

An Interior-Point Approach to Trajectory Optimization

Nicolas Bérend, J. Frederic Bonnans, Julien Laurent-Varin, Mounir Haddou, Christophe Talbot

▶ **To cite this version:**

Nicolas Bérend, J. Frederic Bonnans, Julien Laurent-Varin, Mounir Haddou, Christophe Talbot. An Interior-Point Approach to Trajectory Optimization. [Research Report] RR-5613, INRIA. 2005, pp.32. inria-00070395

HAL Id: inria-00070395
https://inria.hal.science/inria-00070395

Submitted on 19 May 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

An Interior-Point Approach to Trajectory Optimization

Nicolas BÉREND — J. Frédéric BONNANS — Julien LAURENT-VARIN —

Mounir HADDOU — Christophe TALBOT

N° 5613

Juin 2005

Thème NUM



ISSN 0249-6399 ISRN INRIA/RR--5613--FR+ENG



An Interior-Point Approach to Trajectory Optimization

Nicolas BÉREND* , J. Frédéric BONNANS† , Julien LAURENT-VARIN‡ ,
Mounir HADDOU§ , Christophe TALBOT¶

Thème NUM — Systèmes numériques
Projet SYDOCO

Rapport de recherche n° 5613 — Juin 2005 — 32 pages

Abstract: This paper presents an interior-point approach for solving optimal control problems. Combining a flexible refinement scheme with dedicated linear algebra solvers, we obtain an efficient algorithm. Numerical results are displayed for various problems, among them several variants of atmospheric reentry.

Key-words: Optimal control, trajectory optimization, atmospheric reentry, launcher, differential equations, Runge-Kutta and symplectic schemes, sparse algebra, band matrices, QR factorization.

This study is supported by CNES and ONERA, in the framework of the CNES fellowship of the second author.

* Nicolas BÉREND: Long-Term Design & System Integration Department, ONERA, BP 72, 29 av. division Leclerc, 92322 Châtillon, France (Nicolas.Berend@onera.fr).

† J. Frédéric BONNANS: Projet Sydoco, Inria-Rocquencourt, Domaine de Voluceau, BP 105, 78153 Le Chesnay, France (Frederic.Bonnans@inria.fr)

‡ Julien LAURENT-VARIN: Projet Sydoco, INRIA Rocquencourt, Domaine de Voluceau, BP 105, 78153 Le Chesnay (Julien.Laurent-Varin@inria.fr)

§ Mounir HADDOU: UMR 6628, MAPMO, BP 6759, 45067 Orléans Cedex 2, France (haddou@labomath.univ-orleans.fr).

¶ Christophe TALBOT: CNES, Launcher Directorate, Rond-Point de l'Espace, 91023 Evry Cedex, France (Christophe.Talbot@cnes.fr).

Unité de recherche INRIA Rocquencourt
Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay Cedex (France)
Téléphone : +33 1 39 63 55 11 —Télécopie : +33 1 39 63 53 30

Une approche par point intérieur pour l'optimisation de trajectoire

Résumé : Cet article présente une approche par point intérieur, pour la résolution des problèmes de commande optimale. Nous obtenons un algorithme efficace en combinant un schéma de raffinement flexible avec une algèbre linéaire adaptée. Des résultats numériques sont obtenus pour différents problèmes, dont quelque variation autour du problème de rentrée atmosphérique.

Mots-clés : Commande optimale, optimisation de trajectoire, rentré atmosphérique, lanceur, equations différentielles, schémas de Runge-Kutta, schémas symplectiques, algèbre linéaire creuse, matrices bandes, factorisation QR.

1 Introduction

This paper discusses the numerical resolution of optimal control problems of systems governed by ordinary differential equations. We deal with trajectory optimization methods. In other words, the object manipulated by the algorithm is a certain function of time (as opposed to Hamilton-Jacobi-Bellman approaches which solve a certain partial differential equation). Trajectory optimization algorithms start from an initial, in general nonfeasible trajectory, and find a trajectory that approximately satisfies some necessary conditions for optimality. In this paper we assume all data to be at least C^2 (twice continuously differentiable), which allows to use second-order Taylor expansions. Our format includes running mixed control and state constraints (which means that some constraints involving both the control and state, or only one of them, are active for all time).

Although there is no full agreement on this terminology, one generally distinguishes direct and indirect methods. Direct methods approximately solve a time discretized version of the optimal control problem by some nonlinear programming algorithm, and then possibly refine the discretization mesh and loop. In indirect methods, the control variable is eliminated thanks to Pontryaguin's principle, and the resulting two point boundary value problem (TPBVP), with only state and costate variables (plus the Lagrange multipliers associated with constraints) is solved without discretizing functions of time; the time integration is performed thanks to a (somewhat hidden) ODE solver. The simplest indirect method is the shooting algorithm, which with the pair (state, costate) at initial time associates its value at the final time. In this way necessary optimality conditions appearing in Pontryaguin's principle are reduced to the final dimensional problem of matching the end point conditions. Since the integration of the reduced (state, costate) differential equation over a long time may be unstable, one often prefers to take as variables the values of state and costate at sampled points, so that integration occurs over small intervals of time. See Stoer and Bulirsch [30].

The common feeling is that indirect methods are more efficient when the intervals of time during which a set of active constraints is active is easy to predict, up to a parametrization of junction times (the endpoints of these intervals). Also, these junction times have to satisfy some regularity conditions, see Maurer [25]. In that case, by using high order integration schemes, one obtains very precise and cheap solvers.

Yet checking if various regularity conditions needed by this approach hold, and writing the optimality conditions correctly, may be a delicate task. It may also happen that little is known on the structure of the solution of the problem. In that case it may be preferable to use direct methods, that are as easy to use as any nonlinear programming solver: in other words, finding a correct initial point requires often some intuition from the user, while some scaling and tuning of algorithmic parameters may help. We refer to Betts [4, 11] for an overview of numerical methods for optimal control problems and comparison of various approaches.

Direct methods generally solve the optimality system of the discretized problem by using Newton or quasi-Newton algorithms. An early reference on Newton's method for (unconstrained) optimal control problems is Mitter [26]. Following the discover of sequential quadratic programming (SQP) algorithms by Han [21] and Powell [29], a first generation

of direct methods in which only control variables appear in the optimization solver were proposed Kraft [22]. Implementations of that time used full matrices and full quasi-Newton approximations of Hessian of Lagrangian (based, for instance, on the BFGS update). This induced a severe limitation of dimension. Next appeared sparse implementations of SQP algorithms and their application to optimal control problems, using sparse finite differences for derivatives (Betts and Huffman [6], or exact values (Bonnans and Launay) [8]. In these codes the optimization variables are both control and states, and the state equation is an equality constraint of the optimization solver. In both cases the quadratic programming (QP) algorithm consists in an active set strategy combined with a reduction of variables similar to reduced gradient methods (see e.g. Gill et al. [16, Section 6.3]). Therefore if many constraints are active, the basis matrix may need a large memory. Also, as for all active set strategies for large scale problems, the number of inner iterations (of the QP) may become extremely large.

These drawbacks pushed some authors to propose interior-point algorithms for optimal control problems. Wright [33] analyses some path-following algorithm for solving convex quadratic problems, and applies them to the resolution of QPs arising in SQP algorithms for general optimal control problems. Vicente [32] discusses interior-point methods for nonlinear programs but (despite the title) optimal control problems are only a motivation for the paper and and not really discussed. Leibfritz and Sachs [24] analyse SQP algorithms with inexact resolution of the QP, and apply this analysis to the resolution of the QP by interior-point algorithms. Bergounioux et al. [3] give numerical comparisons of two interior-point solvers to an active-set strategy algorithm.

All the above papers put the emphasis on the optimization solver, and do not take into account the influence of discretization errors. An exception is Betts [5] where a refinement strategy is proposed. This strategy is heuristic and takes into account only discretization errors for the state equation.

There is also a literature specifically devoted to the analysis of discretization errors, mainly the analysis of distance between the solution of the continuous problem and the one of the discretized problem; we review some of its results at the end of section 2.

Our contribution is to present an interior-point algorithm in which the accuracy of discretization is controlled at each step, thanks to a flexible built-in refinement scheme. The latter takes into account dual as well as primal errors. In addition, we use a dedicated linear algebra solver (based on QR factorization of sparse matrices) in order to achieve a fast implementation. We apply the method to various problems, in particular to the atmospheric reentry problem.

The paper is organized as follows. We review the theory of error estimates for unconstrained optimal control problems, and some extensions to constrained problems, in Section 2. Then we introduce an "optimal refinement" procedure in Section 3. Fast dedicated numerical algebra is presented in Section 4. The interior-point approach for constrained problems is discussed in Section 5. Then we turn to numerical results. Section 6 is devoted to Goddard's problem. Section 7 presents an application to Fuller's problem. Finally reentry problems are dealt with in section 8.

2 Unconstrained problems: error estimates

In this section we briefly review the recent analysis of error estimates for unconstrained optimal control problems by two of the authors in [9]. It combines the formulation of the optimality system of the discretized problem due to Hager [17], with the theory or error orders for one step methods. Adding an additional state variable if necessary, we may assume that the cost is a function of the final state:

$$\text{Min } \Phi(y(T)); \quad \dot{y}(t) = f(y(t), u(t)), \quad t \in [0, T]; \quad y(0) = y^0. \tag{1}$$

Here f and Φ are C^∞ functions $\mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^n$ and $\mathbb{R}^n \to \mathbb{R}$, respectively. Use now a Runge-Kutta type discretization, where the control variable is discretized similarly to the state variable, i.e., that with each "inner state" y_{ki} is associated an inner control u_{ki}:

$$\left\{ \begin{array}{lll} \text{Min} & \Phi(y_N); & \\ y_{k+1} & = y_k + h_k \sum_{i=1}^s b_i f(y_{ki}, u_{ki}), & k = 0, \ldots, N-1, \\ y_{ki} & = y_k + h_k \sum_{j=1}^s a_{ij} f(y_{kj}, u_{kj}), & i = 1, \ldots, s, \\ y_0 & = y^0. & \end{array} \right. \tag{2}$$

The positive time steps h_k are such that $\sum_{k=0}^{N-1} h_k = T$. The Runge-Kutta scheme is parameterized by its coefficients (a, b), an $s \times s$ matrix and a vector of $\mathbb{R}^s$ respectively. Assuming all coefficients b_i to be nonzero, it was shown by Hager [17] that (after some change of variables in the Lagrange multipliers) the optimality system can be written in the following form:

$$\left\{ \begin{array}{rcll} y_{k+1} & = & y_k + h_k \sum_{i=1}^s b_i f(y_{ki}, (u_{ki}), & k = 0, \ldots, N-1, \\ y_{ki} & = & y_k + h_k \sum_{j=1}^s a_{ij} f(y_{kj}, u_{kj}), & i = 1, \ldots, s, \\ p_{k+1} & = & p_k - h_k \sum_{i=1}^s \hat{b}_i f_y(y_{ki}, u_{ki})^\top p_{ki}, & k = 0, \ldots, N-1, \\ p_{ki} & = & p_k - h_k \sum_{j=1}^s \hat{a}_{ij} f_y(y_{kj}, u_{kj})^\top p_{kj}, & i = 1, \ldots, s, \\ 0 & = & f_u(y_k, u_k)^\top p_k, & k = 0, \ldots, N-1, \\ 0 & = & f_u(y_{ki}, u_{ki})^\top p_{ki}, & i = 1, \ldots, s, \\ y_0 & = & y^0, \qquad p_N \; = \; \Phi'(y_N), & \end{array} \right. \tag{3}$$

where

$$\hat{b} := \quad \text{and} \quad \hat{a}_{ij} := (b_i b_j - b_j a_{ij})/b_i, \quad \text{for all } i \text{ and } j. \tag{4}$$

The above system may be interpreted as a *partitioned Kunge-Kutta discretization scheme* (i.e., a Runge-Kutta discretization with coefficients that depend on the index of the differential variable) for the optimality conditions, stated below, of problem (1):

$$\left\{ \begin{array}{rcll} \dot{y}(t) & = & f(y(t), u(t)), & t \in [0, T], \\ \dot{p}(t) & = & -f_y(y(t), u(t))^\top p(t), & t \in [0, T], \\ p(T) & = & \Phi'(y(T)), \quad y(0) = y^0, & \\ 0 & = & f_u(y(t), u(t))^\top p(t), & t \in [0, T]. \end{array} \right. \tag{5}$$

If the Hamiltonian function $H(y,u,p) := p \cdot f(u,y)$ is, in the neighborhood of the optimal trajectory, a strongly convex function of u, then we can eliminate the control from the above algebraic constraint (thanks to the implicit function theorem) so that (5) reduces to a two points boundary value problem. In that case, if the solution of the discretized problem is, as we may hope, close to the one of the continuous problem, we can as well eliminate controls from the algebraic equations in (3). We obtain so-called continuous and discrete *reduced* (i.e., without control variables) formulations, and the discrete reduced formulation appears as a partitioned Runge-Kutta discretization of (5).

The theory of error orders for partitioned Runge-Kutta discretization schemes is now well-understood, see [18, 19, 20]. When (4) holds it can be proved that the scheme is symplectic (see e.g. [18] for a detailed study of symplectic schemes). Conditions for error order up to 4 were obtained "by hands" in [17]. One may find in [9] a simplification of order conditions derived from [18, 19, 20] for symplectic schemes, that allows to state them for order up to 6.

The above mentioned results apply only for unconstrained problems with strongly convex Hamiltonians. Dontchev et al. [13] show that, for problems with control constraints, high order Runge-Kutta schemes typically will give (because of the constraint) an error of order two. For mixed control and state constraints, Malanowski et al. obtain an error estimate of the order of the stepsize. A similar result is obtained by Dontchev and Hager [12] for first-order state constraints. Extending these results to other classes of optimal control problem (as for Goddard's problem of section 6) is a significant challenge.

3 Mesh Refinement

Consider an ordinary differential equation, whose end point conditions will be discussed later:

$$\dot{z}(t) = F(z(t)), \quad t \in [0,T], \tag{6}$$

where F is a C^∞ mapping $\mathbb{R}^q \to \mathbb{R}^q$. The corresponding one step discretization is

$$z_{k+1} = z_k + h_k \Phi(z_k, h_k), \tag{7}$$

where $\Phi : \mathbb{R}^q \times \mathbb{R} \to \mathbb{R}^q$. Again $\{h_k\}$ are positive time steps, such that $\sum_{k=0}^{N-1} h_k = T$. For a Cauchy problem, i.e., if the initial condition $z(0)$ is given, the value of time steps is obtained by induction: given all h_i for $i < k$, as well as z_k, define the local error at step k as $e_k := \hat{z}_k(h_k) - z_{k+1}$, where $\hat{z}_k(t)$ satisfies $\hat{z}_k(0) = z_k$ and $\dot{\hat{z}}_k(t) = F(\hat{z}_k(t))$. The theory of error estimates tells us that the local error on step k is (at least for the Runge-Kutta schemes that we have in mind) of the form $e_k = C_k h_k^{p+1} + o(h_k^{p+1})$, where the order p is, in principle, known (or can be identified numerically). The error estimate, for a first trial of the value of h_k, may therefore be estimated by comparing the local variation of the differential variable with the one computed by a scheme of higher order. This gives, if the time step is small enough, an estimate $\hat{e}_k$ of the "true" local error e_k, satisfying

$$\hat{e}_k = C_k h_k^{p+1} + o(h_k^{p+1}); \tag{8}$$

the latter can be used to set h_k to a value that ensures that the local error is below a given threshold.

For a two points boundary value problem, the situation is different. Typically the discretized problem is solved first using a rough estimate of the size of time steps, and then ones tries to refine the mesh, i.e., to add other discretization points. Consider the *optimal refinement problem*, which is defined as problem of having the sum of local errors below a given threshold E. Since $e_k = C_k h_k^{p+1} + o(h_k^{p+1})$, dividing the step h_k into q_k smaller steps of size h_k/q_k, where q_k is a positive integer, reduces the principal term of the local error to $C_k(h_k/q_k)^{p+1}$ on each smaller step, i.e. a total of e_k/q_k^p on the q_k steps. Neglecting the contribution of $o(h_k^{p+1})$ to the local error, we see that the optimal refinement problem can be formulated as follows:

$$\operatorname*{Min}_{q\in\mathbb{N}_*^N} \sum_{k=1}^N q_k; \quad \sum_{k=1}^N \frac{\hat{e}_k}{q_k^p} \le E. \tag{9}$$

Here N_* denotes the set of positive integers, and $\hat{e}_k$ is again the estimate of local error obtained with a higher order scheme. Let us prove that this nonlinear integer programming problem can be solved at low price by the algorithm below:

Algorithm 1. (Optimal refinement)

For $k = 1, \ldots, N$ do $q_k := 1$. **End for**
While $\sum_{k=1}^N e_k/q_k^p > E$ do
  Compute $k_g \in \operatorname{argmax}_k \{e_k\,(1/q_k^p - 1/(q_k+1)^p)\}$
  $q_{k_g} := q_{k_g} + 1$.
End While

Call local gain the amount $g_k := \hat{e}_k\,(1/q_k^p - 1/(q_k+1)^p)$. This is the amount by which the estimate of sum of local errors is decreased by increasing q_k by one. The algorithm consists simply in adding points where the local gain is maximal.

Lemma 2. Algorithm 1 stops after a finite number of steps, and its output is the solution of problem (9).

Proof. We leave to the reader the proof of finiteness of algorithm 1, and prove its optimality. Consider the related problem of minimizing the sum of estimates of local errors, subject to a given number of added points:

$$(D_m) \qquad \operatorname*{Min}_{q\in\mathbb{N}_*^N} \sum_{k=1}^N \frac{\hat{e}_k}{q_k^p}; \quad \sum_{k=1}^N q_k = N + m.$$

The value function denoted of this problem, denote by E_m, is strictly decreasing (we may assume all $\hat{e}_k$ to be positive) and has limit 0 when $m \uparrow +\infty$. If $E_0 < E$ then there is no need to add points and the conclusion holds. Otherwise, there exists a unique $r \in I\!N_*$ such that

$$E_r \le E < E_{r-1}. \tag{10}$$

Any solution of (D_r) is solution of (9), since by the definition of r it is not possible to satisfy the constraint of (9) by adding less than E_r points. We end the proof by showing that the output of algoritm 1, denoted by $\{\bar{q}_k\}$, is solution of (D_r).

The proof is by induction over the values of $r \in I\!N$ such that (10) holds. If $r = 0$ the conclusion holds (no point is added). Assume that it holds for all integer until $r-1$, and take a value of E such that (9) has value $N+r$. Let $\bar{q}$ be the output of algorithm 1, and let k_0 be the index for which a time step has been added at the last step of algorithm 1. Set $\hat{q}_k = \bar{q}_k$ for all $k \neq k_0$, and $\hat{q}_{k_0} = \bar{q}_{k_0} - 1$.

Let $\{q_k\}$ satisfy the constraint of (D_r). We have to prove that the amount $\Delta := \sum_{k=1}^N \hat{e}_k(1/\bar{q}_k^p - 1/q_k^p)$ is nonpositive. There exists i be such that $q_i > 1$. Set $\tilde{q}_k = q_k$ for all $k \neq i$, and $\tilde{q}_i = q_i - 1$. We may write $\Delta = \Delta_1 + \Delta_2$, where

$$\Delta_1 := \sum_{k=1}^N \hat{e}_k \left(\frac{1}{\hat{q}_k^p} - \frac{1}{\tilde{q}_k^p} \right); \quad \Delta_2 := \hat{e}_{k_0} \left(\frac{1}{\bar{q}_{k_0}^p} - \frac{1}{(\bar{q}_{k_0} - 1)^p} \right) - \hat{e}_i \left(\frac{1}{q_i^p} - \frac{1}{(q_i - 1)^p} \right). \quad (11)$$

Since by our induction $\{\hat{q}_k\}$ is solution of (D_{r-1}), and $\tilde{q}$ is feasible for (D_{r-1}), we have that $\Delta_1 \leq 0$. Let us prove that $\Delta_2 \leq 0$ for a certain i. If $\bar{q}_{k_0} \leq q_{k_0}$, then we may take $i = k_0$. Otherwise, in view of the constraint of (D_r), we may take $i \neq k_0$ such that $\bar{q}_i < q_i$. Since k_0 is the index of maximal local gain we have that

$$\Delta_2 \leq \hat{e}_i \left(\frac{1}{\bar{q}_i^p} - \frac{1}{(\bar{q}_i - 1)^p} \right) - \hat{e}_i \left(\frac{1}{q_i^p} - \frac{1}{(q_i - 1)^p} \right) < 0. \quad (12)$$

The conclusion follows. □

Remark 3. A fast implementation of algorithm 1 is obtained by sorting the time steps along the values of the local gain, and that needs at most $O(N \log N)$ operations. Each step of the algorithm needs to update this ordering, which needs at most $O(\log N)$ operations. We obtain that the algorithm needs at most $O((N+m)\log N)$ operations. This is negligible with respect to the number of operations needed for computing Newton steps, as we will see in the next section.

4 Linear algebra

Solving the discrete optimality system (3) by a Newton step needs to solve linear systems whose Jacobian matrix is, when variables are ordered by increasing values of time, a band matrix. We recall some classical results on band matrices, refering e.g. to [2] for details. Given a square matrix A of size n_A, denote by q the band size, i.e., the smallest integer such that $A_{ij} = 0$ if $|i-j| > q$ (for instance, a diagonal matrix has band size 0).

Lemma 4. The computation of the QR factorization, based on Givens rotations, of a matrix A of band size q is such that the (upper triangular) factor R has band size $2q$. The number of operations for factorization is of order $q^2 n_A$, and the number of operations for solving the linear system (after factorization) is of order $q n_A$.

We apply this result to the Jacobian of (3). We may assume that $m = O(n)$. Then the band size satisfies $q = O(sn)$, where s is the number of inner steps in the Runge-Kutta scheme, and $n_A = O(snN)$. We obtain the following.

Lemma 5. The computation of the QR factorization, based on Givens rotations, of the Jacobian of (3) needs $O(s^3n^3N)$ operations for factorization, and $O(s^2n^2N)$ operations for solving the linear system (after factorization).

Remark 6. (i) The cost of factorization of the Jacobian is of the same order that the integration of the state equation (the control being given) by a fully implicit scheme, and the ratio is of order s^2 for a semi implicit scheme (with $O(s)$ implicit steps). This means that the computation of the Newton step is not much more expensive that the simulation of the system by an implicit scheme.
(ii) The ratio of number of operations between factorization and solve is of order sn. This means that, generally, factorization will be by far the most expensive part of resolution.
(iv) For constrained problems we must add rows (constraints) and variables (Lagrange multipliers). However, provided the number of constraints is of order n, the conclusion of lemma 5 also holds in that case.

5 Interior-point algorithms for constrained problems

5.1 Interior point

Consider now a constrained optimal control problem of the following form:

$$\text{Min} \int_0^T \ell(y(t),u(t))\mathrm{d}t + \Phi(y(T)); \begin{cases} \dot{y}(t) &= f(y(t),u(t)), \quad t \in [0,T]; \\ 0 &\leq g(y(t),u(t)), \quad t \in [0,T]; \\ y(0) &= y^0. \end{cases} \tag{13}$$

Here $g : I\!R^n \times I\!R^m \to I\!R^r$ is a mapping of class C^∞. The idea of logarithmic penalty is, for a (small) parameter ε, to introduce the penalizer running cost

$$\ell_\varepsilon(y,u) := \begin{cases} \ell(y,u) - \varepsilon \sum_{i=1}^q \log[g_i(y,u)] & \text{if } g_i(y,u) > 0,\ i = 1,\ldots,r, \\ +\infty, & \text{otherwise,} \end{cases} \tag{14}$$

and the related penalized optimal control problem:

$$\text{Min} \int_0^T \ell_\varepsilon(y(t),u(t))\mathrm{d}t + \Phi(y(T)); \begin{cases} \dot{y}(t) &= f(y(t),u(t)), \quad t \in [0,T], \\ y(0) &= y^0. \end{cases} \tag{15}$$

A feasible point of this problem is such that the state equation holds, $g_i(y(t),u(t)) > 0$ a.e., and in addition, $\log[g_i(y(t),u(t))]$ is Lebesgue integrable, for all $i = 1$ to r.

Little is known on the theoretical properties of logarithmic penalty for optimal control problems. The only reference we know is Bonnans and Guilbaud [7]. Under the hypothesis

that running constraints are in fact bound constraints on controls, they prove convergence of optimal control and states (and also costates) for either (strongly) convex linear quadratic problems, or in a larger class in which the Hamiltonian function is convex with respect to the control. In addition it is proved that controls remain at distance at least $c\varepsilon$ from its bounds, where $c > 0$ is uniform for all sufficiently small ε. Asymptotic expansions of the solution for small ε might help for designing efficient algorithms; a rather specific case of control constraints is discussed in Alvarez, Bonnans and Laurent-Varin [1]. Nothing seems to have been done for problems with state constraints.

In practice an initial feasible point is often unknown. In that case it is advisable to rewrite the running constraint as

$$g(y(t), u(t)) - e(t) = 0; \;\; e(t) \geq 0, \quad t \in [0, T], \tag{16}$$

and the penalized problem as

$$\begin{array}{l} \text{Min} \displaystyle\int_0^T \left[\ell_\varepsilon(y(t), u(t)) - \varepsilon \sum_{i=1}^{q} \log[e_i(t)] \right] \mathrm{d}t + \Phi(y(T)); \\ \dot{y}(t) = f(y(t), u(t)); \;\; g(y(t), u(t)) - e(t) = 0; \;\; e(t) \geq 0, \;\; t \in [0, T]; \\ y(0) = y^0. \end{array} \tag{17}$$

Clearly problems (15) and (17) are equivalent. The advantage of formulation (17) is that, given a starting point for which the slack variable is positive, we may still consider Newton steps on the optimality system (even if the equality constraints are not satisfied). So our practical implementations are based on formulation (17).

The penalized problem (15) is unconstrained with an integral and final cost. Adding an additional state variable y_{n+1} such that $y_{n+1}(0) = 0$ and $\dot{y}_{n+1}(t) = \ell_\varepsilon(y(t), u(t))$ allows to reduce it to a problem with final cost only, that can be discretized by the Runge-Kutta schemes discussed in subsection 2. For a given value of ε the error analysis of that section applies when the steps vanish. The constants, however, will depend on ε.

An interesting open problem is to obtain error estimates for given parameter ε and discretization steps. This is obviously difficult, since without logarithmic penalty, one may encounter a reduction order when state constraints are active (it is known that in general a Runge-Kutta scheme applied to an algebraic differential system suffers from order reduction). In addition junction points for constraints (i.e., times when the constraint begins or stops to be active) need a specific analysis. In the case of control constraints and strongly convex Hamiltonian functions, some preliminary results will be presented in [23].

Related to this is the question of the choice of a path of convergence, i.e., at which relative speed should the parameter ε and the discretization steps converge to 0. Again this is essentially an open problem. In our present implementation we simply require that the error estimate is not more than a constant times ε.

The idea of logarithmic penalty could as well have been used on a discretized version of problem (13). In view of the discussion of section 2, a natural extension of discretization (2)

would be

$$\left\{\begin{array}{lll} \text{Min} & \Phi(y_N); & \\ y_{k+1} & = y_k + h_k \sum_{i=1}^{s} b_i f(y_{ki}, u_{ki}), & k = 0, \ldots, N-1, \\ y_{ki} & = y_k + h_k \sum_{j=1}^{s} a_{ij} f(y_{kj}, u_{kj}), & i = 1, \ldots, s, \\ 0 & \leq g(y_{ki}, u_{ki}), & i = 1, \ldots, s, \\ y_0 & = y^0. & \end{array}\right. \tag{18}$$

where we use here the extended formulation: y has $n+1$ components and $f_{n+1} = \ell$. It appears that the logarithmic penalization of (18) is nothing but the Runge-Kutta discretization of (17). This remark is useful since, if we estimate the distance of the solution u_h^ε of the discretized problem to the solution $\bar{u}$ of the original one by the following inequalities (for conveniently chosen norms) then each norm in the right hand side corresponds either to a discretization estimate or to the estimate of variation due to logarithmic penalty:

$$\left\{\begin{array}{l} \|\bar{u} - u_h^\varepsilon\| \leq \|\bar{u} - u^\varepsilon\| + \|u^\varepsilon - u_h^\varepsilon\|, \\ \|\bar{u} - u_h^\varepsilon\| \leq \|\bar{u} - u_h\| + \|u_h - u_h^\varepsilon\|. \end{array}\right. \tag{19}$$

Here u^ε (resp. u_h) denotes the solution of problem (15) (resp. (18)).

5.2 The Dogleg procedure

The first-order optimality conditions of the optimal control problem with logarithmic penalty is a set of nonlinear equations, say of the form $F(X) = 0$ where X is the set of optimization parameters and Lagrange multipliers, and F is a smooth mapping with square Jacobian matrix. We solve the nonlinear equation by applying a Dogleg method to the least square formulation $\text{Min}_X \|F(X)\|^2$, as in Moré and Sorensen [27] and Powell [28].

6 Application to Goddard's problem

Goddard's problem, stated in 1919 (see e.g. Tsiotras and Kelley [31]) consists in maximizing the final altitude of a rocket with vertical ascent. The model involves three state variables: altitude h, speed v and mass m, and the thrust F as control variable. The objective is to maximize the final altitude, with a prescribed final mass, and free final time. The dynamics is

$$\left\{\begin{array}{rcl} \dot{h}(t) & = & v(t), \\ \dot{v}(t) & = & (F(t) - D(v(t), h(t)))/m(t) - g(h(t)), \\ \dot{m}(t) & = & -F(t)/c \end{array}\right. \tag{20}$$

where D is the drag function, with arguments speed and altitude, $g(h)$ is the gravity field, and $c > 0$ is the speed of ejection of gas. The final mass is given, and we have a bound on thrust: $0 \leq F(t) \leq F_{\max}$.

Starting from a speed close to zero, it happens that the optimal thrust is not (as numerical experiments show) to set the thrust at $F_{\max}$ as long as the final mass is not attained. The reason is that aerodynamic forces would, in that case, tend to reduce significantly the speed.

Therefore (although this is apparently not proved yet) the optimal law is to set the thrust at $F_{\max}$ for a certain time, then to set it to values in $(0, F_{\max})$ (this is the singular arc) and finally to set it to zero. The term singular arc comes from the fact that, since the control appears linearly in the dynamics (and not in the cost function) it cannot be computed by minimizing the Hamiltonian function when the latter is a constant function of the control. There exists a nice piece of theory that allows to obtain the control law during the singular arc (see [10, 31]), but of course we do not use it in our numerical experiments.

Although our code may use arbitrary Runge-Kutta coefficients, we have used the Gauss-Legendre scheme of order 2 and of order 4, so that we may see how the code behaves when there are many time steps. The results are synthetized in Table 1 and 2, where for each major iteration, corresponding to the resolution of the penalized problem for a given value of ε. We display the values of the major iteration It, the number of dogleg iterations n_{it}, the size N of the mesh, the number of points to be added (predicted by the refinement procedure), the current value of ε and the threshold on error estimates. Observe the great accuracy of the refinement procedure, that essentially predicts in one shot the points to be added, for order 2 as well for order 4 schemes. The number of inner iterations remains small. It is fair to say, however, that the reduction procedure for ε is here quite conservative (division by 2). Stronger reductions will be performed in the other applications presented in this paper.

The optimal control and states are displayed on figures 1-4, for each value of the parameter ε. We observe the barrier effect for large ε, and the convergence to a three arc-structure, one of them being singular.

We draw in figures 6 and 7, for each value of ε, the switching function derivative of Hamiltonian function w.r.t. the control), whose expression is $p_v/m - p_m/c$. For our numerical test we choose $D(v,h) := K_D v^2 \exp(-\frac{h_c}{h_0}(h-h_0))$ and $g(h) = g_0 h_0^2/h^2$, with $K_D = 310$, $c = 0.5$, $h_0 = 1$, $h_c = 500$ and $F_{\max} = 3.5$.

The density of mesh, after each major iteration is displayed in figures 8 and 9. The original mesh has 100 equal steps. We can observe on the final mesh a high density in the region corresponding to the singular arc.

7 Fuller's problem

Let us show how our method behaves when applied to Fuller's problem [14, 15]. This is a simple academic problem with, as we will see, a non regular junction point. The problem is as follows:

$$\text{Min } \tfrac{1}{2}\int_0^T x_1^2(t)\mathrm{d}t;\ \ \dot{x_1}(t) = x_2(t),\ \dot{x_2}(t) = u(t) \in [-1,1],\ t \in [0,T];\ x(0) = a,$$

where $a \in I\!R^2$ is given. Note that the Hamiltonian function $H(x,u,p) = \frac{1}{2}x_1^2 + p_1 x_2 + p_2 u$ is, as in Goddard's problem, linear w.r.t. the control variable. It was shown in [14, 15] that an infinite number of switches may occur before reaching the singular arc where the state has a null value.

It	N	n_{it}	Points	ε	E
1	100	13	+43	1.0e-3	5.0e-4
	143	3	+1		
	144	3	+0		
2	144	6	+63	5.0e-4	2.5e-4
	207	3	+1		
	208	3	+0		
3	208	6	+102	2.5e-4	1.25e-4
	310	3	+0		
4	310	6	+163	1.25e-4	6.25e-5
	473	3	+0		
5	473	5	+283	6.25e-5	3.125e-5
	756	3	+0		
6	756	5	+484	3.125e-5	1.5625e-5
	1240	3	+0		
7	1240	5	+862	1.5625e-5	7.8125e-6
	2102	2	+0		
8	2102	5	+1458	7.8125e-6	3.90625e-6
	3560	2	+0		
9	3560	5	+2663	3.90625e-6	1.95313e-6
	6223	2	+0		

Table 1: Goddard's Problem: order 2 scheme

It	N	n_{it}	Points	ε	E
1	100	14	+0	1.0e-3	5.0e-4
2	100	8	+0	5.0e-4	2.5e-4
3	100	8	+0	2.5e-4	1.25e-4
4	100	9	+0	1.25e-4	6.25e-5
5	100	10	+1	6.25e-5	3.125e-5
	101	6	+0		
6	101	15	+9	3.125e-5	1.5625e-5
	110	7	+1		
	111	5	+0		
7	111	7	+21	1.5625e-5	7.8125e-6
	132	8	+0		
8	132	9	+19	7.8125e-6	3.90625e-6
	151	8	+1		
	152	3	+0		
9	152	6	+39	3.90625e-6	1.95313e-6
	191	4	+0		
10	191	6	+69	1.95313e-6	9.76563e-7
	260	7	+0		

Table 2: Goddard's Problem: order 4 scheme

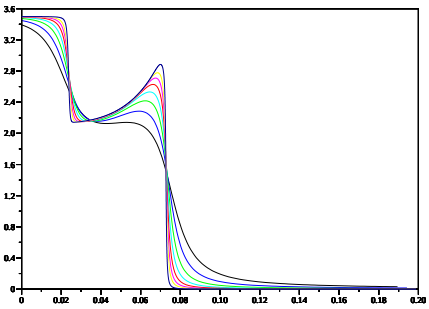


Figure 1: Thrust law / Time

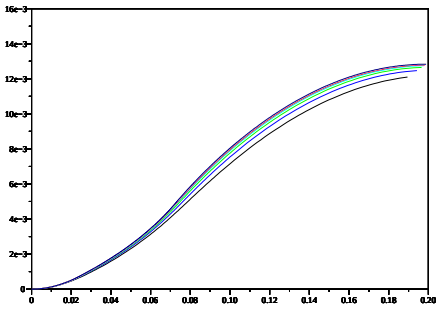


Figure 2: Altitude / Time

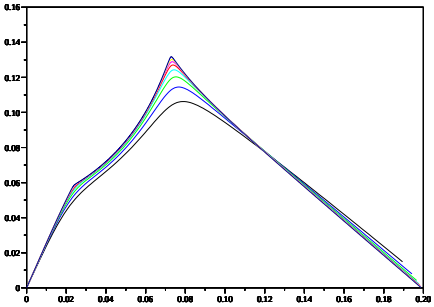


Figure 3: Velocity / Time

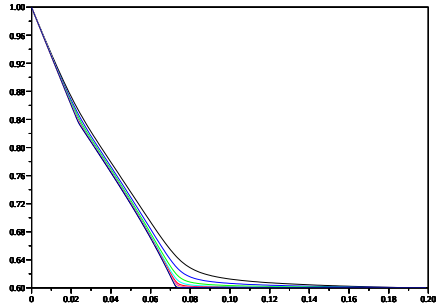


Figure 4: Mass / Time

Figure 5: Goddard's problem

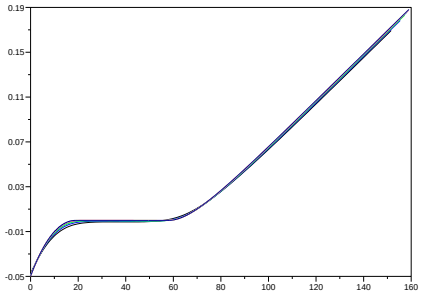


Figure 6: Switching function (Goddard)

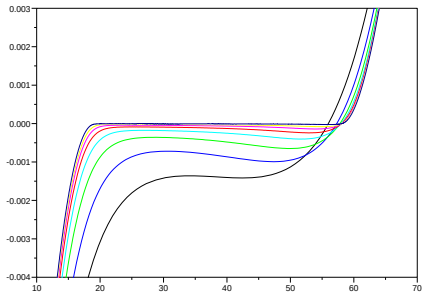


Figure 7: Zoom on Switching function

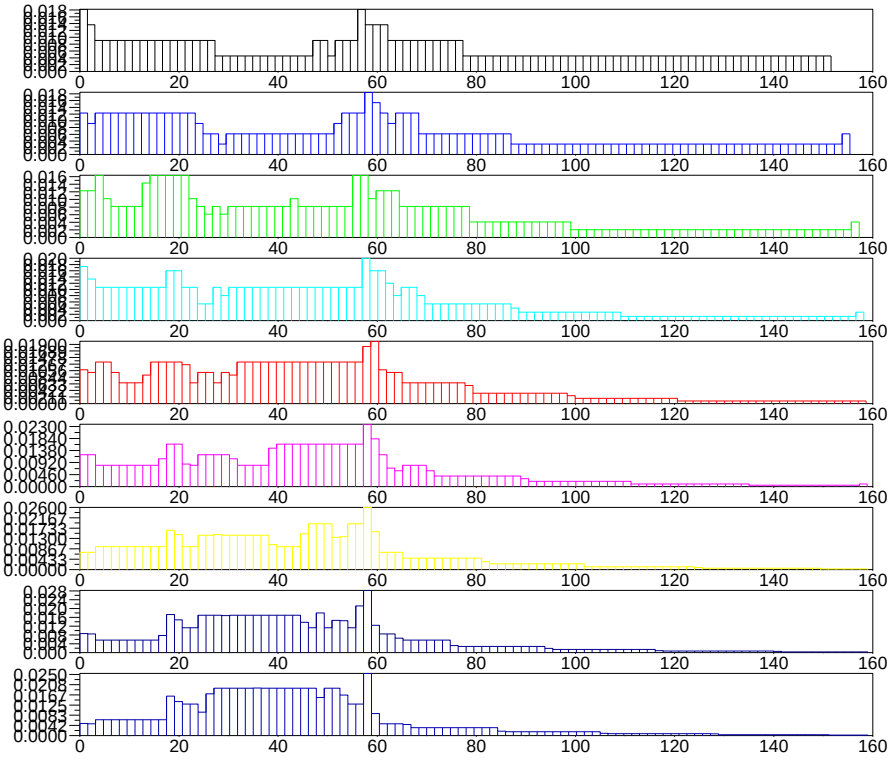


Figure 8: Mesh order 2 (Goddard)

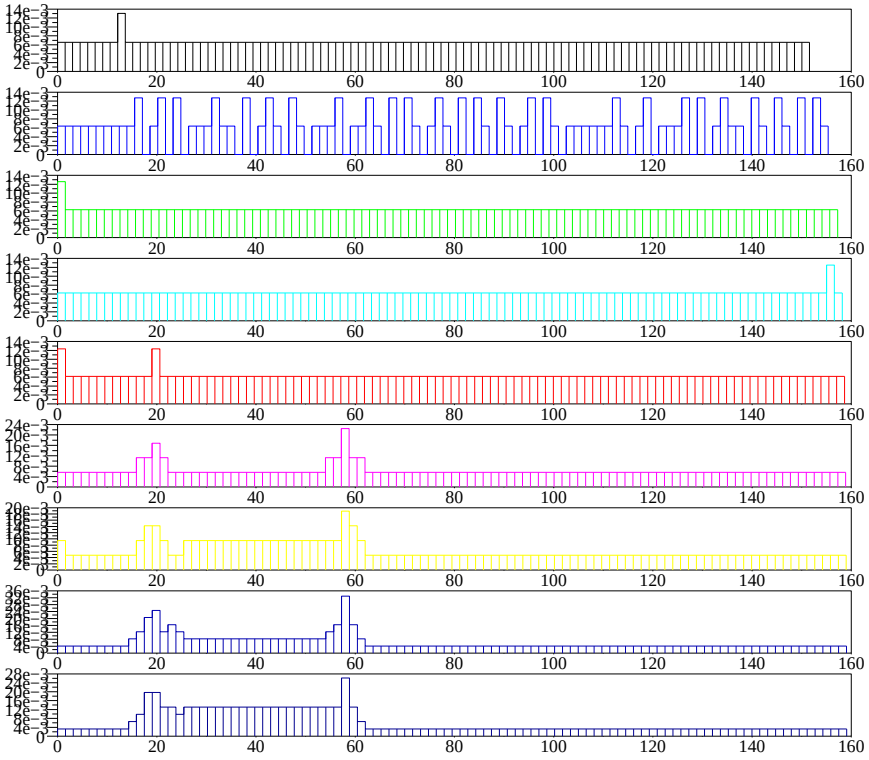


Figure 9: Mesh order 4 (Goddard)

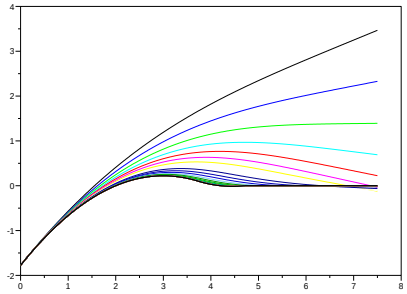


Figure 10: x_1 / Time

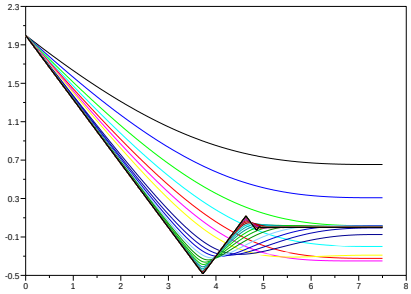


Figure 11: x_2 / Time

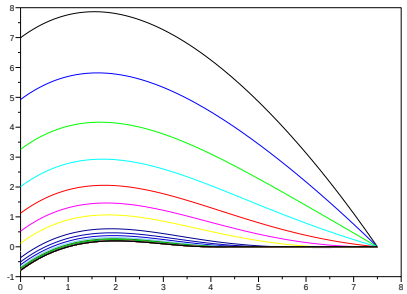


Figure 12: p_1 / Time

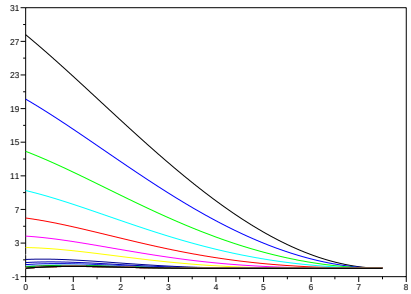


Figure 13: p_2 / Time

Figure 14: Fuller's problem

On this example the refinement procedure had difficulties, probably in relation with the above mentioned behavior, so we display the results with a fixed grid of 400 points. We set a to the value $(-1.778, 2.0)$ in order to obtain an infinite number of switches.

Figures 10 to 14 show the states, costate and control variables. It happens that the algorithm does not converge if ε decreases too rapidly. Therefore we choose a linear evolution of the form : $\varepsilon_{k+1} = \frac{1}{2}\varepsilon_k$. We start with $\varepsilon_0 = 16$, and stop for $\varepsilon_{36} = 4.65710^{-10}$.

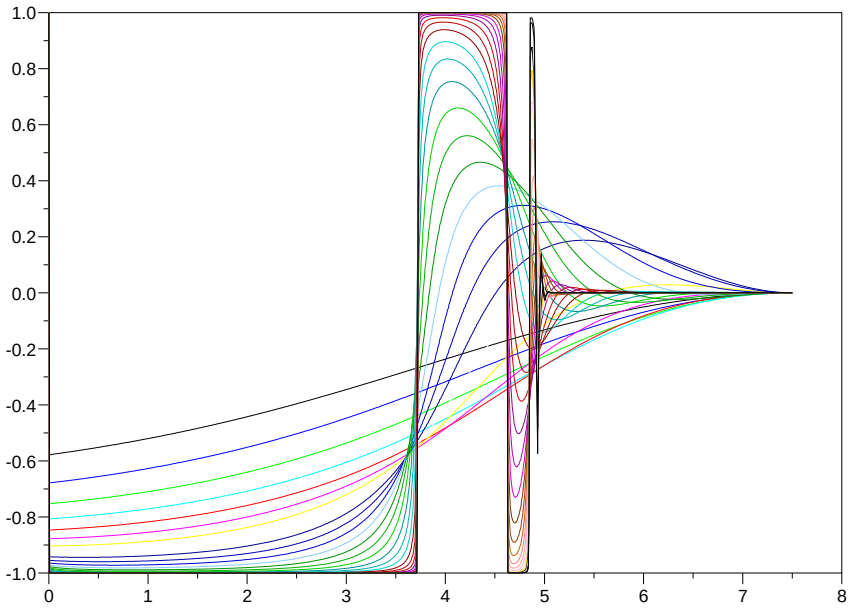


Figure 15: u / Time (Fuller)

8 Atmospheric Reentry

8.1 Model

In this section we study a problem of atmospheric reentry for a space shuttle, already considered in Betts [5, Chap 5, p. 133] to which we refer for a precise statement of state equation and constraints. It is enough to say that there are six state variables: altitude, longitude, latitude, velocity, flight path angle, and azimuth. The two control variables are the angle of attack and bank angle. There are bound constraints on controls, and an optional running constraints on the heating flux, all of them identical to those in [5]. In addition we have an optional running constraint of non positive flight path angle, which prevents skipping.

We maximize either the cross range or the range. Since we start at latitude 0 moving eastward, cross range maximization amounts to maximize the final altitude, while range maximization amount to maximize the final longitude. We use again the Gauss-Legendre discretization schemes.

We number problems as follows (the third column refers to constraints other than bounds on control variables):

Problem Number	Cost function	Constraints
1	Max Cross range	None
2	Max Cross range	Heating
3	Max Cross range	Path Angle
4	Max Cross range	Heating and path angle
5	Max Range	None
6	Max Range	Heating
7	Max Range	Path Angle

8.2 Initialization procedure

As already mentioned, the initialization procedure is an important component for the success of the method, especially for reentry trajectories since it is known that the trajectory is very sensitive to the control.

8.2.1 Max cross range with only control constraints

We initialize variables by the following heuristics: we integrate the optimality system starting from $t = 0$ and with zero initial costate. Of course the final conditions are not satisfied.

8.2.2 Maximum cross range with heating flux or path angle constraints

For these problems we initialize the control, state and costate with the value obtained by solving the problem without heating flux or path angle constraints, with a given value of the

penalty parameter $\varepsilon \geq 1$. Then we initialize the slack variables and Lagrange multipliers in the following way:

$$e_i = \min(\max(\tilde{g}_i, 1 - \sqrt(\varepsilon)), -1 + \sqrt{\varepsilon}) \quad (21)$$

$$\lambda = \varepsilon(e-1)^{-1}; \quad \mu = \varepsilon(e+1)^{-1}. \quad (22)$$

Here $\tilde{g}$ is the scaled path constraint function, that (if the constraint holds) belongs to $[-1, 1]$. Thus the inequality conditions on the slack variables are satisfied.

8.2.3 For max range without any constraints

We initialize the problem with the solution obtained for the max cross range cost function, and with $\varepsilon = 16$.

8.2.4 For max range with each constraints

We initialise the problem with the solution obtained for the max range cost function, and with $\varepsilon = 1$.

8.3 Update of the penalty parameter

We update the penalty parameter ε in the following way :

$$\varepsilon_{k+1} = \min\left(\tfrac{1}{2}\varepsilon_k, \varepsilon^{3/2}\right).$$

8.4 Numerical results for cross range maximization

All figures display the evolution of some variable as a function of time. We first display the results for cross range maximization.

8.4.1 Cross range maximization with only control constraints (Problem 1)

We first display the results for a grid of 100 equidistant points, without refinement. Table 3 shows the evolution of the interior point parameter, the number of dogleg iterations for each ε, the CPU time for each ε and the performance index (here, the final latitude), to be compared with the value 34.1412 degree given in [5]. The following graph represents the evolution of the performance index as function of ε.

We observe that, although ε decreases rapidly in the last iterations, the number of Newton steps at each iteration remains small and, in fact, tends to decrease. The time needed for one Newton step is approximately 3 seconds. Figures 16 and 17 display the performance index and the angle of attack as function of ε.

In table 4 we can notice the use the refinement policy. We can notice that numerically, there is no difference between performance index for the resolution without refinement. The

density of discretization points for obtaining this precision is displayed in the histogram of Figure 20.

There is a fast pull-up manoeuvre close to the final time in order to comply with the prescribed final value of the flight path angle. This can be seen in the display of the angle of attack, and accordingly the density of discretization points is higher at the end of the trajectory.

Figures 21 to 25 compare the outputs of problems 1 to 4.

8.4.2 Cross range maximization with flux heating constraint (Problem 2)

We display our results for the problem with heating flux (problem 2), without refinement of the initial grid of 100 equidistant points. The execution report is given in Table 5, and the values of heating flux and angle of attack, as function of time, for several values of ε, are given in Figures 18 and 19. We can observe that the angle of attack is now far from being almost constant, as it was in Problem 1. With this additional constraint we obtain a worse (as was to be expected) index performance, by about 10 %.

8.4.3 Cross range maximization with path angle constraint (Problem 3)

We add the constraint of non positive path flight angle. As for heating constraint, we initialize with the unconstrained problem trajectory, and obtain results in Table 6. It appears now that the degradation of the criterion is of only about 0.1 %, although the constraint was not satisfied in Problem 1.

8.4.4 Cross range maximization with heating flux and flight path angle constraint (Problem 4)

Problem 4 involves both constraints of flight path angle constraint and heating constraint. The algorithm converges untill $\varepsilon = 0.0056$ but not for smaller values (perhaps a more conservative policy for the decrease of ε should be considered). See Table 7.

8.5 Numerical results, for range maximization

In the modelisation chosen, we would like to try an other performance index. The natural new one is to maximize the forward range of the Space Shuttle. Figures 26 to 30 compare the outputs of the various range maximization problems.

8.5.1 Range maximization with only control constraints (Problem 5)

We show in table 8 the behavior of the algorithm. The number of dogleg iterations decreases and in the last major iterations is no more than 3. Note that the cost function is almost constant for $\varepsilon \leq 0.125$.

ε	Iterations	CPU time	Final latitude
16	37	1 min 12s	24.31
8	10	23s	24.82
4	8	20s	25.79
2	8	21s	27.48
1	7	20s	29.67
0.5	7	18s	31.59
0.25	6	18s	32.94
0.125	7	22s	33.66
$4.41942\,10^{-02}$	6	19s	34.01
$9.29068\,10^{-03}$	6	20s	34.11
$8.95512\,10^{-04}$	5	16s	34.1283
$2.67983\,10^{-05}$	5	17s	34.1288

Table 3: Problem 1: Cross range

8.5.2 Range maximization with heating flux constraints (Problem 6)

For this problem the number of dogleg iterations is much larger. In the last iterations the cost function does not change much, and hence, we may hope that the last solution computed with $\varepsilon = 0.044$ is not far from the optimum. Interestingly, the problem seems to have three boundary arcs.

8.5.3 Range maximization with path angle constraints (Problem 7)

Here the smallest value of ε for which convergence occurs is ε=8.95E-4 and again the cost function does not change much in the last iterations, so that we may hope to have computed a reasonably good approximate solution.

9 Conclusion and perspectives

We have presented an approach for solving optimal control problems, based on the interior-point methodology combined with a refinement procedure. Our present implementation allows to solve efficiently various applied problems. The cost of a Newton step is of the same order as a simulation with a fully implicit scheme. The refinement procedure leads to higher computing times, but since the number of operations is proportional to the number of time steps, and the number of added steps seems to be minimal, this is the price to pay for precision (perhaps in some example we could test if some of the time steps could be merged). The software, however, encountered difficulties for the reentry problem when both constraints on flight path angle and heating flux are active, for small values of the parameter ε.

ε	Iterations	Final latitude	Time steps	CPU time
16	1	24.31	100	3s
8	10	24.82	100	23s
4	8	25.79	100	23s
2	8+3	27.48	100+2	29s
1	7+3	29.68	102+42	37s
0.5	6+3+2	31.59	144+49+9	1min 45s
0.25	7+3	32.94	202+31	2min 40s
0.125	7+2	33.66	233+27	2min 56s
0.0441942	6+2	34.01	260+93	3min 56s
0.0092907	6+2+2	34.11	353+366+5	17min 9s

Table 4: Problem 1: Cross range with refinement

ε	Iterations	CPU time	Final latitude
0.1	13	1min 02s	27.85 deg
0.0316228	5	26s	29.62 deg
0.0056234	4	19s	30.42 deg
0.0004217	5	24s	30.59 deg
0.0000087	6	29s	30.61 deg

Table 5: Problem 2: Cross range, heating flux constraint

ε	Iterations	CPU time	Final latitude
0.1	14	39s	33.46
0.032	11	38s	33.95
0.0056	9	35s	34.06
0.00042	7	30s	34.087
0.0000087	9	41s	34.090

Table 6: Problem 3: Cross range, path angle constraint

ε	Iterations	CPU time	Final latitude
0.1	15	1 min 19s	26.98
0.032	11	1 min 12s	29.45
0.0056	11	1 min 7s	30.38

Table 7: Problem 4: Cross range, heating flux and path angle constraints

ε	Iterations	Longitude	CPU time
16	28	171.53	1min 16s
8	11	176.47	43s
4	9	181.13	36s
2	8	184.55	32s
1	7	186.27	28s
0.5	5	186.89	20s
0.25	3	187.08	14s
0.125	3	187.16	13s
0.0441942	3	187.16	13s
0.00929068	3	187.16	14s
0.000895512	2	187.16	10s
$2.67983e-05$	2	187.16	13s

Table 8: Problem 5: Max range

ε	Iterations	Longitude	CPU time
1	386	167.95	24 min 06s
0.5	21	170.79	1 min 44s
0.25	20	172.36	1 min 36s
0.125	17	173.22	1 min 21s
0.0441942	18	173.86	1 min 26s

Table 9: Problem 6: Max range, heating flux constraint

ε	Iterations	Longitude	CPU time
0.125	11	178.09	1 min 09s
0.0441942	5	179.66	0 min 47s
0.00929068	6	180.42	0 min 53s
0.000895512	4	180.61	1 min 11s
$2.67983e-05$	6	180.63	1 min 23s

Table 10: Problem 7: Max range, path angle constraint

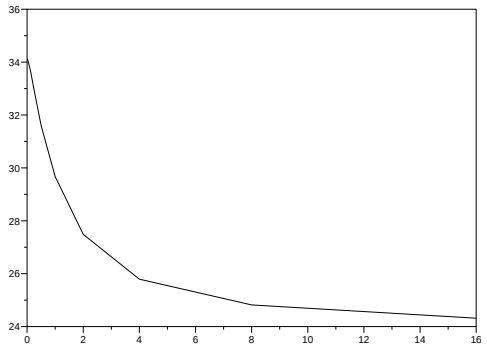


Figure 16: Final latitude (deg) / ε

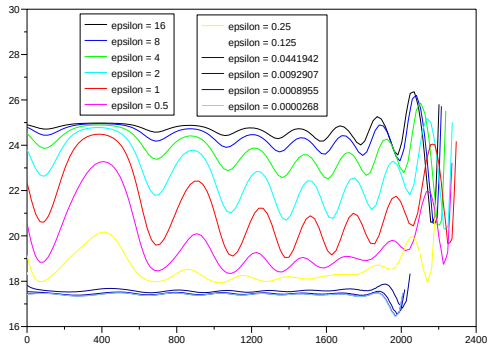


Figure 17: Angle of Attack (deg) / Time (s)

Graphs for evolution w.r.t. ε in Problem 1

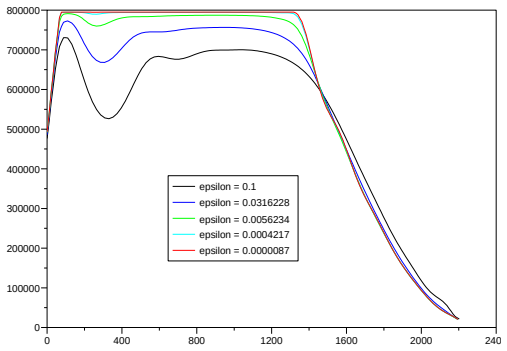


Figure 18: Heating (W/m^2) / Time (s)

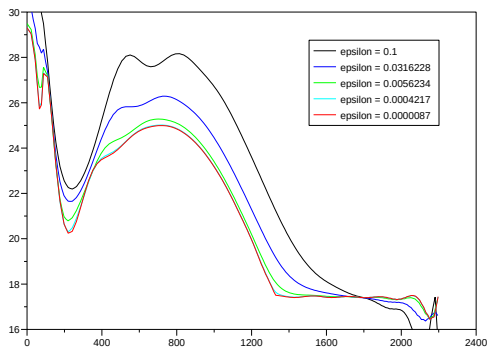


Figure 19: Angle of Attack (deg) / Time (s)

Graphs for evolution w.r.t. ε in Problem 2

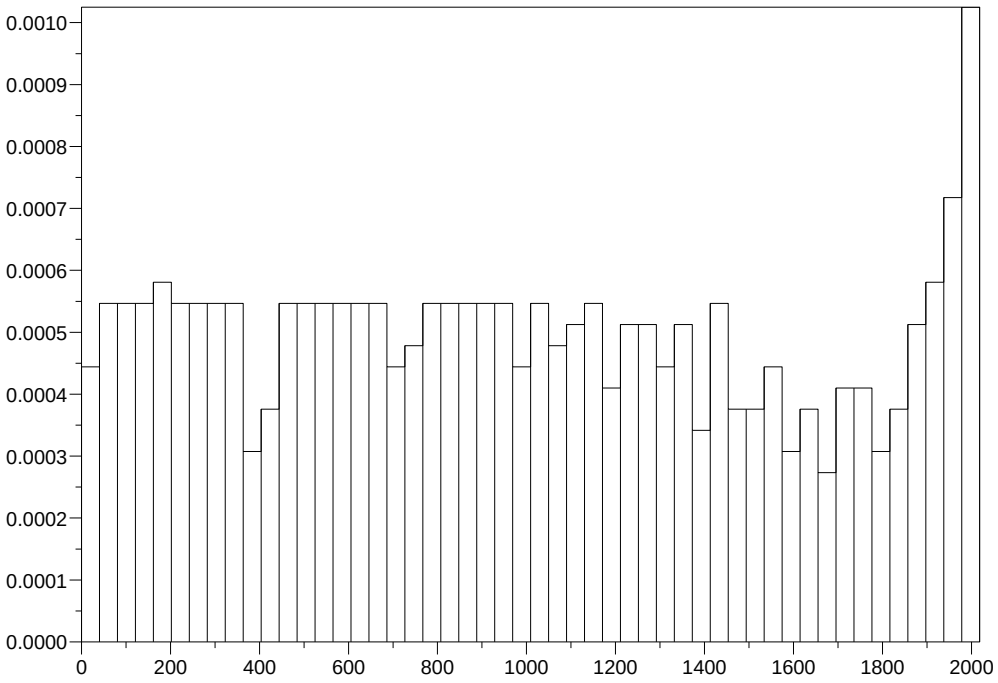


Figure 20: Final density mesh in Problem 1

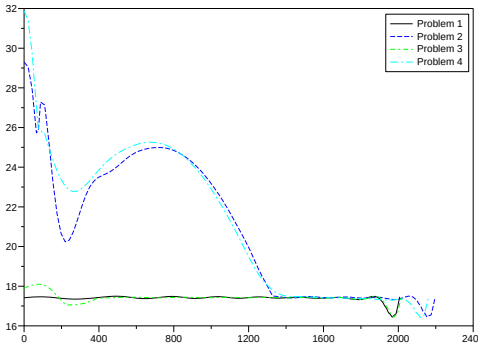


Figure 21: Angle of attack (deg) / Time (s)

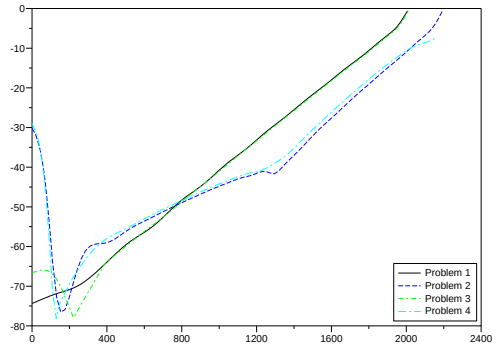


Figure 22: Bank angle (deg) / Time (s)

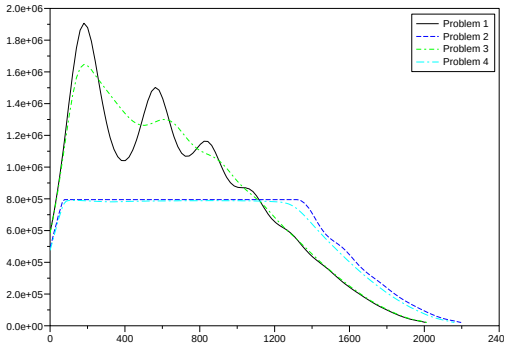


Figure 23: Heating (W/m^2) / Time (s)

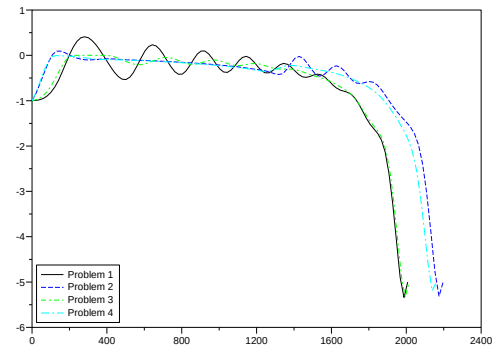


Figure 24: Flight path angle (deg) / Time (s)

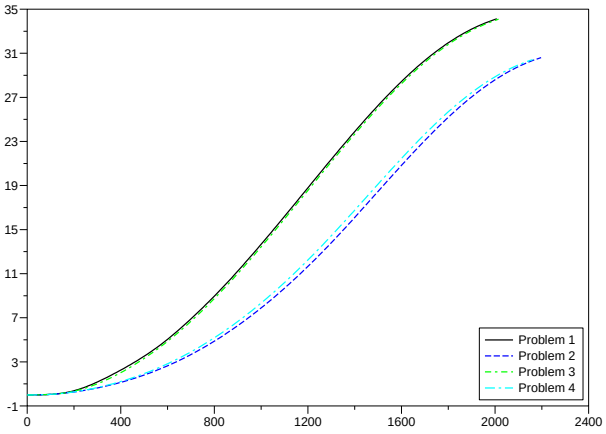


Figure 25: Latitude (deg) / Time (s)

Graphs for Problem 1 to 4

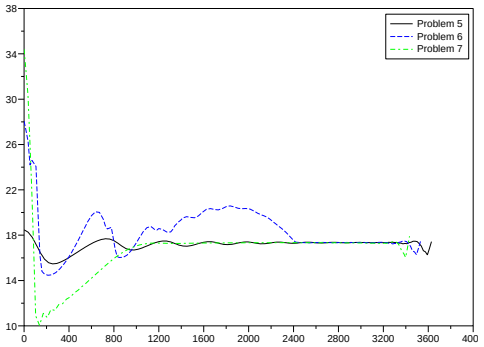


Figure 26: Angle of attack (deg) / Time (s)

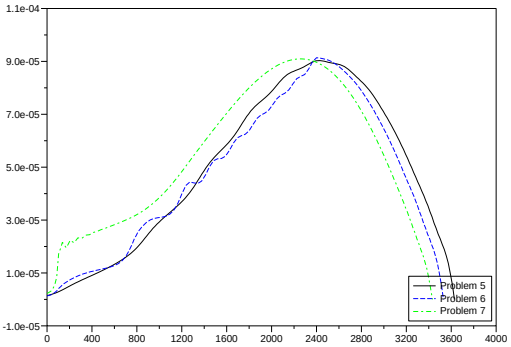


Figure 27: Bank angle (deg) / Time (s)

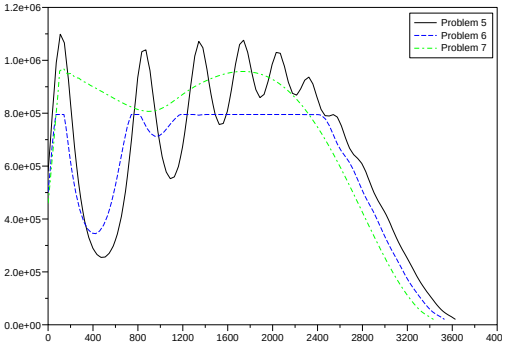


Figure 28: Heating (W/m^2) / Time (s)

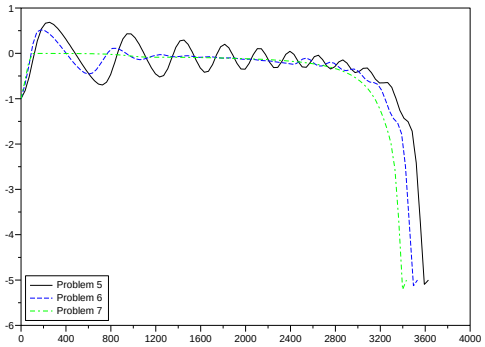


Figure 29: Flight path angle (deg) / Time (s)

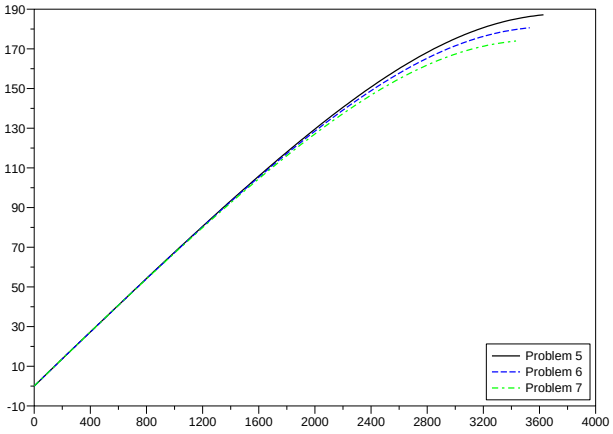


Figure 30: Longitude (deg) / Time (s)

Graphs for Problem 5 to 7

Theoretical advances might help to give precise error estimates (to the solution of the original problem) and to understand what is the optimal path in the (ε, E) plane (E is the estimate of integration errors). Also, our dogleg procedure is quite simple and would deserve to be improved by taking into account the nature of optimality conditions. The way of decreasing the parameter ε is important for the efficiency of the method. We also observed that factorization is much more expensive that solving (when measured by the number of operations; the need for memory is similar). Therefore a simplified Newton strategy might help. It is therefore clear that there is a lot of room for improvement for this method.

References

[1] F. Alvarez, J.F. Bonnans, and J. Laurent-Varin. Expansion of solutions of optimal control problems with logarithmic penalty: a special case. To appear, 2005.

[2] N. Bérend, J.F. Bonnans, J. Laurent-Varin, M. Haddou, and C. Talbot. Fast linear algebra for multiarc trajectory optimization. Nonconvex Optimization and Its Applications Series. Kluwer, 2005.

[3] M. Bergounioux, M. Haddou, M. Hintermüller, and K. Kunisch. A comparison of a Moreau-Yosida-based active set strategy and interior point methods for constrained optimal control problems. *SIAM Journal on Optimization*, 11:495–521 (electronic), 2000.

[4] J.T. Betts. Survey of numerical methods for trajectory optimization. *AIAA J. of Guidance, Control and Dynamics*, 21:193–207, 1998.

[5] J.T. Betts. *Practical methods for optimal control using nonlinear programming.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2001.

[6] J.T. Betts and W.P. Huffman. Application of sparse nonliner programming to trajectory optimization. *AIAA J. of Guidance, Control and Dynamics*, 15:198–206, 1992.

[7] J.F. Bonnans and Th. Guilbaud. Using logarithmic penalties in the shooting algorithm for optimal control problems. *Optimal Control, Applications and Methods*, 24, 2003. To appear.

[8] J.F. Bonnans and G. Launay. Large scale direct optimal control applied to a re-entry problem. *AIAA J. of Guidance, Control and Dynamics*, 21:996–1000, 1998.

[9] J.F. Bonnans and J. Laurent-Varin. Computation of order conditions for symplectic partitioned runge-kutta schemes with application to optimal control. Technical Report 5398, INRIA, 2004. http://www.inria.fr/rrrt/rr-5398.html.

[10] A. E. Bryson and Y.-C. Ho. *Applied optimal control.* Hemisphere Publishing, New-York, 1975.

[11] R. Bulirsch, E. Nerz, and H.J. Peschand O. von Stryk. *Combining direct and indirect methods in optimal control: range maximization of a hang glider*, pages 273–288. Birkhäuser, Basel, 1993.

[12] A.L. Dontchev and W.W. Hager. The Euler approximation in state constrained optimal control. *Mathematics of Computation*, 70:173–203, 2001.

[13] A.L. Dontchev, W.W. Hager, and V.M. Veliov. Second-order Runge-Kutta approximations in control constrained optimal control. *SIAM Journal on Numerical Analysis*, 38:202–226 (electronic), 2000.

[14] A.T. Fuller. Relay control systems optimized for various performance criteria. In *Proc. IFAC Congress, Moscow*, pages 510–519. Butterworth, London, 1961.

[15] A.T. Fuller. Study of an optimum non-linear control system. *J. of Electronics and Control*, 15:63–71, 1963.

[16] P.E. Gill, W. Murray, and M.H. Wright. *Practical optimization.* Academic Press, London, 1981.

[17] W. Hager. Runge-Kutta methods in optimal control and the transformed adjoint system. *Numerische Mathematik*, 87(2):247–282, 2000.

[18] E. Hairer, Ch. Lubich, and G. Wanner. *Geometric numerical integration*, volume 31 of *Springer Series in Computational Mathematics.* Springer-Verlag, Berlin, 2002.

[19] E. Hairer, S. P. Nørsett, and G. Wanner. *Solving ordinary differential equations. I*, volume 8 of *Springer Series in Computational Mathematics.* Springer-Verlag, Berlin, second edition, 1993.

[20] E. Hairer and G. Wanner. *Solving ordinary differential equations. II.* Springer-Verlag, Berlin, second edition, 1996.

[21] S.P. Han. A globally convergent method for nonlinear programming. *J. Optimization theory and applications*, 22:297–309, 1977.

[22] D. Kraft. Finite-difference gradients versus error-quadrature gradients in the solution of parameterized optimal control problems. *Optimal Control, Appl. and Methods*, 2:191–199, 1981.

[23] J. Laurent-Varin. PhD thesis, Ecole Polytechnique, 2005. In preparation.

[24] F. Leibfritz and E.W. Sachs. Inexact SQP interior point methods and large scale optimal control problems. *SIAM Journal on Control and Optimization*, 38:272–293 (electronic), 1999.

[25] H. Maurer. On the minimum principle for optimal control problems with state constraints. Schriftenreihe des Rechenzentrum 41, Universität Münster, 1979.

[26] S.K. Mitter. Successive approximation methods for the solution of optimal control problems. *Automatica*, 3:135–149, 1966.

[27] J.J. Moré and D.C. Sorensen. Computing a trust region step. *SIAM J. Scientific and Statistical Computing*, 4:553–572, 1983.

[28] M.J.D. Powell. A method for nonlinear constraints in minimization problems. In *Optimization (Sympos., Univ. Keele, Keele, 1968)*, pages 283–298. Academic Press, London, 1969.

[29] M.J.D. Powell. Algorithms for nonlinear constraints that use Lagrangian functions. *Mathematical Programming*, 14:224–248, 1978.

[30] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*. Springer-Verlag, New-York, 1993.

[31] P. Tsiotras and H.J. Kelley. Drag-law effects in the Goddard problem. *Automatica*, 27-3:481–490, 1991.

[32] L.N. Vicente. On interior-point Newton algorithms for discretized optimal control problems with state constraints. *Optimization Methods and Software*, 8:249–275, 1998.

[33] S.J. Wright. Interior point methods for optimal control of discrete time systems. *Journal of Optimization Theory and Applications*, 77:161–187, 1993.



Unité de recherche INRIA Rocquencourt
Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399