Adaptive Time–Mesh Refinement in Optimal Control Problems with State Constraints *

Luís Tiago Paiva

SYSTEC–ISR, Faculdade de Engenharia, Universidade do Porto
4200-465, Porto, Portugal
ltpaiva@fe.up.pt

Fernando A.C.C. Fontes

SYSTEC–ISR, Faculdade de Engenharia, Universidade do Porto
4200-465, Porto, Portugal
faf@fe.up.pt

Abstract

When using direct methods to solve continuous-time nonlinear optimal control problems, regular time meshes having equidistant spacing are most frequently used. However, in some cases, these meshes cannot cope accurately with nonlinear behaviour and increasing uniformly the number of mesh nodes may lead to a more complex problem. We propose an adaptive time–mesh refinement algorithm, considering different levels of refinement and several mesh refinement criteria. Namely, we use information of the adjoint multipliers to decide where to refine further. This technique is here tested to solve two optimal control problems. One involving nonholonomic vehicles with state constraints which is characterized by having strong nonlinearities and by discontinuous controls; the other is also a nonlinear problem of a compartmental SEIR system. The proposed strategy leads to results with higher accuracy and yet with lower overall computational time, when compared to results obtained by meshes having equidistant spacing. We also apply the necessary condition of optimality in the form of the Maximum Principle of Pontryagin to characterize the solution and to validate the numerical results.

1 Introduction

Over the past decades, direct methods have become increasingly useful when computing the numerical solution of nonlinear optimal control problems (OCP) [1]. These methods directly optimize the discretized OCP without using the maximum principle and they are known to provide a robust approach for a wide variety of optimal control problems.

*Paper submitted to Discrete and Continuous Dynamical Systems (to appear 2015).

In a direct collocation method, the control and the state are discretized in an appropriately chosen mesh of the time interval. Then, the continuous–time OCP is transcribed into a finite–dimensional nonlinear programming problem (NLP) which can be solved using widely available software packages [13]. Most frequentely, in the discretization procedure, regular time meshes having equidistant spacing are used. However, in some cases, these meshes are not the most adequate to deal with nonlinear behaviours. One way to improve the accuracy of the results, while maintaining reasonable computational time and memory requirement, is to construct a mesh having different time steps. The best location for the smaller steps sizes is, in general, not known a priori, so the mesh will be refined iteratively.

In a mesh–refinement procedure the problem is solved, typically, in an initial coarse uniform mesh in order to capture the basic structure of the solution and of the error. Then, this initial mesh is repeatedly refined according to a chosen strategy until some stopping criterion is attained.

Several mesh refinement methods employing direct collocation methods have been described in the recent years. In [1] and [3] a mesh refinement procedure involving an integer programming technique is developed for changing the discretization in order to improve the accuracy of the approximation. In [19] a density function is used to generate a fixed-order mesh on which the problem is solved. In [16] an approach based on varying the order of the approximation in each mesh interval and using the exponential convergence rate of a Gaussian quadrature collocation method is reported. In [14] and [15], an adaptive mesh refinement strategy based on block–structured refinement method for solving continuous–time nonlinear OCP is presented. It is a purely direct method approach and the convergence is achieved by increasing the number of nodes and by selecting their placement according the refinement criterion. In this algorithm, just one refinement criterion can be defined and it coincides with the stopping criterion.

In this paper, we propose an adaptive mesh refinement algorithm for OCP. This algorithm improves on the ideas advanced in [14, 15]. There are three new main features in this algorithm: (a) we introduce several levels of refinement, obtaining a multi–level time–mesh in a single iteration – such concept allows us to implement the nodes collocation in a faster and clever way; (b) we also introduce different refinement criteria – the relative error of the primal variables, the relative error of the dual variables and a combination of both criteria can be used in the refinement procedure; (c) we consider distinct criteria for refining the mesh and for stopping the refinement procedure – the refinement strategy can be driven by the information given by the dual variables and it can be stopped according to the information given by the primal variables. Here, we show that there are advantages in choosing the error of the adjoint multipliers as a refinement criterion, namely in lowering the computational time. This is because the adjoint multipliers are solution to a linear differential equation system, easily solved with high accuracy, and because they give sensitivity information. It is still a direct method approach, although we use some information given by necessary conditions, namely, the adjoint differential equation, which is use in the indirect methods context. To decrease the overall computational time, the solution computed in the previous iteration is used as a warm start in the next one, which proved to be of major importance to improve computational efficiency.

In order to validate the results obtained for this problem, we apply the Maximum Principle of Pontryagin. This analysis allows us to characterize the optimal

trajectory and control and it also provides us the differential equation system for the multipliers. This last information is needed to estimate the error in the multipliers for one of the refinement criteria.

This technique is applied to solve two different problems, both with pathwise state constraints: (P_1) which involves nonholonomic systems [9], namely a car–like system problem [14], and (P_2) involving a compartmental SEIR model [4, 10, 12] which describes the spreading of an infectious disease. The results show solutions with higher accuracy obtained in a overall computational time that is lower when compared to the ones computed with a mesh having equidistant spacing and to the ones computed with a mesh designed as suggested in [14].

When using this strategy, an OCP is solved using less nodes in the overall procedure which revert in significant savings in memory and computational cost. In addition, there is no need to decide *a priori* an adequate mesh meeting our accuracy specifications, which is a major advantage of this procedure. With this mesh–refinement strategy, nonlinear OCP solvers can be used in real–time optimization, in particular in model predictive control [6], since approximate solutions can be produced even when the optimizer is interrupted at an early stage.

This paper is organized as follows. Section 2 states the general form of the optimal control problem considered as well as the corresponding necessary conditions of optimality in the form of a Maximum Principle. Section 3 describes the proposed time–mesh refinement strategy and the algorithm to implement it. Both problems (P_1) and (P_2) are described in section 4 along with the numerical results highlighting the features of the algorithm proposed. In section 5, we apply the necessary condition of optimality to characterize the solution and to validate the numerical results. We end with some concluding remarks.

2 The Optimal Control Problem

Let us consider the following optimal control problem, in Bolza form, with input and state constraints [17]:

$$\text{Minimize } \mathrm{J}(\mathbf{x},\mathbf{u}) = \int_{t_0}^{t_f} \mathrm{L}\left(t,\mathbf{x}(t),\mathbf{u}(t)\right)\mathrm{d}t + \mathrm{G}\left(\mathbf{x}(t_0),\mathbf{x}(t_f)\right)$$

subject to

- the dynamic constraints

$$\dot{\mathbf{x}}(t) = \mathbf{f}(t,\mathbf{x}(t),\mathbf{u}(t)) \quad \text{a.e. } t \in [t_0,t_f]\ ,$$

- the input constraints

$$\mathbf{u}(t) \in \mathbb{U}(t) \subset \mathbb{R}^m \quad \text{a.e. } t \in [t_0,t_f]\ ,$$

- the pathwise state constraint

$$\mathbf{h}\left(\mathbf{x}(t)\right) \leq 0 \quad \forall t \in [t_0,t_f]\ ,$$

and

- the end–point constraints

$$\mathbf{x}(t_0) \in \mathbb{X}_0 \subset \mathbb{R}^n \quad \text{and} \quad \mathbf{x}(t_f) \in \mathbb{X}_1 \subset \mathbb{R}^n \,,$$

where $\mathbf{x} : [t_0, t_f] \to \mathbb{R}^n$ is the state, $\mathbf{u} : [t_0, t_f] \to \mathbb{R}^m$ is the control and $t \in [t_0, t_f]$ is time. The functions involved comprise the running cost $\mathrm{L} : [t_0, t_f] \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$, the terminal cost $\mathrm{G} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$, the dynamic function $\mathbf{f} : [t_0, t_f] \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ and the state constraint $\mathbf{h} : \mathbb{R}^n \to \mathbb{R}^k$.

In this paper we use a smooth version of the Maximum Principle for state constrained problems which is valid under the following hypotheses. There exists a scalar $\xi > 0$ such that:

H1. $\mathbf{f}(\cdot, \mathbf{x}, \cdot)$ is $\mathcal{L} \times \mathcal{B}^m$ measurable for fixed $\mathbf{x}$;

H2. $\mathbf{f}(t, \cdot, \mathbf{u})$ is continuously differentiable on $\bar{\mathbf{x}} + \xi\,\mathbb{B}$, $\forall \mathbf{u} \in \mathbb{U}$, a.e. $t \in [t_0, t_f]$;

H3. There exists $C_u > 0$ such that $||\mathbf{f}(t, \mathbf{x}, \cdot)|| \leq C_u$ for $\bar{\mathbf{x}} + \xi\,\mathbb{B}$, $\forall \mathbf{u} \in \mathbb{U}$, a.e. $t \in [t_0, t_f]$;

H4. G is continuously differentiable on $\bar{\mathbf{x}} + \xi\,\mathbb{B}$;

H5. $\mathbb{U}$ is compact;

H6. $\mathbf{h}$ is continuously differentiable on $\bar{\mathbf{x}} + \xi\,\mathbb{B}$.

Here $\mathbb{B}$ denotes the closed unit ball.

A feasible process $(\mathbf{x}^*, \mathbf{u}^*)$ is a $W^{1,1}$ *local minimizer* if there exists $\delta > 0$ such that $(\mathbf{x}^*, \mathbf{u}^*)$ minimizes $\mathrm{J}(\mathbf{x}, \mathbf{u})$ for all feasible processes $(\mathbf{x}, \mathbf{u})$ which satisfy $\| \mathbf{x} - \mathbf{x}^* \|_{W^{1,1}} \leq \delta$ [17].

Let us consider the Hamiltonian

$$\mathrm{H}\left(t, \mathbf{x}(t), \mathbf{p}(t), \mathbf{u}(t)\right) = \mathbf{p}(t) \cdot \mathbf{f}(t, \mathbf{x}(t), \mathbf{u}(t)) - \lambda \mathrm{L}\left(t, \mathbf{x}(t), \mathbf{u}(t)\right) \,,$$

a smooth version of the Maximum Principle for state constrained problems in [17, p. 329], and its remark d) in page 331.

Theorem 2.1 *Let $(\mathbf{x}^*, \mathbf{u}^*)$ be a $W^{1,1}$ local minimizer of OCP. Assume hypotheses (H1)–(H6) are satisfied. Then, there exist an absolutely continuous function $\mathbf{p} \in W^{1,1}([t_0, t_f]\,; \mathbb{R}^n)$, a scalar $\lambda \geq 0$ and positive Radon measures $\mu_i \in C^{\oplus}\left([t_0, t_f]\right)$, $i = 1, \ldots, k$ satisfying*

- *the nontriviality condition*

$$(\mathbf{p}, \mu, \lambda) \neq (0, 0, 0) \,, \tag{NT}$$

- *the adjoint system*

$$-\dot{\mathbf{p}}(t) = \nabla \mathrm{H}\left(t, \mathbf{x}^*(t), \mathbf{q}(t), \mathbf{u}^*(t)\right) \,, \tag{AS}$$

- *the transversality condition*

$$(\mathbf{p}(t_0), -\mathbf{q}(t_f)) \in \lambda\, \mathrm{G}_{\mathbf{x}}\left(\mathbf{x}^*(t_0), \mathbf{x}^*(t_f)\right) + N_{\mathbb{X}_0 \times \mathbb{X}_1}\left(\mathbf{x}^*(t_0), \mathbf{x}^*(t_f)\right) \,, \tag{T}$$

- *the Weierstrass condition*

$$\mathrm{H}\left(t, \mathbf{x}^*(t), \mathbf{q}(t), \mathbf{u}^*(t)\right) = \max_{\mathbf{u}\in\mathbb{U}(t)} \mathrm{H}\left(t, \mathbf{x}^*(t), \mathbf{q}(t), \mathbf{u}\right) , \tag{WC}$$

- *the complementary slackness condition*

$$\mathrm{supp}\{\mu_i\} \subset \{t : h_i\left(\mathbf{x}(t)\right) = 0\} , \tag{CS}$$

and $\mathbf{q} : [t_0, t_f] \to \mathbb{R}^n$ *is a normalized function with bounded total variation defined as*

$$\mathbf{q}(t) = \begin{cases} \mathbf{p}(t) + \displaystyle\int_{[t_0,t)} \sum_{i=1}^{k} \nabla h_i(\mathbf{x}^*(s)) \mathrm{d}\mu_i(s) & t \in [t_0, t) \\ \mathbf{p}(t_f) + \displaystyle\int_{[t_0,t_f]} \sum_{i=1}^{k} \nabla h_i(\mathbf{x}^*(s)) \mathrm{d}\mu_i(s) & t = t_f . \end{cases} \tag{1}$$

We also use a direct method approach where the control and the state are discretized in a chosen mesh in the time interval. Then, the continuous–time optimal control problem is transcribed into a finite–dimensional nonlinear programming problem via Hermite–Simpson method [2]. To solve this nonlinear programming problem we use a numerical solver – IPOPT [18].

3 Adaptive Mesh Refinement Algorithm

The strategy proposed here is an improvement of the one reported in [14, 15] and, again, it is based on block-structured adaptive refinement method which became popular within fluid mechanics since multigrid algorithms can be used for time and space domains.

In this approach, we are concerned in solving the problem, in a first step, on a coarse mesh and, according to a decision based on some refinement criteria, the mesh is refined locally or entirely, using different levels of refinement. The discretized problem is then solved on the new refined mesh using information from the coarser mesh solution of the previous step. According to this strategy, we do not need to remove nodes.

3.1 Single–level and Multi–level Mesh Refinement

The adaptive mesh refinement process starts by discretising the time interval $[t_0, t_f]$ in a coarse mesh, π_0, containing N_0 equidistant nodes. After being transcribed into a NLP problem, the OCP is solved in this coarse mesh to catch the main structure of the problem. Then, the mesh is progressively refined. According to some refinement criteria, the mesh is divided in K mesh intervals

$$\mathcal{S}_k = [\tau_{k-1}, \tau_k[, \quad k = 1, \ldots, K-1 \qquad \text{and} \qquad \mathcal{S}_K = [\tau_{K-1}, \tau_K]$$

where $(\tau_0, \ldots, \tau_K)$ coincide with nodes. These mesh intervals $\mathcal{S}_k$ form a partition of the time interval, that is,

$$\bigcup_{k=1}^{K} \mathcal{S}_k = [t_0, t_f] \quad \text{and} \quad \bigcap_{k=1}^{K} \mathcal{S}_k = \emptyset ,$$

while the mesh nodes have the property that

$$\tau_0 < \tau_1 < \tau_2 < \ldots < \tau_K \, .$$

According to the algorithm reported in [14], the subintervals $\mathcal{S}_k$ that verify the refinement criteria are refined by adding a fixed number $\mathcal{N}$ of equidistant nodes between each two mesh points in such way that the refined mesh π_{j+1} contains the nodes of the prior one π_j. This property is an important feature in block–structured schemes. The procedure is repeated until the stopping criterion is achieved. We also considered a more conservative approach by refining the neighbours of $\mathcal{S}_k$, *i.e.*, $\mathcal{S}_{k-1}$ and $\mathcal{S}_{k+1}$. The resulting mesh using this strategy is denoted further ahead by π_R.

Now, we present an improved version of this algorithm by introducing different levels of refinement in a single iteration. After selecting the intervals $\mathcal{S}_k$ that verify the refinement criteria, they are divided into smaller subintervals according to the user–defined levels of refinement

$$\bar{\varepsilon} = [\varepsilon_1, \, \varepsilon_2, \, \ldots, \, \varepsilon_m] \, .$$

For example, the higher levels can be defined as multiple powers of 10 of the first level $\bar{\varepsilon} = \left[1, \, 10, \, 10^2, \, \ldots, \, 10^m\right] \varepsilon_1$.

A subinterval $\mathcal{S}_{k,i}$ is at the i^{th} level of refinement if

$$S_{k,i} = \{t \in S_k \ : \ \varepsilon(t) \in [\varepsilon_i, \varepsilon_{i+1}[\} \tag{2}$$

for $i = 1, \ldots, m$, and it will be refined by adding $\mathcal{N}^i$ of equidistant nodes between each two mesh points. This procedure adds more node points to the subintervals in higher levels of refinement, corresponding to higher errors, and it adds less node points to those in lower refinement levels.

By defining several levels of refinement, we get a multi–level time–mesh in a single iteration. The resulting mesh using this strategy is denoted further ahead by π_{ML}.

3.2 Refinement and Stopping Criteria

In order to proceed with the mesh refinement strategy, we have to define some refinement criteria and a stopping criterion. We consider three refinement criteria:

1. the estimate of the relative error of the trajectory (primal variables) ($\varepsilon_{\mathbf{x}}$),
2. the estimate of the relative error of the adjoint multipliers (dual variables) ($\varepsilon_{\mathbf{q}}$),
3. a combination of both criteria.

For the stopping criterion, we consider a threshold for the relative error of the trajectory.

In the first refinement criterion, the relative error estimate is, at each time, the difference between the obtained state trajectory and an higher order approximation of the solution of the dynamic differential equation. In this case, the solution is given by piecewise cubic polynomials using Hermite interpolation and, then, it is integrated using the Romberg quadrature. At each refinement iteration, the local

error ($\varepsilon_{\mathbf{x}}$) is computed and this information is taken into account when deciding if the refinement procedure should continue.

In the second case, we consider the multipliers $\mathbf{q}_{\text{MP}}$ which are solution of the differential equation system (AS), (T) and (1) given by the Maximum Principle 2.1, and we also consider the multipliers $\mathbf{q}_{\text{KKT}}$ obtained by applying the Kuhn–Tucker conditions to nonlinear optimization problem which results from the transcription of the optimal control. The relative error estimate is, at each time, the difference between the multipliers $\mathbf{q}_{\text{KKT}}$ computed by the numerical solver and $\mathbf{q}_{\text{MP}}$ computed by integrating numerically the adjoint equation given by the Maximum Principle. This criterion is chosen because these multipliers give sensitivity information. Furthermore, $\mathbf{q}_{\text{MP}}$ are solution to a linear differential equation system, which can be easily solved in a faster way and with higher accuracy. At each refinement step, the local error of the multipliers ($\varepsilon_{\mathbf{q}}$) is evaluated

$$\varepsilon_{\mathbf{q}} = ||\mathbf{q}_{\text{MP}} - \mathbf{q}_{\text{KKT}}||$$

and the procedure selects which time intervals should be further refined.

In the last case, we use both refinement criteria simultaneously and the procedure will continue until the stopping criterion is satisfied.

As stopping criterion, we consider the L^∞ norm of the relative error of the primal variables ($\varepsilon_{\mathbf{x}}$). Even when using the relative error of the adjoint multipliers ($\varepsilon_{\mathbf{q}}$) as refinement criteria, we still need to estimate the error on the trajectory since it is the stopping criterion. However, we do not need to compute $\varepsilon_{\mathbf{x}}(t)$ for all t – as in the case of the refinement criteria – but just an estimate of $\left|\left|\varepsilon_{\mathbf{x}}^{(j)}\right|\right|_\infty$ which is much faster to obtain.

3.3 Warm Start

Since the proposed procedure increases the number of nodes, more computational time would be expected. To decrease the CPU time, when going from a coarse mesh to a refined one progressively, the previous solution is used as a warm start for the next iteration. To create this warm start, the solution obtained in the coarse mesh is projected in the refined one using the cubic Hermite interpolation. This action proved to be vital in the decreasing of the overall computational time. In particular, we notice that the number of iterations of the NLP solver remains within the same order of magnitude when we considerably increase the number of nodes.

3.4 Algorithm Implementation

The overall procedure is described in Algorithm 1. To test the algorithm, the proposed procedure was implement in MATLAB R2008a combined with the Imperial College London Optimal Control Software – ICLOCS – version 0.1b [5]. ICLOCS is an optimal control interface and it uses the IPOPT – Interior Point OPTimizer – solver, which is an open-source software package for large-scale nonlinear optimization [18]. The problems are solved in a computer with a Intel™ Core© 2 CPU 6600@2.40 GHz.

```
Data: Cost functions, dynamics, constraints, initial/terminal boundaries,
      parameters, refinement and stopping criteria $\left(\varepsilon_{\mathbf{x}}^{\max}$ and $\varepsilon_{\mathbf{q}}^{\max}\right)$
Result: optimal trajectories, controls
initialization;
select a time–mesh;
discretize and transcribe the OCP;
solve the NLP;
estimate the discretization error – $\varepsilon_{\mathbf{x}}^{(0)}$;
estimate the error on the multipliers – $\varepsilon_{\mathbf{q}}^{(0)}$;
while stopping criteria not met do
    select the mesh subintervals $\mathcal{S}_{j,i}$ to be refined ;
    apply the discretization scheme according to the multi–level refinement
    criteria;
    transcribe the OCP ;
    create warm start;
    solve the NLP;
    estimate the discretization error – $\varepsilon_{\mathbf{x}}^{(j)}$;
    estimate the error on the multipliers – $\varepsilon_{\mathbf{q}}^{(j)}$;
end
```

Algorithm 1: Adaptive time–mesh refinement algorithm considering both refinement criteria

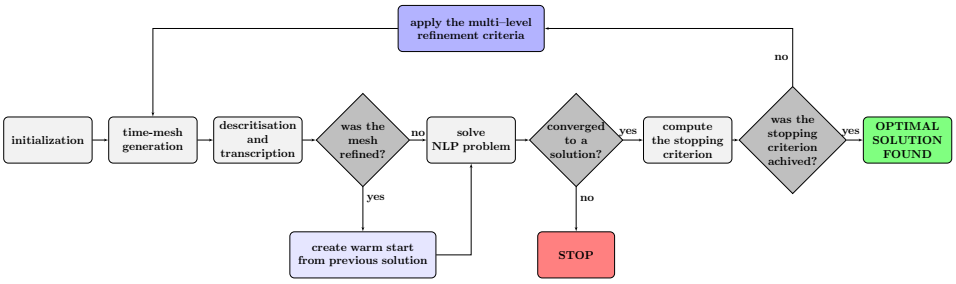


Figure 1: Adaptive time–mesh refinement diagram

4 Application

To test and to validate the proposed algorithm, a problem involving a nonholonomic system [9, 14] and another one regarding the SEIR model [4, 10, 12] are solved considering, in both cases, a pathwise state constraint.

4.1 Car–like System

A car–like system that moves in a plane generally has three degrees of freedom: translation along the two axes in the plane and rotation about the axis perpendicular to the plane. Nevertheless, these vehicles cannot move freely in all three degrees of motion due to their steering constraints. These kind of models are highly nonlinear and, for that reason, it is expected that a refined mesh having non equidistant spacing is more suitable. Such problems belong to the class of nonholonomic systems [9]. It is known that they require discontinuous controls. So, methods in which the parameterizations of the control are based on polynomial, such as pseudo-spectral methods, are less adequate.

In Fig. 2, the geometry of a car–like system is given. For a given time t, $(x(t), y(t))$ is the position of mid-point of the axle connecting the rear wheels, $\psi(t)$ is the yaw angle, $\delta(t)$ is the steering angle and l is the wheelbase of the vehicle, *i.e.*, the distance between its front and rear wheels. We consider also the curvature $c(t)$ which relates to the steering angle δ and the minimum turning radius by

$$c(t) = \frac{\tan(\delta(t))}{l} \quad \text{and} \quad R_{\min} = \frac{1}{|c_{\max}|}\,.$$

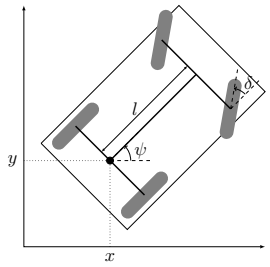


Figure 2: Car–like system geometry

Let us consider $t \in [0, 10]$, in seconds, $\mathbf{x}(t) = (x(t), y(t), \psi(t))$ and $\mathbf{u}(t) = (u(t), c(t))$. Aiming minimum energy, the car–like system problem (P_1) can be stated as:

$$\text{Minimize} \int_0^{10} u^2(t)\mathrm{dt} \tag{3}$$

subject to

- the dynamic constraints

$$\begin{aligned} \dot{x}(t) &= u(t)\cos(\psi(t)) && \text{a.e. } t \in [0, 10] \\ \dot{y}(t) &= u(t)\sin(\psi(t)) && \text{a.e. } t \in [0, 10] \\ \dot{\psi}(t) &= u(t)\,c(t) && \text{a.e. } t \in [0, 10]\,, \end{aligned} \tag{4}$$

where $u(t)$ is the speed and $c(t)$ is the curvature,

- the input constraints

$$\begin{aligned} 0 \leq u(t) \leq 1 \qquad & \text{a.e. } t \in [0, 10] \\ -0.7 \leq c(t) \leq 0.7 \qquad & \text{a.e. } t \in [0, 10]\,, \end{aligned}$$

- the end–point constraints

$$\mathbf{x}(0) = \mathbf{x}_0 = (x_0,\, y_0,\, \psi_0) = (0, 0, 0) \tag{5}$$

$$\mathbf{x}(10) \in \mathbb{X}_1 = \left\{(x, y, \psi) : (x - x_f)^2 + (y - y_f)^2 + (\psi - \psi_f)^2 \leq r^2\right\}, \tag{6}$$

where $r^2 = 0.1$ and $\mathbf{x}_f = (x_f,\, y_f,\, \psi_f) = (10, 0, 0)$ is a user–defined target point, and

- the pathwise state constraint

$$h\left(\mathbf{x}(t)\right) = (\bar{y} - y(t)) - 10\left(\bar{x} - x(t)\right)^2 \leq 0\,, \quad \forall t \in [0, 10]\,, \tag{7}$$

where $(\bar{x}, \bar{y}) = (5, 1)$ and $k = 10$.

The goal is to drive this car–like system with minimum energy from $\mathbf{x}_0$ to some point near $\mathbf{x}_f$ according to the terminal condition (6) while overcoming the state constraint (7).

4.2 The SEIR Model

The SEIR model is a compartmental model that describes the spreading of an infectious disease among a population (N) by dividing it into four different compartments: susceptible (S), exposed but not yet infectious (E), infectious (I) and recovered (R) [4, 10, 12]. SEIR models can represent many human infectious diseases such as measles, pox, flu or dengue. According to [4], we can add to the dynamical system given in [12] an extra variable (W), which stands for the number of vaccinated people and which is governed by the differential equation $\dot{W}(t) = u(t)S(t)$. Then, the ordinary differential equation governing $\dot{R}$ can be replaced with the one for $\dot{N}$.

We consider $t \in [0, 20]$, in years, $\mathbf{x}(t) = (S(t), E(t), I(t), N(t), W(t))$ and $\mathbf{u}(t) = u(t)$. This problem (P_2) can be stated as:

$$\text{Minimize} \int_0^{20} 0.1\, I(t) + u^2(t)\mathrm{dt} \tag{8}$$

subject to

- the dynamic constraints

$$\begin{aligned} \dot{S}(t) &= bN(t) - dS(t) - cS(t)I(t) - u(t)S(t) && \text{a.e. } t \in [0, 20] \\ \dot{E}(t) &= cS(t)I(t) - (e + d)E(t) && \text{a.e. } t \in [0, 20] \\ \dot{I}(t) &= eE(t) - (g + a + d)I(t) && \text{a.e. } t \in [0, 20] \\ \dot{N}(t) &= (b - d)N(t) - aI(t) && \text{a.e. } t \in [0, 20] \\ \dot{W}(t) &= u(t)S(t) && \text{a.e. } t \in [0, 20]\,, \end{aligned} \tag{9}$$

where $u(t)$ represents the percentage of susceptible individuals being vaccinated per unit of time,

- the input constraints

$$0 \leq u(t) \leq 1\,, \quad \text{a.e. } t \in [0, 20]$$

- the end–point constraints

$$\mathbf{x}(t_0) = (S_0,\, E_0,\, I_0,\, N_0,\, W_0)\,,$$

  and

- the state constraint $h\,(t, \mathbf{x}(t)) = S(t) - 1100 \leq 0\,,\ \forall t \in [0, 20]$ [4].

This problem is nonlinear, thus an adaptive mesh is expected to be more adequate.

Table 1: Parameters and their clinically approved values [12]

Parameters	Definition of Parameters	Clinical values
b	natural birth rate	0.525
d	natural death rate	0.5
c	incidence coefficient	0.001
e	exposed to infectious rate	0.5
g	recovery rate	0.1
a	disease induced death rate	0.2
S_0	initial susceptible population	1000
E_0	initial exposed population	100
I_0	initial infected population	50
R_0	initial recovered population	15
N_0	initial population	1165
W_0	initial vaccinated Population	0

4.3 Numerical Results

Both problems, (P_1) and (P_2), are solved considering four meshes:

a) π_{ML}: obtained by the multi–level time–mesh refinement strategy;

b) π_{R}: obtained by the (single-level) time–mesh refinement strategy;

c) π_{F}: considering equidistant spacing with the smallest time step of π_{ML};

d) π_{S}: considering equidistant spacing with the same number of nodes of π_{ML}.

For problem (P_1), we consider

$$\begin{aligned}
\varepsilon_{\mathbf{x}}^{\max} &= 5 \times 10^{-5} \\
\varepsilon_{\mathbf{q}}^{\max} &= 5 \times 10^{-4} \\
\bar{\varepsilon}_{\mathbf{x}} &= \left[1, 10, 10^2, 10^3, 10^4, 10^5\right] \varepsilon_{\mathbf{x}}^{\max} \\
\bar{\varepsilon}_{\mathbf{q}} &= \left[1, 10, 10^2, 10^3, 10^4\right] \varepsilon_{\mathbf{q}}^{\max}
\end{aligned}$$

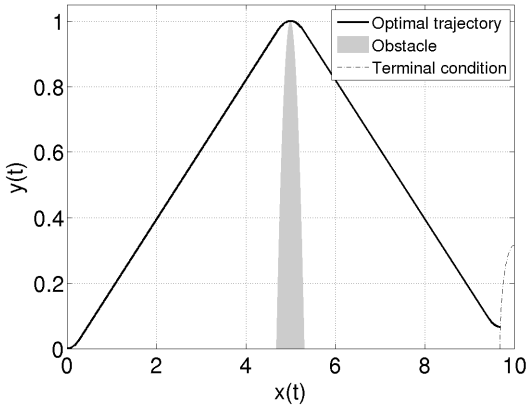


Figure 3: Optimal trajectory

where $\varepsilon_{\mathbf{x}}^{\max}$ is used in the stopping criterion and the vectors $\varepsilon_{\mathbf{x}}^{\max}$ and $\varepsilon_{\mathbf{q}}^{\max}$ are used in the refinement criteria.

As it can be seen in Fig. 3, the car–like system successfully overcomes the obstacle and it stops when the terminal condition (6) is satisfied.

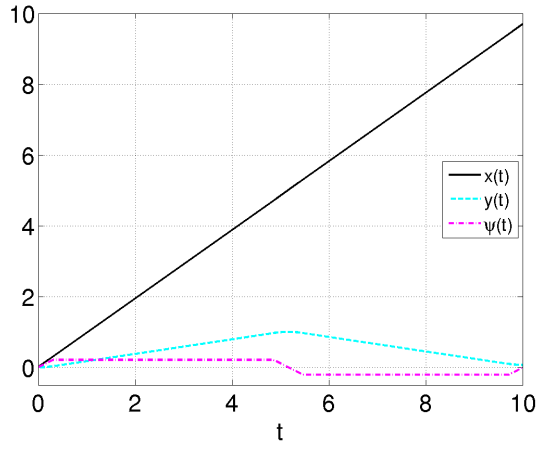
According to Fig. 4b, where the controls associated to (P_1) are shown, the constraint for the curvature $c(\cdot)$ becomes active at the start, in the middle and at the end of the trajectory.

The local errors of the trajectory for all meshes are shown in Fig. 5a using a logarithmic scale. The subintervals that need refinement are at the start, in the middle and at the end of the time interval, since the local errors are greater than the user–specified threshold, coinciding with the subintervals where the curvature is nonzero and the constraint for the curvature becomes active. We also notice that there are different subintervals belonging to different levels of refinement which indicates that the procedure to generate π_{ML} is quite distinct from the one used to generate π_{R}.

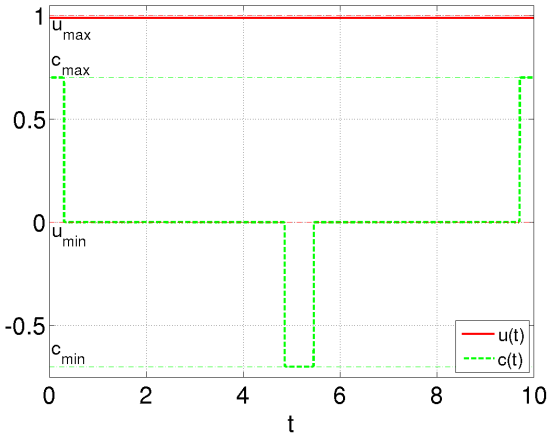
The adjoint multipliers are presented in Fig. 4c and the local error associated to them is shown in Fig. 5b. Comparing Fig. 5a with Fig. 5b, we see that the errors in the trajectory and in the adjoint multiplier have a similar structure along time, further validating the use of the adjoint multiplier in the refinement criteria.

The numerical results concerning the four meshes are shown in Table 2, which shows information about the number of nodes, the smallest time step, the number of iterations needed to solve the NLP problem, the objective functional, the maximum absolute local errors of the trajectory and the $\mathbf{q}$ multipliers, and the CPU times for solving the OCP problem and for computing the local error as well.

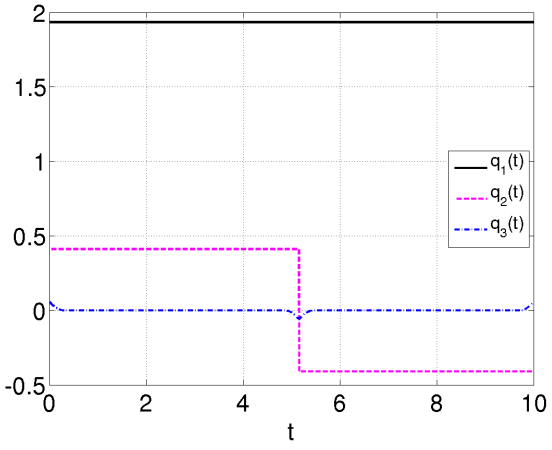
According to Table 2, the mesh π_{ML} has only 32.4% of the nodes of π_{F}, nevertheless both meshes have maximum absolute local errors of the same order of magnitude. Since the procedure to obtain π_{ML} uses a warm start at each refinement iteration, the OCP can be solved three consecutive times and it is still much faster than to solve this problem with the mesh π_{F}. In fact, computing the solution using π_{ML} takes only 4% of the time needed to get a solution using π_{F}, causing significant savings in memory and computational cost. The use of multi–level refinement algorithm in real time optimization problems, such as MPC, has benefits



(a) Optimal solution

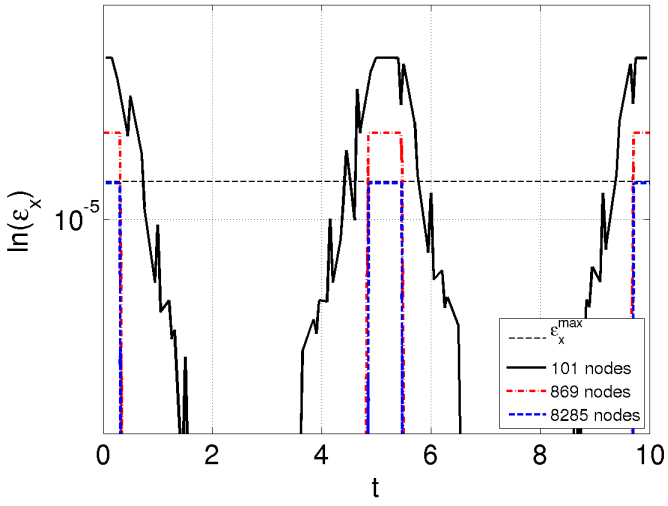


(b) Optimal control

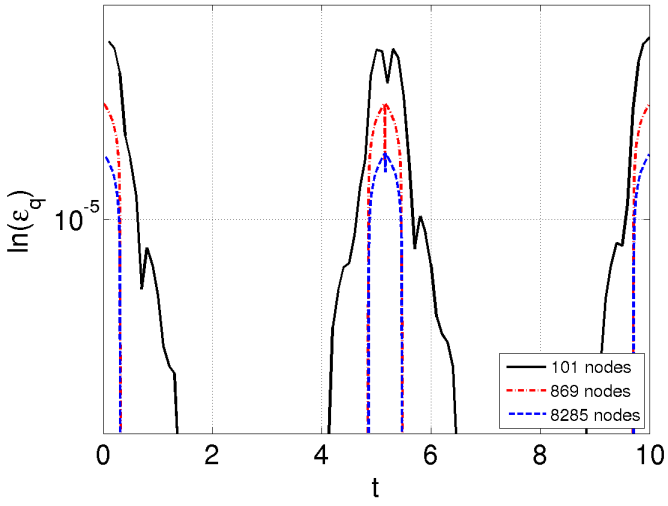


(c) Adjoint multipliers

Figure 4: Numerical results using π_{ML}



(a) Error on the trajectory $\left(\varepsilon_{\mathbf{x}}^{(j)}\right)$



(b) Error on the adjoint multipliers $\left(\varepsilon_{\mathbf{q}}^{(j)}\right)$

Figure 5: Local error for all meshes

Table 2: Comparing results for the Car–like system problem (P_1)

π_j	N_j	Δt_j	I_j	Objective	$\left\|\varepsilon_{\mathbf{x}}^{(j)}\right\|_\infty$	$\left\|\varepsilon_{\mathbf{q}}^{(j)}\right\|_\infty$	CPU time (s) Solver	$\varepsilon_{\mathbf{x}}$	$\varepsilon_{\mathbf{q}}$
π_0	101	$1/100$	25	**9.7**700173	1.029E^{-2}	2.443E^{-2}	1.859	0.638	0.004
π_1	869	$1/25600$	34	**9.7805**247	4.096E^{-4}	1.444E^{-3}	8.895	6.345	0.007
π_2	8285	$1/25600$	50	**9.7805**398	4.714E^{-5}	1.684E^{-4}	186.870	102.973	0.018
π_{ML}	8285	$1/25600$	109	**9.7805**398	4.714E^{-5}	1.684E^{-4}	197.624	109.956	0.029
π_0	101	$1/100$	25	**9.7**700173	1.029E^{-2}	2.443E^{-2}	1.859	0.638	0.004
π_1	182	$1/400$	21	**9.7**798518	2.572E^{-3}	8.180E^{-3}	2.806	1.626	0.004
π_2	577	$1/1600$	31	**9.780**4932	6.407E^{-4}	3.061E^{-3}	7.938	5.494	0.007
π_3	2193	$1/6400$	38	**9.7805**272	1.598E^{-4}	7.837E^{-4}	35.156	23.129	0.013
π_4	8707	$1/25600$	31	**9.7805**392	3.996E^{-5}	1.971E^{-4}	162.731	118.277	0.021
π_{R}	8707	$1/25600$	153	**9.7805**392	3.996E^{-5}	1.971E^{-4}	210.490	149.164	0.049
π_{F}	25601	$1/25600$	406	**9.7805**377	3.996E^{-5}	–	4840.185	773.192	–
π_{S}	8285	$1/8284$	80	**9.7805**577	1.236E^{-4}	–	333.428	113.912	–

since it is possible to obtain a solution very quickly even if the procedure is interrupted in an early stage. According to Table 2, if the procedure is interrupted after 12 seconds, a solution with local error lower than 4.096×10^{-4} is obtained.

The mesh π_{S} has the same number of nodes of π_{ML} but considering equidistant spacing. The analysis of the solution obtained using this mesh allows us to verify the importance of nodes collocation, *i.e.*, having a mesh with non–equidistant spacing nodes produces a solution with higher accuracy than the one obtain using a mesh with equidistant spacing nodes. Moreover, the CPU time spent to compute solution using π_{ML} is $31,3\%$ lower then the one spent to obtain a solution using π_{S}, emphasizing the relevance of using meshes with non–equidistant spacing nodes.

When comparing both refined meshes, we notice that the process to compute the mesh π_{ML} having several refinement levels in a single iteration took only 2 refinement iterations, producing a mesh that has 95% of the nodes of π_{R}. The solution is obtained 6% faster when compared to the CPU time spent to compute the solution using π_{R}. We could expect to be even faster but, as shown in the Table 2, the second refinement iteration takes 50 IPOPT iterations to converge to the optimal solution. This event occurs because the solution of the previous refinement iteration, which is used to create a warm start, has only about 10% of the nodes, having less information about the structure of the solution.

In terms of CPU time, we can see that it is much faster to compute $\varepsilon_{\mathbf{q}}$ than $\varepsilon_{\mathbf{x}}$. In the all procedures, the use of $\varepsilon_{\mathbf{q}}$ in the refinement criterion reduces the computational time, making the refinement algorithm faster.

With respect to IPOPT and according to Table 2, even if the number of nodes is increasing fast at each refinement step, the number of IPOPT iterations are of the same order of magnitude. This is another advantage of the mesh refinement strategy.

Table 3: Results for the vaccination problem (P_2)

π_j	N_j	Δt_j	I_j	Objective	$\left\|\varepsilon_{\mathbf{x}}^{(j)}\right\|_\infty$	$\left\|\varepsilon_{\mathbf{q}}^{(j)}\right\|_\infty$	CPU time (s) Solver	$\varepsilon_{\mathbf{x}}$	$\varepsilon_{\mathbf{q}}$
π_0	101	$1/100$	56	**25**.65774542	1.246E^{-3}	1.246E^{-2}	4.121	0.907	0.007
π_1	1589	$1/6400$	24	**25.5**8223462	1.285E^{-4}	1.285E^{-4}	21.877	14.611	0.012
π_2	5498	$1/6400$	23	**25.5787**9504	4.191E^{-5}	4.491E^{-5}	64.236	76.057	0.018
π_{ML}	5498	$1/6400$	103	**25.5787**9504	4.191E^{-5}	4.491E^{-5}	90.234	91.575	0.037
π_0	101	$1/100$	56	**25**.65774542	1.246E^{-3}	1.246E^{-2}	4.121	0.907	0.007
π_1	401	$1/400$	18	**25.5**9608912	3.115E^{-4}	4.894E^{-3}	4.672	3.423	0.009
π_2	1601	$1/1600$	21	**25.5**8021428	7.786E^{-5}	1.261E^{-3}	16.850	16.439	0.014
π_3	5789	$1/6400$	25	**25.5787**5577	3.998E^{-5}	3.379E^{-5}	85.464	83.978	0.022
π_{R}	5789	$1/6400$	120	**25.5787**5577	3.998E^{-5}	3.379E^{-5}	111.117	204.747	0.052
π_{F}	6401	$1/6400$	70	**25.5787**1473	3.646E^{-5}	–	292.155	99.496	–
π_{S}	5498	$1/5497$	66	**25.578**4340705	2.266E^{-4}	–	254.806	87.912	–

For problem (P_2), we consider

$$\begin{aligned}
\varepsilon_{\mathbf{x}}^{\max} &= 5 \times 10^{-5} \\
\varepsilon_{\mathbf{q}}^{\max} &= 5 \times 10^{-4} \\
\bar{\varepsilon}_{\mathbf{x}} &= \left[1, 5, 10, 50, 10^2\right] \varepsilon_{\mathbf{x}}^{\max} \\
\bar{\varepsilon}_{\mathbf{q}} &= \left[1, 5, 10, 50, 10^2\right] \varepsilon_{\mathbf{q}}^{\max}
\end{aligned}$$

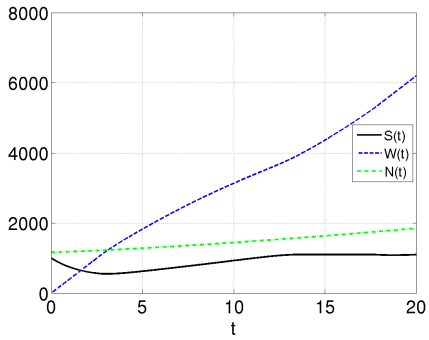
The optimal trajectory is shown in Fig. 6a and 6b and the control is presented in Fig. 6c. The local errors of the trajectory for all meshes are shown in Fig. 6d using the logarithmic scale. Regarding the latter figure, we see that the time domain needs to be entirely refined in the first step and there are different subintervals belonging to different levels of refinement.

The numerical results concerning the four meshes are shown in Table 3, which, as before, shows information about the number of nodes, the smallest time step, the number of iterations needed to solve the NLP problem, the objective functional, the maximum absolute local error of the trajectory and the CPU times for solving the OCP problem and for computing the local error as well.

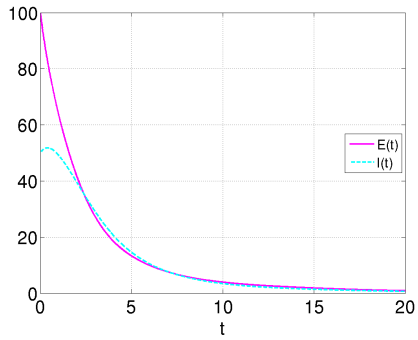
According to Table 3, the mesh π_{ML} has 85% of the nodes of π_{F} and computing the solution using π_{ML} takes only 30% of the CPU time needed to get a solution using π_{F}. Nevertheless, we get solution with the same accuracy.

The solution obtained using the mesh π_{S}, which has the same number of nodes of π_{ML} but with equidistant spacing, has less accuracy than the one computed on π_{ML}. Moreover, computing the solution on π_{S} took 3 times the CPU time spent to get the solution using π_{ML}.

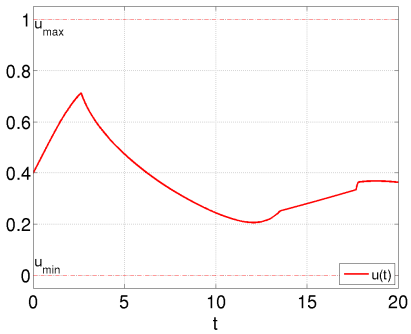
When comparing both refined meshes, we notice that the process to compute the mesh π_{ML} having several refinement levels in a single iteration took only 2 refinement iterations, producing a mesh that has 95% of the nodes of π_{R}. The use



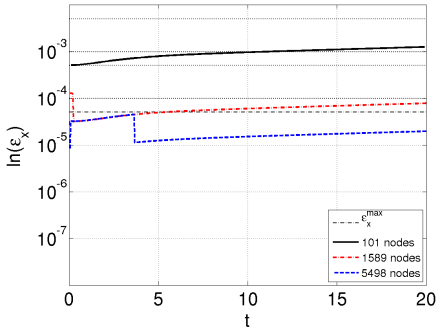
(a) Optimal trajectory: $S(t)$, $W(t)$, $N(t)$



(b) Optimal trajectory: $E(t)$, $I(t)$



(c) Vaccination strategy: $u(t)$



(d) Error on the trajectory: $\left(\varepsilon_{\mathbf{x}}^{(j)}\right)$

Figure 6: Results for problem (P_2)

of the refinement levels brings a significant improvement in the overall computing time, since the solution is obtained 19% faster when compared to the time spent to compute the solution using π_{R}.

With respect to CPU time, once again, we see that it is much faster to compute $\varepsilon_{\mathbf{q}}$ than $\varepsilon_{\mathbf{x}}$. Also in this application, the use of $\varepsilon_{\mathbf{q}}$ as refinement criterion reduces the computational time, making the refinement algorithm quicker.

In terms of IPOPT and according to Table 3, the number of IPOPT iterations are of the same order of magnitude at each refinement step, even if the number of nodes is increasing.

5 Characterization of the Solution using the Necessary Conditions of Optimality

In this section we establish necessary conditions of optimality for the car–like system problem (P_1) considering the pathwise state constraint (7). Then, we verify that the solution obtained numerically satisfies the necessary conditions. Similar analysis to the second problem (P_2) involving the SEIR model is reported in [4].

Considering the problem (P_1), let us recall

$$
\begin{aligned}
&\mathrm{L}\,(t,\mathbf{x},\mathbf{u}) = u^2\,,\\
&\mathrm{G}\,(\mathbf{x}(t_0),\mathbf{x}(t_f)) = 0\,,\\
&\mathbf{f}(t,\mathbf{x},\mathbf{u}) = [u\cos(\psi),\, u\sin(\psi),\, u\,c]\,,\\
&h\,(\mathbf{x}) = (1-y) - 10\,(5-x)^2\,,\\
&\mathbb{U} = [0,1]\times[-0.7,0.7]\,,\\
&\mathbf{x}(t_0) = (0,0,0)\,,\\
&\mathbb{X}_1 = \left\{(x,y,\psi) : (x-x_f)^2 + (y-y_f)^2 + (\psi-\psi_f)^2 \le r^2\right\}\,.
\end{aligned}
$$

The assumptions (H1)–(H6) presented in section 2 are satisfied.

The maximum Principle is satisfied in normal form, *i.e.*, with $\lambda = 1$. This can be checked by finding a set of multipliers satisfying the conditions with $\lambda = 1$, or verifying that certain inward–pointing conditions are satisfied along the trajectory when the state constraint is active (see [8, 7, 11] and references therein). In this case, we can verify that when the trajectory is in a neighbourhood of the state constraint boundary there is a control that drives the car–like system away from the boundary.

Proposition 1 *Consider problem (P_1). A local minimizer $(\mathbf{x}^*,\mathbf{u}^*)$ satisfies*

1. $(\mathbf{p},\mu,\lambda) \neq (0,0,0)$,
2. $\begin{cases} p_1(t) = p_1(t_0) \in \mathbb{R}\,, & t\in[t_0,t_f] \\ p_2(t) = p_2(t_0) \in \mathbb{R}\,, & t\in[t_0,t_f] \end{cases}$,
3. $-\mathbf{q}(t_f) \in N_{\mathbb{X}_1}\left(x_f^*, y_f^*, \psi_f^*\right) = \alpha\left(2\left(x_f^* - x_f\right), 2\left(y_f^* - y_f\right), 2\left(\psi_f^* - \psi_f\right)\right)$ *where* $\alpha > 0$ *and* $(x_f^*, y_f^*, \psi_f^*) = (x^*(t_f), y^*(t_f), \psi^*(t_f))$,
4. $\mathrm{H}\,(t,\mathbf{x}^*,\mathbf{q},\mathbf{u}^*) = \max\limits_{u,c\in\mathbb{U}} q_1 u\cos(\psi^*) + q_2 u\sin(\psi^*) + q_3 uc - u^2\,$,
5. $\mathrm{supp}\{\mu\} \subset I\,(\mathbf{x}^*) = \left\{t\in[t_0,t_f] : (\bar{y} - y(t)) - k\,(\bar{x} - x(t))^2 = 0\right\}$.

Proof 1 *The nontriviality condition (NT) is ensured with $\lambda = 1$.*

Considering $\mathbf{p}(t) = (p_1(t),\, p_2(t),\, p_3(t))$ and $\mathbf{q}(t) = (q_1(t),\, q_2(t),\, q_3(t))$, we define the Hamiltonian as

$$
\mathrm{H}(t,\mathbf{x},\mathbf{q},\mathbf{u}) = q_1 u\cos(\psi) + q_2 u\sin(\psi) + q_3 uc - u^2\,. \tag{10}
$$

From the adjoint system

$$
\begin{aligned}
&-\dot{p}_1(t) = \mathrm{H}_x(t,\mathbf{x}^*,\mathbf{q},\mathbf{u}^*) = 0 && \text{a.e. } t\in[t_0,t_f]\\
&-\dot{p}_2(t) = \mathrm{H}_y(t,\mathbf{x}^*,\mathbf{q},\mathbf{u}^*) = 0 && \text{a.e. } t\in[t_0,t_f]\\
&-\dot{p}_3(t) = \mathrm{H}_\psi(t,\mathbf{x}^*,\mathbf{q},\mathbf{u}^*) = -q_1 u^*\sin(\psi^*) + q_2 u^*\cos(\psi^*) && \text{a.e. } t\in[t_0,t_f]\,.
\end{aligned} \tag{11}
$$

Since $\mathbf{p}(\cdot)$ *is an absolutely continuous function, we get*

$$p_1(t) = p_1(t_0) \in \mathbb{R}\,, \quad t \in [t_0, t_f]$$
$$p_2(t) = p_2(t_0) \in \mathbb{R}\,, \quad t \in [t_0, t_f]$$

concluding that $q_1(\cdot)$ *and* $q_2(\cdot)$ *may change their value just when* $t \in \{t \in [t_0, t_f] : h(\mathbf{x}^*(t)) = 0\}$. *In fact, according to Fig. 4c,* $q_1(t)$ *is a constant function and* $q_2(t)$ *changes its value just at one time instant when the trajectory hits the state constraint.*

Considering $\mathrm{G}(\cdot) = 0$, *the end–point constraints* (5) *and* (6), *the transversality condition* (T) *reads*

$$-\mathbf{q}(t_f) \in N_{\mathbb{X}_1}\left(x_f^*, y_f^*, \psi_f^*\right) = \alpha\left(2\left(x_f^* - x_f\right), 2\left(y_f^* - y_f\right), 2\left(\psi_f^* - \psi_f\right)\right)$$

where $\alpha > 0$ *and* $(x_f^*, y_f^*, \psi_f^*) = (x^*(t_f), y^*(t_f), \psi^*(t_f))$, *and according to the Weierstrass condition* (WC)

$$\mathrm{H}\left(t, \mathbf{x}^*, \mathbf{q}, \mathbf{u}^*\right) = \max_{u,c \in \mathbb{U}} q_1 u \cos(\psi^*) + q_2 u \sin(\psi^*) + q_3 uc - u^2\,. \tag{12}$$

Recalling the pathwise state constraint (7), *from* (CS) *we obtain*

$$\operatorname{supp}\{\mu\} \subset I\left(\mathbf{x}^*\right) = \left\{t \in [t_0, t_f] : (\bar{y} - y(t)) - k\left(\bar{x} - x(t)\right)^2 = 0\right\} \tag{13}$$

for some $(\bar{x}, \bar{y}) \in \mathbb{R}^2$ *and* $k \in \mathbb{R}$. *Moreover,*

$$\nabla h\left(\mathbf{x}^*(t)\right) = (2k(\bar{x} - x^*(t)), -1, 0) \tag{14}$$

and

$$q_1(t) = p_1(t) + 2k \int_{[t_0,t)} (\bar{x} - x^*(t)) \mathrm{d}\mu(s) \tag{15}$$

$$q_2(t) = p_2(t) - \int_{[t_0,t)} \mathrm{d}\mu(s) \tag{16}$$

$$q_3(t) = p_3(t)\,, \tag{17}$$

implying $q_3(\cdot)$ *is an absolutely continuous function.*

Applying the Maximum Principle we obtain a set of conditions characterizing the optimal solution which are in agreement with the numerical results (cf. Fig. 4c)

Proposition 2 *Let* $(\mathbf{x}^*, \mathbf{u}^*)$ *be a local minimizer to the problem* (P_1). *Applying the Maximum Principle 2.1, we conclude that*

1. *if* $u^*, c^* \in \operatorname{int} \mathbb{U}$ *then*

$$\begin{cases} u^* = \dfrac{q_1}{2}\cos(\psi^*) + \dfrac{q_2}{2}\sin(\psi^*) \\ q_3 = 0 \end{cases}, \tag{18}$$

2. *if* $c^* = c_{\min}$, *then* $q_3 \leq 0$, *and*

3. *if* $c^* = c_{\max}$, *then* $q_3 \geq 0$.

Proof 2 *Let us first consider the case when* $\mathbf{u} \in \operatorname{int} \mathbb{U}$. *In this case, the Hamiltonian, stated in proposition 1, is maximized when*

$$\mathrm{H}_{\mathbf{u}}(t, \mathbf{x}^*, \mathbf{q}, \mathbf{u}^*) = 0\,.$$

When $\mathbf{u}^* \in \operatorname{int} \mathbb{U}$ *we get*

$$\begin{cases} \mathrm{H}_u(t, \mathbf{x}^*, \mathbf{q}, \mathbf{u}^*) = q_1 \cos(\psi^*) + q_2 \sin(\psi^*) + q_3 c^* - 2u^* = 0 \\ \mathrm{H}_c(t, \mathbf{x}^*, \mathbf{q}, \mathbf{u}^*) = q_3 u^* = 0 \end{cases}$$

implying

$$\begin{cases} u^* = \dfrac{q_1}{2} \cos(\psi^*) + \dfrac{q_2}{2} \sin(\psi^*) \\ q_3 = 0 \end{cases}.$$

When $c^* = c_{\min}$, *the Weierstrass condition* (WC) *reads*

$$q_3 u^* \left(c_{\min} - c\right) \geq 0\,.$$

Since $u^* \geq 0$ *and* $c \geq c_{\min}$, *we conclude* $q_3 \leq 0$ *when* $c^* = c_{\min}$. *Furthermore, it can be shown that* $q_3 \geq 0$ *when* $c^* = c_{\max}$.

Remark 1 *We use the numerical results for* ψ^*, c^* *and* $\mathbf{q}$ *to evaluate* u^* *and we compare it against the* $\hat{u}^*$ *given by the numerical procedure. According to Fig. 7a, they coincide.*

In Fig. 7b, we provide the graphics of c^* *and* q_3 *obtained numerically and we can verify relationships of the proposition 2.*

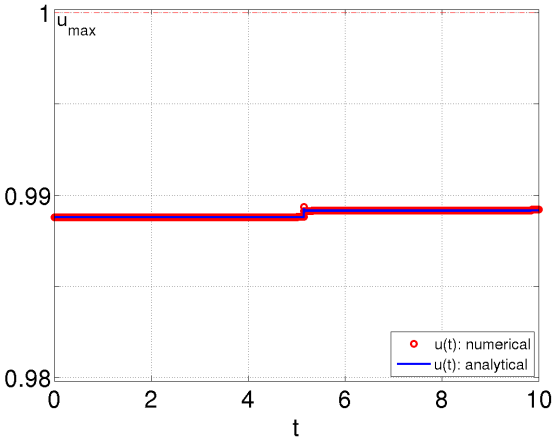
6 Conclusions

We develop an adaptive mesh refinement algorithm providing local mesh resolution refinement only where it is required. In the end, the OCP is solved using an adapted mesh which has less nodes in the overall procedure, yet with higher concentration of nodes in time subintervals where the trajectory shows higher nonlinear behaviour. This procedure showed significant savings in memory and computational cost when compared to equidistant meshes.

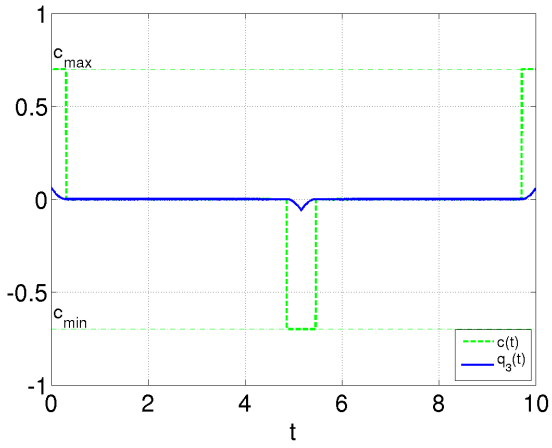
When using this strategy, where the mesh is progressively refined to catch special features of the problem, there is no need to define *a priori* the most appropriately mesh, which is another advantage of this procedure. According to the proposed algorithm, we do not need to remove nodes, nevertheless this algorithm can be extended to a version that includes to coarsen the mesh. This feature would be of relevance in the context of model predictive control (MPC) in which sequences of similar optimal control problems are solved.

Due to the fast response of the algorithm, it can be used to solve real–time optimization problems, in particular, in MPC. The use of adaptive mesh refinement algorithm in real time optimization problems, such as MPC, has additional benefits since it is possible to quickly obtain a solution even if the procedure is interrupted at an early stage.

The applications presented in this paper demonstrate the advantage of the proposed adaptive mesh strategy, which leads to results with greater accuracy and with lower overall computational time when compared to other commonly used approaches.



(a) $u^*(t)$: analytical *vs* numerical



(b) Relationship between $c^*(t)$ and $q_3(t)$

Figure 7: Solution characterization

Acknowledgements

We would like to thank Prof. Maria do Rosário de Pinho for her interest in this topic and for helpful suggestions. The support of FCT – Fundação para a Ciência e Tecnologia – under Grants PTDC/EEA-CRO/116014/2009 and PTDC/EEI-AUT/1450/2012 and of the European Union Seventh Framework Programme [FP7-PEOPLE-2010-ITN] under grant agreement n. 64735-SADCO are greatly acknowledged.

References

[1] J. T. Betts, *Practical methods for optimal control using nonlinear programming*, SIAM, 2001.

[2] J. T. Betts, N. Biehn, S. L. Campbell and W. P. Huffman, Compensating for order variation in mesh refinement for direct transcription methods, *Journal of Computational and Applied Mathematics*, **125** (2000), 147–158.

[3] J. T. Betts and W. P. Huffman, Mesh refinement in direct transcription methods for optimal control, *Optimal Control Applications and Methods*, **19** (1998), 1–21.

[4] M. H. A. Biswas, L. T. Paiva and M. d. R. de Pinho, A SEIR model for control of infectious diseases with constraints, *Mathematical Biosciences and Engineering*, **11** (2014), 761–784.

[5] P. Falugi, E. Kerrigan and E. Van Wyk, *Imperial College London Optimal Control Software. User Guide (ICLOCS)*, Department of Electrical Engineering, Imperial College London, London, UK, 2010.

[6] F. A. C. C. Fontes, A general framework to design stabilizing nonlinear model predictive controllers, *Systems and Control Letters*, **42** (2001), 127–143.

[7] F. A. C. C. Fontes and H. Frankowska, Normality and nondegeneracy for optimal control problems with state constraints, *Journal of Optimization Theory and Applications*, 22.

[8] F. A. C. C. Fontes and S. O. Lopes, Normal forms of necessary conditions for dynamic optimization problems with pathwise inequality constraints, *Journal of Mathematical Analysis and Applications*, **399** (2013), 27–37.

[9] I. Kolmanovsky and N. McClamroch, Developments in nonholonomic control problems, *IEEE Control Systems*, **15** (1995), 20–36.

[10] I. Kornienko, L. T. Paiva and M. d. R. d. Pinho, Introducing state constraints in optimal control for health problems, *Procedia Technology*, **17** (2014), 415–422.

[11] S. O. Lopes, F. A. Fontes and M. d. R. de Pinho, On constraint qualifications for nondegenerate necessary conditions of optimality applied to optimal control problems, *Discrete and Continuous Dynamical Systems (DCDS-A)*, **29** (2011), 559–575.

[12] R. M. Neilan and S. Lenhart, An introduction to optimal control with an application in disease modeling, *Modeling paradigms and analysis of disease transmission models*, **75** (2010), 67–81.

[13] L. T. Paiva, *Optimal Control in Constrained and Hybrid Nonlinear System: Solvers and Interfaces*, Technical report, Faculdade de Engenharia, Universidade do Porto, 2013.

[14] L. T. Paiva and F. A. C. C. Fontes, Mesh refinement strategy for optimal control problems, *AIP Conference Proceedings*, **1558** (2013), 590–593, Proceding of the ICNAAM 2013 - 11th International Conference on Numerical Analysis and Applied Mathematics.

[15] L. T. Paiva and F. A. C. C. Fontes, Time–mesh refinement in optimal control problems for nonholonomic vehicles, *Procedia Technology*, **17** (2014), 178–185.

[16] M. A. Patterson, W. W. Hager and A. V. Rao, A ph mesh refinement method for optimal control, *Optimal Control Applications and Methods*, Doi:10.1002/oca.2114.

[17] R. B. Vinter, *Optimal Control*, Springer, 2000.

[18] A. Wächter and L. T. Biegler, On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming, *Mathematical Programming*, **106** (2006), 25–57.

[19] Y. Zhao and P. Tsiotras, Density functions for mesh refinement in numerical optimal control, *Journal of Guidance, Control, and Dynamics*, **34** (2011), 271–277.