4 Station-keeping

Once a spacecraft, typically a satellite, has been successfully injected on its nominal orbit, it is still necessary to counteract those disturbances that would lead the spacecraft to drift away from the desired path. Among the possible disturbances acting on the keplerian motion, the main effects are due to the third body effects, the non-spherical shape of the main attractor, the solar pressure, the magnetic field of the main attractor, the atmospheric drag... The kinds of disturbance to be taken into account depend on the nominal orbit, i.e., some disturbances can be neglected when the associated effect is much smaller than the others, or in general when the effects are not perceptible. In the following example it will be assumed a geostationary orbit: these orbits are exploited because of the apparent null longitudinal motion and the consequent small need of tracking procedures, a feature that is particularly appreciated when dealing with telecommunications or meteorological observations. This has determined a rapid growth of the number of satellites on the geostationary orbit, and, in parallel, a increased precision over the reduced tolerance requested. The control manoeuvres can be actuated employing chemical thrusters pulsing cyclically, but, in order to increase the efficiency and thus the lifetime, it is possible to use low-thrust actuators continuously; this is when intervenes nonlinear control, allowing a fine regulation of the station-keeping procedure.

In the following it will be presented a derivation of the dynamics in a proper reference frame, along with the models of the different perturbations considered.

6.4.1 Derivation of the dynamics

First of all, the reference frame adopted is called *Earth-Centered Earth-Fixed* (ECEF), i.e., a cartesian frame rotating jointly with the planet, whose origin lies in the center of the Earth, whose x and y axes lie in the equatorial plane and whose z axis points toward the North pole. Shifting to a spherical coordinates system, the λ and φ coordinates represent longitude and latitude, respectively, while the r coordinate identifies the distance from the origin. The relation between the cartesian and the spherical system is

$$\begin{Bmatrix} x \\ y \\ z \end{Bmatrix} = \begin{Bmatrix} r\cos\varphi\cos\lambda \\ r\cos\varphi\sin\lambda \\ r\sin\varphi \end{Bmatrix}. \tag{6.84}$$

It is sensible to use the ECEF frame since it is easy to express the deviations of the spacecraft from the nominal position, but also because the perturbing accelerations can be computed recurring to potentials. The dynamics are obtained starting from the Lagrangian L of the system, defined as

$$L = L\left(\mathbf{q}, \dot{\mathbf{q}}\right) = T\left(\mathbf{q}, \dot{\mathbf{q}}\right) - V\left(\mathbf{q}, \dot{\mathbf{q}}\right), \tag{6.85}$$

where T and V are the kinetic and potential energy of the system, respectively, and with $\mathbf{q}$ have been expressed the generalized coordinates. The equation of motion

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = Q_i, \tag{6.86}$$

where the generalized vector $\mathbf{Q}$ has been added to take into account the non-conservative forces acting on the system, yields the dynamics; for the specific case being studied, the Lagrangian is

$$L = \frac{1}{2}\left[\dot{r}^2 + r^2\dot{\varphi}^2 + \left(r^2\cos^2\varphi\right)\left(\dot{\lambda}+\omega\right)^2\right] - W\left(r,\lambda,\varphi\right), \tag{6.87}$$

where ω is the angular velocity of the reference system and W is the potential associated to the conservative forces. Applying equation (6.86) to (6.87) yields

$$\begin{aligned} \ddot{r} - r\dot{\varphi}^2 - r\cos^2\varphi\left(\dot{\lambda}+\omega\right)^2 &= \frac{\partial W}{\partial r}, \\ \frac{\mathrm{d}}{\mathrm{d}t}\left[r^2\cos^2\varphi\left(\dot{\lambda}+\omega\right)\right] &= \frac{\partial W}{\partial \lambda}, \\ \frac{\mathrm{d}}{\mathrm{d}t}\left(r^2\dot{\varphi}\right) + r^2\sin\varphi\cos\varphi\left(\dot{\lambda}+\omega\right)^2 &= \frac{\partial W}{\partial \varphi}. \end{aligned} \tag{6.88}$$

Actually, the partial derivatives of the potential with respect to λ and φ are not accelerations, and looking at the gradient operator in spherical coordinates

$$\nabla\left(\cdot\right) = \frac{\partial\left(\cdot\right)}{\partial r}\hat{r} + \frac{1}{r\cos\varphi}\frac{\partial\left(\cdot\right)}{\partial\lambda}\hat{\lambda} + \frac{1}{r}\frac{\partial\left(\cdot\right)}{\partial\varphi}\hat{\varphi}, \tag{6.89}$$

makes clear that, in order to maintain dimensional consistency, it is sufficient to multiply by $r\cos\varphi$ the accelerations along λ and by r the accelerations along φ; taking into account the accelerations $\tilde{\mathbf{a}}$ that do not stem from a potential, the dynamics become

$$\begin{aligned} \ddot{r} - r\dot{\varphi}^2 - r\cos^2\varphi\left(\dot{\lambda}+\omega\right)^2 &= w_r + \tilde{a}_r, \\ \frac{\mathrm{d}}{\mathrm{d}t}\left[r^2\cos^2\varphi\left(\dot{\lambda}+\omega\right)\right] &= \left(w_\lambda + \tilde{a}_\lambda\right) r\cos\varphi, \\ \frac{\mathrm{d}}{\mathrm{d}t}\left(r^2\dot{\varphi}\right) + r^2\sin\varphi\cos\varphi\left(\dot{\lambda}+\omega\right)^2 &= \left(w_\varphi + \tilde{a}_\varphi\right) r. \end{aligned} \tag{6.90}$$

Up to now, the longitude has been expressed with λ, but it is sensible to define a nominal longitude λ_n that will be constant, since the reference frame is rotating jointly with the Earth, so that the actual longitude will be

$$\lambda = \lambda_n + \varepsilon, \tag{6.91}$$

where ε is the deviation from the nominal longitude; of course, since λ_n is a constant, the variation of the longitude will be $\dot{\lambda} = \dot{\varepsilon}$. Furthermore, it is possible to split the radial component of potential in two terms

$$w_r = -\frac{\mu}{r^2} + \tilde{w}_r, \tag{6.92}$$

separating the effect of the main attractor, i.e., the Earth, from those of the other conservative forces; taking into account the last two remarks, and computing the time derivatives, the dynamics results to be

$$\begin{aligned} \ddot{r} - r\dot{\varphi}^2 - r\cos^2\varphi\left(\dot{\varepsilon}+\omega\right)^2 &= -\mu/r^2 + a_r, \\ 2r\dot{r}\cos^2\varphi\left(\dot{\varepsilon}+\omega\right) - 2r^2\dot{\varphi}\sin\varphi\cos\varphi\left(\dot{\varepsilon}+\omega\right) + r^2\cos^2\varphi\ddot{\varepsilon} &= a_\varepsilon r\cos\varphi, \\ 2r\dot{r}\dot{\varphi} + r^2\ddot{\varphi} + r^2\sin\varphi\cos\varphi\left(\dot{\varepsilon}+\omega\right)^2 &= a_\varphi r, \end{aligned} \tag{6.93}$$

having defined

$$\begin{aligned} a_r &= a_r\left(r,\varepsilon,\varphi\right) = \tilde{w}_r + \tilde{a}_r, \\ a_\varepsilon &= a_\varepsilon\left(r,\varepsilon,\varphi\right) = w_\lambda + \tilde{a}_\lambda, \\ a_\varphi &= a_\varphi\left(r,\varepsilon,\varphi\right) = w_\varphi + \tilde{a}_\varphi. \end{aligned} \tag{6.94}$$

Simplifying, the dynamics are finally written as

$$\begin{aligned} \ddot{r} - r\dot{\varphi}^2 - r\cos^2\varphi\left(\dot{\varepsilon}+\omega\right)^2 + \mu/r^2 &= a_r\left(r,\varepsilon,\varphi\right), \\ 2\dot{r}\cos\varphi\left(\dot{\varepsilon}+\omega\right) - 2r\dot{\varphi}\sin\varphi\left(\dot{\varepsilon}+\omega\right) + r\cos\varphi\ddot{\varepsilon} &= a_\varepsilon\left(r,\varepsilon,\varphi\right), \\ 2\dot{r}\dot{\varphi} + r\ddot{\varphi} + r\sin\varphi\cos\varphi\left(\dot{\varepsilon}+\omega\right)^2 &= a_\varphi\left(r,\varepsilon,\varphi\right). \end{aligned} \tag{6.95}$$

Regarding the perturbations, it has been chosen to take into account only the effects due to the non-spherical shape of the Earth, following the method presented in [30]; first, the perturbation is expressed as a potential, in this case a gravitational potential in a reference frame centred in the centre of mass of the attractor from which it has already been subtracted the first order term

$$\begin{aligned} U_g = U_g\left(r,\lambda,\varphi\right) = \frac{\mu}{r}\sum_{l=2}^{\infty}\sum_{m=0}^{l}\left(\frac{R}{r}\right)^l \frac{1}{2^l l!}\left(1-\sin^2\varphi\right)^{m/2} \\ \frac{\mathrm{d}^{[l+m]}}{\mathrm{d}\left(\sin\varphi\right)^{[l+m]}}\left(\sin^2\varphi-1\right)^l\left(C_{l,m}\cos m\lambda + S_{l,m}\sin m\lambda\right), \end{aligned} \tag{6.96}$$

where R is the Earth's radius, and $C_{l,m}$ and $S_{l,m}$ are the coefficients of the series expansion. The components of the acceleration in the spherical reference frame are obtained by applying the gradient operator to the potential

$$\mathbf{a}_g = \frac{\partial U}{\partial r}\hat{r} + \frac{1}{r\cos\varphi}\frac{\partial U}{\partial \lambda}\hat{\lambda} + \frac{1}{r}\frac{\partial U}{\partial \varphi}\hat{\varphi}. \tag{6.97}$$

It has been decided to limit the series expansion to the third order, neglecting the terms C_{21} and S_{21} that are considerably smaller than the others, thus obtaining the following expressions for the perturbing accelerations:

$$\begin{aligned} a_{g_r} = &- 3\frac{\mu R^2}{r^4}\left[\left(\frac{3}{2}\sin^2\varphi - \frac{1}{2}\right)C_{20} + \left(3\cos^2\varphi\right)\left(C_{22}\cos 2\lambda + S_{22}\sin 2\lambda\right)\right] + \\ &- 4\frac{\mu R^3}{r^5}\left\{\left(\frac{5}{2}\sin^3\varphi - \frac{3}{2}\sin\varphi\right)C_{30} + \right. \\ &+ \left(\frac{15}{2}\sin^2\varphi\cos\varphi - \frac{3}{2}\cos\varphi\right)\left(C_{31}\cos\lambda + S_{31}\sin\lambda\right) + \\ &+ \left(15\sin\varphi\cos^2\varphi\right)\left(C_{32}\cos 2\lambda + S_{32}\sin 2\lambda\right) + \\ &\left. + \left(15\cos^3\varphi\right)\left[C_{33}\cos\lambda\left(1 - 4\sin^2\lambda\right) + S_{33}\sin\lambda\left(4\cos^2\lambda - 1\right)\right]\right\} \end{aligned} \tag{6.98}$$

$$\begin{aligned} a_{g_\lambda} = &+ \frac{\mu R^2}{r^4}\left(3\cos^2\varphi\right)\left(2S_{22}\cos 2\lambda - 2C_{22}\sin 2\lambda\right) + \\ &+ \frac{\mu R^3}{r^5\cos\varphi}\left\{\left(\frac{15}{2}\sin^2\varphi\cos\varphi - \frac{3}{2}\cos\varphi\right)\left(S_{31}\cos\lambda - C_{31}\sin\lambda\right) + \right. \\ &+ \left(15\sin\varphi\cos^2\varphi\right)\left(2S_{32}\cos 2\lambda - 2C_{32}\sin 2\lambda\right) + \\ &\left. + \left(15\cos^3\varphi\right)\left[C_{33}\left(12\sin^3\lambda - 9\sin\lambda\right) + S_{33}\left(12\sin^3\lambda - 9\sin\lambda\right)\right]\right\} \end{aligned} \tag{6.99}$$

$$\begin{aligned} a_{g_\varphi} = &+ \frac{\mu R^2}{r^4}\left[\left(3\sin\varphi\cos\varphi\right)C_{20} + \left(6\sin\varphi\cos\varphi\right)\left(C_{22}\cos 2\lambda + S_{22}\sin 2\lambda\right)\right] + \\ &+ \frac{\mu R^3}{r^5}\left\{\left(\frac{15}{2}\sin^2\varphi\cos\varphi - \frac{3}{2}\cos\varphi\right)C_{30} + \right. \\ &+ \left[\frac{3}{2}\sin\varphi + \frac{15}{2}\left(2\sin\varphi\cos^2 - \sin^3\varphi\right)\right]\left(C_{31}\cos\lambda + S_{31}\sin\lambda\right) + \\ &+ \left(15\cos^3\varphi - 30\sin^2\varphi\cos\varphi\right)\left(C_{32}\cos 2\lambda + S_{32}\sin 2\lambda\right) + \\ &+ \left(-45\sin\varphi\cos^2\varphi\right)\left[C_{33}\cos\lambda\left(1 - 4\sin^2\lambda\right) + \right. \\ &\left. \left. + S_{33}\sin\lambda\left(4\cos^2\lambda - 1\right)\right]\right\} \end{aligned} \tag{6.100}$$

It is apparent that the variables λ and φ are always used inside trigonometric functions, and this fact would severely limit the possibility of factorizing the dynamics; to overcome this, the trigonometric functions have been expanded

as Taylor series around the nominal position $(\lambda_n, \varphi_n = 0)$ and truncated to the second order, providing a good starting point for the factorization process; the reader is directed to [30] for the complete Taylor series associated to the perturbations.

6.4.2 Problem set-up

The dynamics of the previous section have been already written in dimensionless units, so that the gravitational parameter of the Earth is $\mu = 1$, the nominal distance of the satellite from the centre of the Earth is $r_n = 1$ and the angular velocity is $\omega = 1$. The state-space form of the problem, having set $x_1 = r$, $x_2 = \varepsilon$, $x_3 = \varphi$, $x_4 = \dot{r}$, $x_5 = \dot{\varepsilon}$ and $x_6 = \dot{\varphi}$, is

$$\begin{cases} \dot{x}_1 = x_4, \\ \dot{x}_2 = x_5, \\ \dot{x}_3 = x_6, \\ \dot{x}_4 = -\dfrac{1}{x_1^2} + x_1 x_6^2 + x_1 \left(x_5 + 1\right)^2 \cos^2 x_3 + a_r, \\ \dot{x}_5 = 2x_6 \left(x_5 + 1\right) \tan x_3 - 2\dfrac{x_4}{x_1} \left(x_5 + 1\right) + \dfrac{1}{x_1 \cos x_3} a_\varepsilon + \dfrac{1}{x_1 \cos x_3} u_\varepsilon, \\ \dot{x}_6 = -2\dfrac{x_4}{x_1} x_6 - \left(x_5 + 1\right)^2 \sin x_3 \cos x_3 + \dfrac{1}{x_1} a_\varphi + \dfrac{1}{x_1} u_\varphi, \end{cases} \tag{6.101}$$

where have been introduced the control accelerations u_ε and u_φ. In order to maintain the nominal state vector

$$\mathbf{x}_n^T = \begin{Bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{Bmatrix}, \tag{6.102}$$

it is necessary to set the final condition $x_1(t_f = \pi) = 1$, otherwise the control would tend to decrease the state vector to zero.The remaining states are left unspecified, to guarantee an optimality in terms of control magnitude, and thus, fuel consumption. To complete the definition of the MCP, the initial condition will be

$$\mathbf{x}_i^T = \begin{Bmatrix} 1 & 0.05^\circ & 0.05^\circ & 0 & 0 & 0 \end{Bmatrix}, \tag{6.103}$$

corresponding to a corner of the typical tolerance window of a geostationary satellite. The unperturbed dynamics have been factorized as

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ a_{41} & 0 & 0 & 0 & a_{45} & a_{46} \\ 0 & 0 & 0 & a_{54} & a_{55} & a_{56} \\ a_{61} & 0 & 0 & a_{64} & a_{65} & a_{66} \end{bmatrix}, \tag{6.104}$$

with

$$
\begin{aligned}
a_{41} &= -\frac{1}{x_1^3} + k_1 x_6^2 + \left(k_2 x_5^2 + 2k_3 x_5 + 1\right) \cos^2 x_3 a, \\
a_{45} &= \left[\left(1 - k_2\right) x_1 x_5 + 2\left(1 - k_3\right)\right] \cos^2 x_3, \\
a_{46} &= \left(1 - k_1\right) x_1 x_6, \\
a_{54} &= -\frac{2}{x_1} - 2\left(1 - k_5\right) \frac{x_5}{x_1}, \\
a_{55} &= 2k_4 x_5 \tan x_3 - 2k_5 \frac{x_4}{x_1}, \\
a_{56} &= 2\left[1 + \left(1 - k_4\right) x_5\right] \tan x_3, \\
a_{61} &= -\frac{\sin 2x_3}{2x_1}, \\
a_{64} &= -2\left(1 - k_6\right) \frac{x_6}{x_1}, \\
a_{65} &= -\left(\frac{1}{2} x_5 + 1\right) \sin 2x_3, \\
a_{66} &= -2k_6 \frac{x_4}{x_1}.
\end{aligned}
$$

The parameters $0 \leq k_i \leq 1$ allow to write many factorizations of the nonlinear dynamics; the $2^6 = 64$ factorizations stemming from assigning to the parameters only the values 0 or 1 will be used to solve the problem. Regarding the perturbations, it is only necessary to sum those 64 factorization with a matrix A_p that expresses a single factorization of the perturbations; details are given in [30]. The remaining matrices are

$$
\mathbf{B} = \begin{bmatrix}
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
\dfrac{1}{x_1 \cos x_3} & 0 \\
0 & \dfrac{1}{x_1}
\end{bmatrix}, \qquad
\mathbf{S} = \begin{bmatrix}
100 & 0 & 0 & 0 & 0 \\
0 & 10 & 0 & 0 & 0 \\
0 & 0 & 100 & 0 & 0 \\
0 & 0 & 0 & 10 & 0 \\
0 & 0 & 0 & 0 & 10
\end{bmatrix}, \tag{6.105}
$$

while $\mathbf{Q} = \mathbf{0}_{6\times 6}$ and $\mathbf{R} = \mathbf{I}_{2\times 2}$. The initial time is set to $t_i = 0$ and the termination tolerance to 10^{-6}.

6.4.3 Results

Of the 64 factorizations employed, 32 two of them result to be always controllable. This is very interesting, since the lack of control along the radial direction could led to a non-controllable system; the issue is tackled by providing a good pool of possible factorizations, among which the algorithm can choose in order to guarantee controllability. The iterations of the

Table 6.7: Iterations of the MASRE algorithm for the station-keeping optimal control problem, CPU time 1529.4 s

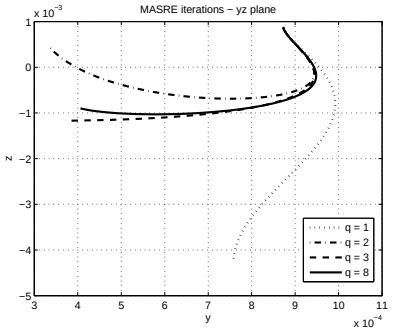
	Error	σ_{min}	J	Matrix
1	5.150×10^{-3}	7.400×10^{-11}	1.539×10^{-7}	$\mathbf{A}_1$
2	5.794×10^{-3}	1.270×10^{-8}	1.731×10^{-7}	$\mathbf{A}_1$
3	3.057×10^{-3}	1.338×10^{-8}	1.750×10^{-7}	$\mathbf{A}_1$
4	7.544×10^{-4}	2.245×10^{-9}	1.691×10^{-7}	$\mathbf{A}_1$
5	1.211×10^{-4}	3.496×10^{-9}	1.707×10^{-7}	$\mathbf{A}_1$
6	1.891×10^{-5}	3.316×10^{-9}	1.708×10^{-7}	$\mathbf{A}_1$
7	3.618×10^{-6}	3.326×10^{-9}	1.706×10^{-7}	$\mathbf{A}_1$
8	9.429×10^{-7}	3.327×10^{-9}	1.707×10^{-7}	$\mathbf{A}_1$

approximate solution are shown in Table 6.7: the sequence converges toward the factorization associated with matrix $\mathbf{A}_1$ from the beginning, and the algorithm stops after only eight iterations, even though the optimization process is very demanding. Regarding the refinement procedure, it must be noticed that the optimal solution is rather different from the approximate one, especially because the MCP's boundary conditions only provide a final value for the radial coordinate; the other coordinate are not specified, so that the basin of attraction of an optimal solution can be quite broad, with the consequence of having a considerable variation of the solution after the refinement. The trajectories of the approximate and optimal solutions are shown in Figure 6.10. The optimal cost function

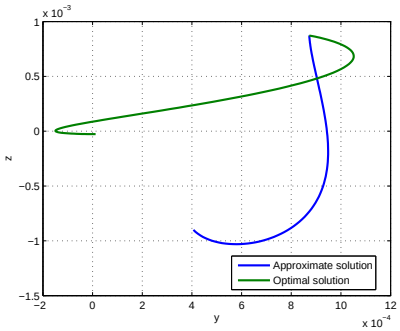
$$J_{opt} = 1.324 \times 10^{-6},$$

is greater than the approximate one, meaning that the approximate solution was violating some of the constraints; the refined solution recovers the consistency, in spite of having a worse cost function.

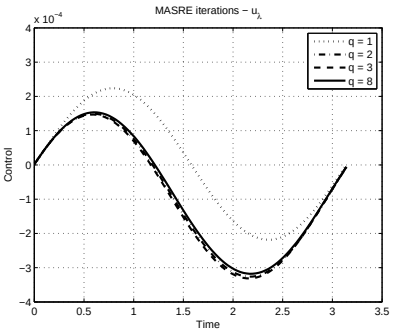
This example has shown the potentiality of the MASRE algorithm: without the need of providing a starting guess of the solution in terms of state and co-state, the problem has been solved even with the lack of radial control; this criticality could not be resolved without the optimization process, as documented in [30].



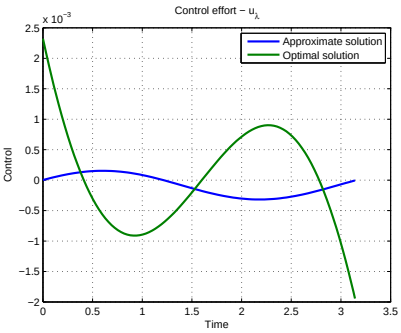
(a) MASRE yz plane iterations



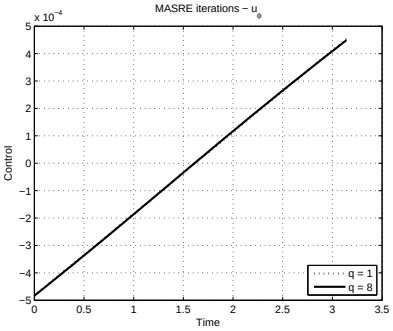
(b) yz plane solutions



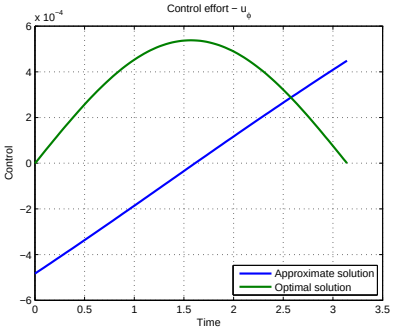
(c) MASRE control iterations - u_λ



(d) Solutions for the control - u_λ



(e) MASRE control iterations - u_φ



(f) Solutions for the control - u_φ

Figure 6.10: Solution of the geostationary station-keeping

Chapter 7

Conclusions

This thesis has been focused on analysing and trying to develop the ASRE algorithm, starting from the theoretical definition of the method; in fact, the practical use of the algorithm is subject to the understanding of its potentialities, that it is interesting to exploit in order to make the most out of the algorithm.

7.1 Main properties

The most interesting aspect of the ASRE algorithm is related to the multiple factorizations that can be obtained starting from a nonlinear vector field; it has been made clear that the solution of the optimal control problem to which the algorithm converges is strongly dependent on the the selected factorization. This fact poses the problem of how to decide which factorization should be used to solve the problem, and, consequently, which are the properties that are looked for when selecting a particular factorization. On the other hand, the ASRE method becomes critical when the procedure encounters singularities, i.e., when the factorization is not defined for some states, or when the pseudo-linear system looses controllability. The multiple-factorizations feature of nonlinear systems has been exploited to tackle those critical aspects, discarding those factorizations that are not consistent with the boundary conditions or not always controllable, and combining linearly the remaining factorizations in order to optimize the controllability at each step of the iteration. The results are quite interesting, showing a correlation between controllability and optimality; still, the resulting method is only able to provide sub-optimal solutions, as confirmed by refining the approximate solution using a fourth-order collocation method. The intrinsic nature of BVPs is of having multiple solutions, and the approximate method, instead of relying on a starting guess to identify a solution, proceeds from a particular factorization out of many possible. As said before, the role played by the starting guess of a classical BVP solver is transferred to the choice of the

factorization, and optimizing this choice produces solutions that are "optimal" in a broader sense, due to the characteristic limitations of any approximate method.

7.2 Critical issues

If, on one hand, the identification of a factorization is a simpler task than providing a starting guess of the solution, on the other hand, the selection of a starting guess allows to finely specify the "kind" of solution that is looked for, e.g., the example in section 6.3.2 shows that the MASRE algorithm is only able to find a very "simple" solution, while direct transcription, employing a specific starting guess, solves the same problem providing a trajectory that is much better from any point of view. The MASRE algorithm simplifies the preliminary task at the cost of loosing insight on the solution that is looked for. Furthermore, even though the factorizations that are not consistent with the boundary conditions are discarded from the start, it is still possible to encounter singularities along the state-space trajectory, and, since it is not known *a priori*, there is no way to overcome this issue, that causes the failure of the integration. Lastly, a limitation of the solver is that only *control affine* system can be treated.

7.3 Future developments

The main improvement that is missing is the possibility of optimizing not only the choice of the factorization of the uncontrolled dynamics, but also the control part, i.e., the matrix $\mathbf{B}$; actually, the methodology would be similar to the one used for optimizing matrix $\mathbf{A}$, and this would allow to consider also those problems that are not control affine, constituting the majority of the cases.

Another possible improvement would be the possibility of managing boundary conditions that are not explicit, i.e., *nonlinear boundary conditions*, expressed with the general form

$$\begin{aligned} \boldsymbol{\chi}\left(\mathbf{x}_i, t_i\right) &= 0, \\ \boldsymbol{\psi}\left(\mathbf{x}_f, t_f\right) &= 0. \end{aligned}$$

This implementation would be useful since it frequently happens that the boundary conditions are expressed as nonlinear equation, whose solution is hard or even impossible to express in closed form.

Lastly, a good control system should also take into account the unavoidable limit posed by the actuators in terms of maximum control effort; the saturation of the control, as well as any other *path constraint* on the state variables, implies the introduction of a saturation function

$$\mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0.$$

Appendix A

User guide to the complete solver

In this appendix it is explained how to use the MATLAB algorithm that has been developed in parallel with this thesis. The main aspects that are covered involve the setting up of the problem's structure and the calibration of the algorithm's settings.

A.1 Problem structure

The data of the problem have to be stored into a single MATLAB function file, whose declaration is

```
function [dyn_fun, A, B, Q, R, S, n, m, k, xi, xf, ti, tf, N, ...
    tol] = Problem
```

where `Problem` identifies the problem file. Next, it is necessary to define the various outputs of this MATLAB function: `dyn_fun` is a string that contains the vectorial expression of the system's uncontrolled dynamics, `B` is the unique, state-dependent control matrix of the control-affine system, `Q` and `R` are the state-dependent weighting matrices of state and control respectively, and `S` is the weighting matrix of the unconstrained final states; e.g., consider the soft-constrained problem

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= x_1 x_2 + u \end{aligned} \tag{A.1}$$

with cost function

$$J = \frac{1}{2} x_{1f}^2 + \frac{1}{2} \int_{t_i}^{t_f} \left(2x_1^2 + x_2^2 + u^2 \right) \mathrm{d}t. \tag{A.2}$$

The file should then contain the following definitions

```
dyn_fun   = '[x2;x1*x2]';
B = '[0;1]';
Q = '[2 0;0 1]';
R = '1';
S = '[1 0;0 0]';
```

The string `S` will be ignored for *hard constrained* problem. Concerning `A`, it is a cell array containing strings that represent the possible factorizations of the uncontrolled dynamics found by the user; referring to equation (A.1), possible factorizations are

$$A_1(\mathbf{x}) = \begin{bmatrix} 0 & 1 \\ 0 & x_1 \end{bmatrix}, \qquad A_2(\mathbf{x}) = \begin{bmatrix} x_2 & 1-x_1 \\ 0 & x_1 \end{bmatrix},$$
$$A_3(\mathbf{x}) = \begin{bmatrix} 0 & 1 \\ x_2 & 0 \end{bmatrix}, \qquad A_4(\mathbf{x}) = \begin{bmatrix} x_2 & 1-x_1 \\ x_2 & 0 \end{bmatrix},$$

so that the file should contain

```
A1 = '[0 1;0 x1]';
A2 = '[x2 1-x1;0 x1]';
A3 = '[0 1;x2 0]';
A4 = '[x2 1-x1;x2 0]';
A = {A1 A2 A3 A4};
```

The provision of factorizations it is actually optional, but still it is advisable to try and compute analytically some of the factorizations for the system. Then, the definition of the problem requires some further data: `n` is the number of states, `m` is the number of control inputs, `ti` and `tf` are the initial and final times. Regarding the boundary conditions, the initial one `xi` must be always and completely assigned while the definition of final condition `xf` depends on the kind of problem to be solved: *hard constrained* requires a completely assigned final condition, *soft constrained* problem requires no final condition and thus `xf` should be left empty, while for *mixed constrained* problems `xf` should contain `NaN` entries for those states that are not prescribed at final time. `k` is the minimum number of factorizations (consistent and controllable) requested to be used inside the algorithm: the factorizations provided by the user will be checked for consistency and controllability, and, if the remaining number of factorizations is less than `k`, the algorithm will try to compute other factorizations. `N` is the number of points on the discrete time grid over which will be provided the approximate solution and `tol` is the termination tolerance of the ASRE algorithm. Below it is shown an example of the complete code for the definition of the problem.

```
function [dyn_fun, A, B, Q, R, S, n, m, k, xi, xf, ti, tf, N, ...
    tol] = Problem

k = 4; N = 101; tol = 1e-6;
ti = 0;     tf = 5;
xi = [2;1]; xf = [4;1];
n = 2; m = 1;

dyn_fun = '[x2;x1*x2]';
A1 = '[0 1;0 x1]';
A2 = '[x2 1-x1;0 x1]';
A3 = '[0 1;x2 0]';
A4 = '[x2 1-x1;x2 0]';
A = {A1 A2 A3 A4};
B = '[0;1]';
Q = '[2 0;0 1]';
R = '1';
S = '[1 0;0 0]';

end
```

The file must then be saved inside a `Problems` sub-folder of the main folder containing the algorithm.

A.2 Algorithm execution and options

Once the problem has been defined and saved in the proper location, it is necessary to open the file `SolveProblem.m` and modify it accordingly: the string `TheProblem` must contain the name of the problem definition file, e.g., `TheProblem = 'Problem'`. Next, there are a few options that can be set:

- `flag_start` is a string that can be set either to `cold` or to `warm`; a *cold start* will execute the whole algorithm, while a *warm start* will only execute the TPBVP solver starting from a previously stored solution of the ASRE algorithm.
- `flag_check` is a string that can be set to `on`, `cons`, `ctrb` or `off`: with `flag_check = 'on'` the algorithm will check both the consistency and the controllability of the factorizations; with `flag_check = 'cons'` the algorithm will only check for consistency; with `flag_check ... = 'ctrb'` the algorithm will only check for controllability; with with `flag_check = 'off'` the algorithm will check neither for consistency nor for controllability.
- `bvp_solver` is a string that allows to select either a fourth-order solver (`bvp_solver = 'bvp4c'`) or a fifth-order solver (`bvp_solver = ... 'bvp5c'`).

During the execution of the algorithm, the command window shows different informations: the results of the consistency and controllability checks, how

many factorizations have been automatically computed and a table that, for each iteration, contains the error, the minimum singular-value of the controllability gramian, the cost function and the matrices toward which the optimization has converged.

A.3 Warnings and errors

Warnings are shown whenever happens any of the following:

- The origin is not an equilibrium point of the uncontrolled dynamics, so Bass algorithm cannot be used and it is adopted "brute force method", which consists in bringing out the states directly
- Bass algorithm is not able to compute a factorization of the uncontrolled dynamics, so that it is adopted the "brute force method".
- The algorithm is not able to find factorizations that are consistent and/or controllable.
- The algorithm can find a number of factorizations that are consistent and controllable that is less than what requested by the user.
- Only one factorization is available, so that no optimization process is possible.

Error messages will be displayed and the algorithm will stop for any of the following cases:

- The `flag_check` string has not been assigned properly.
- No factorization is available.

A.4 Outputs

The output of the function `CompleteSolver.m` consists of four elements:

- `sol_approx` is a structure that contains the solution of the ASRE algorithm in terms of states, co-states, controls and diagnostics.
- `sol_BVP` is a structure that contains the "raw" solution of the BVP solver, i.e., the solution of the problem

$$\dot{\mathbf{z}} = \mathbf{f}(\mathbf{z}, t), \tag{A.3}$$

over the interval $[a, b]$ subject to two-point boundary value conditions $\mathbf{g}\left(\mathbf{z}(a), \mathbf{z}(b)\right) = 0$, where $\mathbf{z}^T = \begin{Bmatrix} \mathbf{x} & \boldsymbol{\lambda} \end{Bmatrix}^T$.

- `sol_opt` is a structure that contains the solution of the BVP solver in terms of states, co-states, controls and diagnostics.
- `warmstart` is a structure that contains the data needed for a warm start.

Acronyms

ASRE Approximating Sequence of Riccati Equation.

BVP Boundary Value Problem.

CRTBP Circular Restricted Three-Body Problem.

HCP Hard Constrained Problem.

IVP Initial Value Problem.

LQR Linear Quadratic Regulator.

LTV Linear Time-Variant.

MASRE modified ASRE.

MCP Mixed Constrained Problem.

NLP Nonlinear Programming.

ODE Ordinary Differential Equation.

SCP Soft Constrained Problem.

SDC State Dependent Coefficient.

SDRE State Dependent Riccati Equation.

STM State-Transition Matrix.

TPBVP Two-Point Boundary Value Problem.

Bibliography

[1] A. Bryson & Y. Ho. *Applied Optimal Control.* John Wiley & Sons, Washington, DC, 1975

[2] J. Betts. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming.* SIAM, Philadelphia, PA, 2010

[3] L. Pontryagin. *The Mathematical Theory of Optimal Processes.* John Wiley & Sons, New York, NY, 1962

[4] A. Isidori. *Nonlinear Control Systems, Third Edition.* Springer, New York, NY, 1995

[5] B. Friedland. *Control System Design: an Introduction to State-Space Methods.* McGraw-Hill, New York, NY, 1986

[6] J. Stoer, R. Bulirsch. *Introduction to Numerical Analysis.* Springer, New York, NY, 2002

[7] G. P. Sutton, O. Biblarz. *Rocket Propulsion Elements.* Wiley & Sons, Washington, DC, 2010

[8] H. Curtis. *Orbital Mechanics for Engineering Students.* Elsevier, Oxford, GB, 2005

[9] G. Fasano & J. D. Pinter *Editors. Modelling and Optimization in Space Engineering.* Springer, New York, NY, 2013

[10] S. P. Banks & K. Dinesh. *Approximate Optimal Control and Stability of Nonlinear Finite- and Infinite-Dimensional Systems.* Annals of Operations Research, vol. 98, 2000

[11] T. Çimen & S. P. Banks. *Nonlinear Optimal Tracking Control with Application to Super-Tankers for Autopilot Design.* Automatica, vol. 40, 2004

[12] T. Çimen & S. P. Banks. *Global Optimal Feedback Control for General Nonlinear Systems with Nonquadratic Performance Criteria.* Systems & Control Letters, vol. 53, 2004

[13] F. Topputo & F. Bernelli-Zazzera. *Approximate Solutions to Nonlinear Optimal Control Problems in Astrodynamics.* Proceedings of the Conference on Dynamics and Control of Space Systems, Porto, PT, 2012.

[14] F. Topputo. *Le varietà invarianti del problema dei tre corpi ristretto: uno strumento innovativo per trasferimenti interplanetari a basso costo.* M. Sc. Thesis, 2003.

[15] T. Çimen. *Systematic and Effective Design of Nonlinear Feedback Controllers via the State-Dependent Riccati Equation (SDRE) Method.* Annual Reviews in Control, vol. 34, 2010

[16] S. C. Beeler. *State-Dependent Riccati Equation Regulation of Systems with State and Control Nonlinearities.* NIA Report No. 2004-8

[17] W. Bass. *ρ-synthesis: a "rhobustness" Margin for Unstructured Nonlinear Time-varying Deviations.* Proceedings of the Conference on Decision and Control, Brighton, GB, 1991

[18] K. D. Hammett, et al. *Controllability Issues in Nonlinear State-Dependent Riccati Equation Control.* Journal of Guidance, Control and Dynamics, vol. 21, 1998

[19] W. Kang & L. Xu. *A Quantitative Measure of Observability and Controllability.* Proceedings of the Conference on Decision and Control, Shangai, P.R.C., 2009

[20] C. C. Paige. *Properties of Numerical Algorithms Relating to Controllability.* IEEE Transactions on Automatic Control, vol. 26, 1981

[21] R. Eising. *Between Controllable and Uncontrollable.* System & Control Letters, vol. 4, 1984

[22] B. Dumitrescu, et al. *Computing the Controllability Radius: a Semidefinite Programming Approach.* IET Control Theory and Applications, vol. 3, 2009

[23] S. Nazari & B. Shafai. *Robust SDC Parameterization for a Class of Extended Linearization Systems.* Proceedings of the American Control Conference, San Francisco, CA, 2011

[24] C. R. Hargraves & S. W. Paris. *Direct Trajectory Optimization Using Nonlinear Programming and Collocation.* Journal of Guidance, Control and Dynamics, vol. 10, 1987

[25] P. J. Enright & B. A. Conway. *Discrete Approximations to Optimal Trajectories Using Direct Transcription and Nonlinear Programming.* Journal of Guidance, Control and Dynamics, vol. 15, 1992

[26] B. A. Conway & K. M. Larson. *Collocation Versus Differential Inclusion in Direct Optimization.* Journal of Guidance, Control and Dynamics, vol. 21, 1998

[27] C. Park & D. J. Scheeres. *Solutions of the Optimal Feedback Control Problem using Hamiltonian Dynamics and Generating Functions.* Proceedings of the Conference on Decision and Control, Maui, HI, 2003

[28] C. Park, et al. *Solving Optimal Control Problems with Generating Functions.* Proceedings of the AAS/AIAA Astrodynamics Specialist Meeting, Big Sky, MT, 2003

[29] L. F. Shampine et al. *Solving Boundary Value Problems for Ordinary Differential Equations in MATLAB with* `bvp4c`. Available at `http://www.mathworks.com/bvp_tutorial`

[30] M. Pasta. *Station keeping di satelliti geostazionari mediante controllo non lineare.* M. Sc. Thesis, 2010