Fast Linear Algebra for Multiarc Trajectory Optimization

N. Bérend[1], J. Frédéric Bonnans[2], J. Laurent-Varin[3], M. Haddou[4], and C. Talbot[5]

[1] Long-Term Design & System Integration Department, ONERA, BP 72, 29 av. division Leclerc, 92322 Châtillon, France (Nicolas.Berend@onera.fr)
[2] Projet Sydoco INRIA, B.P. 105, 78153 Le Chesnay, France (Frederic.Bonnans@inria.fr)
[3] Long-Term Design & System Integration Department, ONERA, and Projet Sydoco, INRIA, BP 105, 78153 Le Chesnay, France (Julien.Laurent-Varin@inria.fr)
[4] UMR 6628, MAPMO, BP 6759, 45067 Orléans Cedex 2, France (haddou@labomath.univ-orleans.fr)
[5] CNES, Launcher Directorate, Rond-Point de l'Espace, 91023 Evry Cedex, France (Christophe.Talbot@cnes.fr)

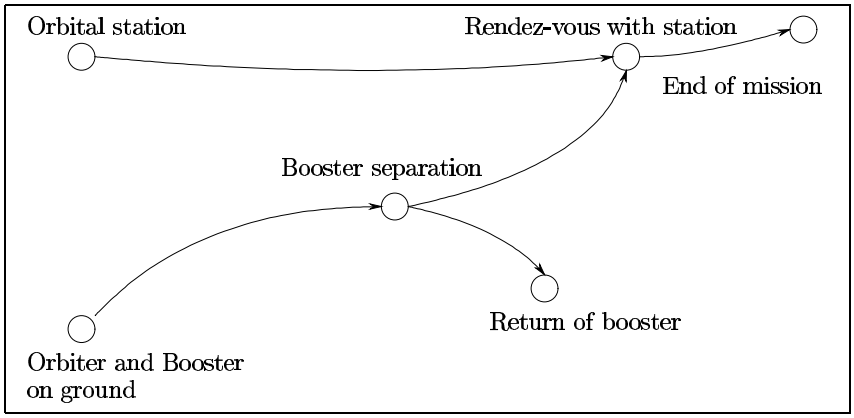
Summary. This paper presents some methods for solving in a fast and reliable way the linear systems arising when solving an optimal control problem by a Runge-Kutta discretization scheme, combined with an interior-point algorithm. Our analysis holds for a multiarc problem, i.e., when several arcs, each of them associated with a dynamics and integral cost, are linked by junction points, called nodes; with the latter are associated junction conditions and a cost function.

Our main result is that a sparse QR band factorization combined with a specific elimination procedure for arcs and nodes allows to factorize the Jacobian of the discrete optimality system in a small number of operations. Combined with an "optimal" refinement procedure, this gives an efficient method that we illustrate on Goddard's problem.

Key words: Optimal control, differential equations, Runge-Kutta and symplectic schemes, sparses algebra, band matrices, QR factorization.

1 Introduction

This paper discusses numerical methods for solving multi arc optimal control problems. Let us give an example of a possible mission to be optimized.



Here a separation between booster and orbiter occurs at some point; the orbiter meets a spatial station and both remain linked for some time. The return trajectory of the booster is to be optimized, as well as the one of the space station. While arcs are connected here through separation or rendez-vous, one may think also of other arcs corresponding to possible events, for instance of failure, that requires to change trajectories.

We concentrate on direct methods, which solve approximately a time discretized version of the optimal control problem, and are able to refine discretization. See [4, 11, 18, 19] for an overview of numerical methods for optimal control problems, and [5, 7] for specific direct methods.

The paper is organized as follows. Section 2 deals with single arc problems. We discuss the error analysis in subsection 2.1, and linear algebra issues in subsection 2.2. An "optimal" refinement technique is presented in 2.4, and 2.5 discusses the monitoring of interior-point algorithms. Section 3 is devoted to multiarc problems. Their structure is presented in subsection 3.1, and the linear algebra is analysed in subsection 3.2. Since our software is still under construction, we present only numerical results for the well known (single arc) Goddard problem in section 4.

2 Single arc problem

2.1 Framework and discretization

We start by describing an optimal control problem with a single arc and, to begin with, without constraints. We may assume without loss of generality (adding an additional state variable if necessary) that there is no integral cost, and we take into account only a final cost:

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

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. Introduce now a Runge-Kutta type discretization, as done in Hager [14]:

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

The positive time steps h_k are such that $\sum_{k=0}^{N-1} h_k = T$. Note that with each "inner state" y_{ki} is associated an inner control u_{ki}. The Runge-Kutta scheme is parameterized by its coefficients (a, b), an $s \times s$ matrix and a vector of $I\!R^n$ respectively. Assuming all coefficients b_i to be nonzero, it was shown by Hager [14] that the optimality system can be written in the following form:

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

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 for the optimality conditions of problem (1) stated below:

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

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 view of (4), this system is symplectic [15, 16, 17]. These references present the theory of order conditions for partitioned Kunge-Kutta schemes. One may find in [8] a simplification of order conditions for symplectic schemes, that allows to state them for order up to 6 (they were obtained in [14] for order up to 4). Yet these results apply only for unconstrained problems with strongly convex Hamiltonians; some results for for constrained optimal control problems may be found in [13, 12].

2.2 Linear algebra: problems without design variables

Here we present a fast and reliable approach to the numerical resolution of the linear systems arising when solving the discrete optimality conditions (3),

in the case when there is no design variable. The starting point is that the Jacobian matrix denoted A is, when variables are ordered by increasing values of time, a band matrix. Let q denote the band size, i.e., such that $A_{ij} = 0$ if $|i-j| > q$ (for instance, a diagonal matrix has band size 0). Remind that n (resp. m) denotes the number of states (resp. controls). Our implementation achieves a band size of $q = (s+1)(2n+m)+n$, where s is the number of inner stages in the Runke Kutta scheme. For a problem with n_g distributed constraints (upper and lower bounds), dealt with by the interior-point approach discussed later, we have a band size of $q = (s+1)(2n+m+2n_g)+n-1$. Note that at each time step there are $(s+1)(2n+m+2n_g)$ variables (state, costate, control and Lagrange multipliers) and there are dynamic constraints with the previous and next state and costate.

Lemma 1. *The number of multiplications for a QR factorization based on Givens rotations, applied to a matrix M of band size q_M, and having N_M lines, with N_M large w.r.t. q_M, is at most $6q_M^2 N_M + O(q_M^2)$ multiplications. The number of multiplications for solving the linear system (after factorization) is $6q_M N_M$.*

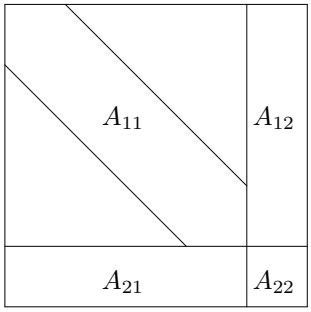
Proof. Row i eliminates element of index $(i+k, i)$ with k varying from 1 to q_M. This results in a fill-in from row i only. Therefore, when elimination of element $(i+k, i)$ occurs, row i of R has only $q+k$ elements, and hence, $4(q_M+k)+O(1)$ operations are performed. Since $\sum_{k=1}^{q_M}(q_M+k) = 3q_M^2/2 + O(q_M)$, the result follows.

Solving needs the resolutions of two linear systems with matrices R and Q. For the one with matrix R, we compute the components of the solution backwards (starting from the last one). Computing component i knowing all components of index $j > i$ needs $2q_M$ multiplications, except for the $2q_M$ last rows, that is a total of $2q_M N_M + O(q_M^2)$ multiplications. For the one with matrix Q, we have to apply the inverse $q_M N_M + O(q_M^2)$ Givens rotations to the right-hand-side, each of them needing 4 multiplications. The conclusion follows. ■

Remark 1. (i) Usually m and n_g are of the order of n, and in this case the cost of factorization (resp. resolution) of the Jacobian A is of order s^3n^3N (resp. s^2n^2N).
(ii) It would be interesting to have an idea of the least possible number of operations for an orthogonal factorization of a band matrix. Our procedure seems to be quite effective, but we do not know how far it is from this lower bound.
(iii) The ratio between factorization and solve is, under the assumptions of the lemma, essentially sn. Typical values of sn will be larger than 10. This means that factorization is by far the most expensive part of resolution.

2.3 Linear algebra: problems with design variables

We start our analysis with a general result on the factorization by elimination. Let A be a $n \times n$ invertible matrix, where $n = n_1 + n_2$, n_1 and n_2 being positive integers. Let A_{ij}, $i, j = 1, 2$, be its decomposition by blocks of size $n_i \times n_j$, as on the figure below (corresponding to the case when A_{11} is a band matrix):



Let (x, y) be solution of the following linear system:

$$A_{11}x + A_{12}y = b_1; \quad A_{21}x + A_{22}y = b_2. \tag{6}$$

Assuming A_{11} to be invertible, we may eliminate x from the first equation, and express (x, y) as

$$y = (A_{22} - A_{21}A_{11}^{-1}A_{12})^{-1}(b_2 - A_{21}A_{11}^{-1}b_1); \quad x = A_{11}^{-1}(b_1 - A_{12}y). \tag{7}$$

The *factorization* step consists in factorizing A_{11}, solving n_2 linear systems of type $A_{11}x = c$, where c is an arbitrary column of A_{12}, and computing the *reduced matrix* $A_R := (A_{22} - A_{21}A_{11}^{-1}A_{12})^{-1}$, and finally factorizing A_R. The *resolution step* needs the resolution of two linear systems with matrix A_{11}, and one with matrix A_R. We obtain the following complexity result:

Lemma 2. *The resolution of (6) based on reduction (7) requires* (i) *for factorization: the factorization of A_{11}, n_2 resolutions with matrix A_{11}, and the factorization of A_R, and* (ii) *for each resolution (after factorization): to solve two linear systems with matrix A_{11} and one with matrix A_R.*

We apply this result to the case of a single arc problem with n_s static parameters, the time dependant variables being eliminated first, using the number of operations estimated in lemma 1. Then static parameters are computed using a full matrix factorization. Lemma 2 implies the following result:

Corollary 1. *A single arc problem with n_s static parameters, using the above elimination procedure, and the QR factorization for time dependant variable, needs $O(n_s q^2 N + q^3 N) + O(n_s^3)$ operations for factorization, and $O(Nq^2 + n_s^2)$ for each resolution (after factorization).*

2.4 Mesh Refinement

For a Cauchy problem (integration of an ordinary differential equation with given intial value) the control of the size of time step is done once for all for each given time step. For a two points boundary value problem, the situation is different. Although the error estimate is of the same nature, one needs to solve a given discretization grid accurately enough before obtaining the local error estimates that allow to predict the needed additional grid points. Some refinement schemes are based on primal errors only, see [6, 5]. We show here how to compute an "optimal" refinement for the state and costate equations.

The theory of error estimates tells us that the local error on step k is of the form $e_k = C_k h_k^{p+1}$, where the order p is, in principle, known. The error estimate may be obtained by comparing the local variation of the differential variable with the one computed by a scheme of higher order. Dividing the step h_k into q_k smaller steps of size h_k/q_k, where q_k is a positive integer, we reduce the 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. This is valid, of course, provided the step is small enough. The problem of reaching a specific error estimate by adding the smallest number of new grid points can therefore be formulated as follows:

$$\underset{q\in\mathbb{N}^N}{\text{Min}} \sum_{k=1}^{N} q_k; \quad \sum_{k=1}^{N} \frac{e_k}{q_k^p} \leq E. \tag{INP}$$

This nonlinear integer programming problem appears to be easily solvable, by adding iteratively points at steps for which the local gain, obtain by incrementing q_k by one, is maximal. See algorithm 1 below, and [3] for details.

Algorithm 1 *Primal Algorithm*

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

2.5 Interior-point algorithms

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

$$\begin{cases} \text{Min} \displaystyle\int_0^T \ell(y(t),u(t))\mathrm{d}t + \Phi(y(T)); \\ \dot{y}(t) = f(y(t),u(t)), \quad t \in [0,T]; \\ 0 \leq g(y(t),u(t)), \quad t \in [0,T]; \\ y(0) = y^0. \end{cases} \tag{8}$$

Introducing a nonnegative slack variable for inequality constraints, a using a logarithmic penalty for the nonnegativity constraint on the slack variable,

and setting $\ell_\varepsilon(y,u) := \ell(y,u) - \varepsilon \sum_{i=1}^{q} g_i(y,u)$, where $\varepsilon > 0$, we obtain the following approximation:

$$\begin{cases} \text{Min} \displaystyle\int_0^T \ell_\varepsilon(y(t),u(t))\mathrm{d}t + \Phi(y(T)); \\ \dot{y}(t) = f(y(t),u(t)), \quad t \in [0,T], \\ y(0) = y^0. \end{cases} \tag{9}$$

The penalized problem (9) is unconstrained with an integral and final cost. Adding an additional state variable 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.

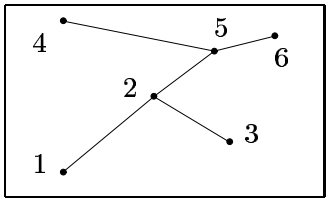
An interesting open problem is to obtain error estimates for given parameter ε and discretization steps. This is obviously difficult, since without logarithmic penalty, one may encounter a reduction order when state constraints are active (it is known that in general a Runge-Kutta scheme applied to an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

Related to this is the question of the choice of a path of convergence, i.e., at which speed should the parameter ε and the discretization steps to 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.

3 Multiarc problems

3.1 Framework

With a multiarc problem is associated a *scenario graph* whose edges are the arcs of the optimal control problem, i.e. time intervals with a given dynamics and integral cost, and vertices are junction points corresponding to starting or ending points, separation, or rendez-vous. For instance, with the mission example stated in the introduction is associated the following scenario graph:



Formally, the graph is a pair (V,E) where V is the finite set of vertices, and $E \subset V \times V$ is the set of edges. With edge $e = (i,j)$, where i and j are vertices, are associated the following variables and data:

$$\begin{array}{ll} y^e(t) \in I\!R^{n_e},\ u^e(t) \in I\!R^{m_e}: & \text{state and control,} \\ \pi^e \in I\!R^{n_{s,e}}: & \text{static optimization parameters, size } n_{s,e}, \\ \ell_e(t, y^e(t), u^e(t), \pi^e): & \text{distributed cost,} \\ f_e(t, y^e(t), u^e(t), \pi^e): & \text{state dynamics,} \\ g_e(t, y^e(t), u^e(t), \pi^e): & \text{distributed constraints,} \end{array} \tag{10}$$

Static optimization parameters are parameters involved at each time step, such as a mass of vehicle or the engine power. Since we reduce the possible variable time (spent on the arc) to a fixed unit time through the usual change of variables, the durations also appear as static parameters in our formulation.

With a vertex $j \in V$ are associated the following variables and data:

$$\begin{array}{ll} z^j: & \text{variables at vertex, size } n_j, \\ \Phi_i(z^j): & \text{cost at vertex,} \\ g_i(t_j, z^j): & \text{distributed constraints.} \end{array} \tag{11}$$

Variables z^j include the date t_i associated with the vertex, a copy of either initial or terminal states of arcs connected to this vertex, and as a copy of all static optimization parameters of these arcs. The multiarc problem may be formulated as follows:

$$\left\{ \begin{array}{l} \text{Min} \displaystyle\sum_{e=(i,j)\in E} \int_{t_i}^{t_j} \ell_e(t, y^e(t), u^e(t))\mathrm{d}t + \sum_{j\in V} \Phi_j(z^j)); \\ \dot{y}^e(t) = f_e(t, y^e(t), u^e(t), \pi^e), \quad t \in [t_i, t_j], \quad \text{for all } e = (i,j) \in E, \\ 0 \le g_e(t, y^e(t), u^e(t), \pi^e), \quad \text{for all } e \in E, \\ 0 \le g_i(t_j, z^j), \quad \text{for all } j \in V, \\ 0 = h_j(t_j, (y^{(.,j)}(t_j), \pi^{(.,j)}), (y^{(j,.)}(t_j), \pi^{(j,.)}), z^j), \quad \text{for all } j \in V. \end{array} \right.$$

The equality constraint at the nodes h_j includes the above mentioned copies of initial or terminal states and staic parameters, as well as equality constraints on initial or terminal states, if any.

3.2 Linear algebra

Here is the main result of this section.

Theorem 2. *Let (V, E) be the scenario graph of the multiarc optimal control problem. Let N_e denote the number of time steps of arc e. Denote by s_e the number of inner steps in the Runge Kutta formula used for arc e, and by $n_{g,e}$ the number of distributed constraints on arc e. Let $n_{s,e}$ and n_v denote the number of static parameters on edge e, and the number of vertices variables on vertex v, respectively.*

Then the band size q_e on arc e satisfies $q_e = s_e O(n_e + m_e + n_{g,e})$, and a procedure (to be detailed below) allows to obtain a complexity for factorization and resolution of the Jacobian of discretized optimality system of order

$$\sum_{e\in E}\left[N_e(q_e^3+n_{s,e}q_e^2)+n_{s,e}^3\right]+\sum_{e\in E}(n_{s,e}^2+q_e^2N_e)(n_{\underline{e}}+n_{\overline{e}})+\sum_{v\in V}n_v^3. \quad (12)$$

Here $\underline{e}$ and $\overline{e}$ denote the starting and ending vertices, respectively, of edge e.

Remark 2. (i) The first sum in (12) represents the order of number of operations needed for the factorization of parts of the Jacobian corresponding to each arc. Therefore the total cost is proportional to the number of vertices and edges (weighted by coefficients depending on the number of associated variables) and the overall complexity (12) seems to be quite low.
(ii) For most real world optimal control problems, we may expect that the dominant term in (12) will be $\sum_{e\in E} q_e^3N_e$, itself of order $\sum_{e\in E} s_e^3n_e^3N_e$.

The elimination procedure is as follows.

Algorithm 3 *Elimination*

1) Factorize the parts of Jacobian corresponding to each arc
2) Eliminate all arc variables
3) Eliminate recursively all node variables, starting from leaves

Proof of theorem 2. That $q_e = s_eO(n_e+m_e+n_{g,e})$ results from the discussion before lemma 1. In view of that lemma, the number of multiplications for step 1 is of order $\sum_{e\in E}[N_e(q_e^3+n_{s,e}q_e^2)+n_{s,e}^3]$. Step 2 needs at most $(n_{\underline{e}}+n_{\overline{e}})$ resolutions of the four blocs matrix and each resolution need $(n_{s,e}^2+q_e^2N_e)$ operations. In step 3, elimination of leaf v connected to vertex w does not interplay with other vertices, and need, by lemma 2, the factorization of matrices of size n_i and n_j, plus solving n_j linear systems of size n_i, which means a number of operations of order $n_i^3+n_j^3+n_jn_i^2$. Since $n_jn_i^2 \le \max(n_i^3,n_j^3)$ this is of order $n_i^3+n_j^3$. The third step being recursive, the conclusion follows. ■

Remark 3. Our implementation of the elimination procedure in C describes matrices as data structures plus a routines allowing factorization, so that sparse matrices of different types may be involved. We have presently three types of matrices: band, full and identity. Only the single arc minimization is implemented presently, but it uses already this general elimination procedure. What lacks for the multiarc problem is the construction of the optimality conditions and its Jacobian, starting from the description of subsection 3.1.

4 Application to Goddard's problem

Goddard's problem, stated in 1919 (see e.g. [20]) consists in maximizing the final altitude of a rocket with vertical ascent. The model involves three state variables: altitude r, speed v and mass m, and the thrust F as control variable.

The objective is to maximize the final altitude, the final time being free. The Dynamics is

$$\begin{cases} \dot{r}(t) = v(t), \\ \dot{v}(t) = (F(t) - D(v(t), r(t)))/m(t) - g(r(t)), \\ \dot{m}(t) = -F(t)/c \end{cases} \tag{13}$$

where D is the drag function, with arguments speed and altitude, $g(r)$ is the gravity field, and c 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. There exists a nice piece of theory that allows to obtain the control law during the singular arc (see [10, 20]), but of course we do not use it in our numerical experiments.

Although our code may use arbitrary Runge-Kutta coefficients, we have used only the Gauss-Legendre of order 2, so that we may see how the code bevahes when there are many time steps. The results are synthetized in Table 1, 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 size N of the mesh, the numer 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. The number of inner iterations remains small. It is fair to say, however, that our reduction procedure for ε is quite conservative (division by 2). Stronger reductions will be the subject of future research.

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 arcs structure, one of them being singular.

The density of mesh, after each major iteration is displayed in figure 5. 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.

5 Conclusion

The main features of our approach are the use of dedicated algebra solvers that exploit the band structure of sparse matrices, and the optimal refinement

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. Results

scheme. The interior-point methodology avoids the need for large number of iterations as in sequential quadratic programming algorithms [5, 7], but it also gives flexibility for refinement that can be performed at any iteration. Of course shooting methods are always less expensive for a given precision, but they are not always stable and frequently need a priori knowledge of the sequence of active constraints.

We apply our methodology to reentry problems in [1] (that paper contains also a detailed analysis of the refinement problem). Numerical computations for multiarc problems will be presented in J. Laurent-Varin's Ph.D. Thesis.

Much remains to be done in several directions. (i) In the case of state constraints the is a loss in error orders that should be taken into account in the analysis. (ii) We do not have any theory telling what should be the order of parameter ε for a given value of discretization errors (i.e., along which path these two parameters should go to zero). (iii) Convergence could be improved (especially for avoiding stationary points that are not local minima) by using step decomposition, see [9, Part III].

Finally heuristics for having good starting points are of course essential and are a research subject by themselves.

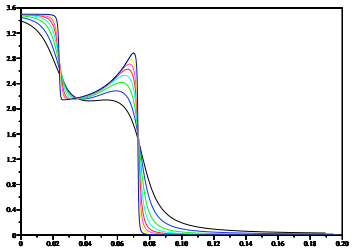


Fig. 1. Thrust law

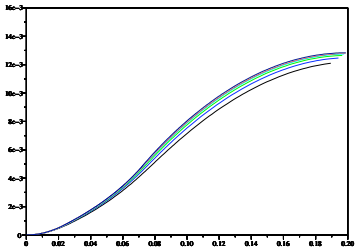


Fig. 2. Altitude

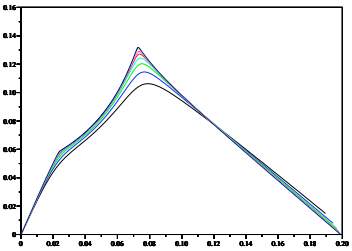


Fig. 3. Velocity

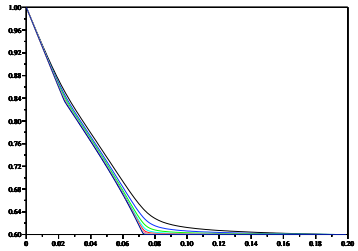


Fig. 4. Mass

References

1. Bérend, N., Bonnans, F., Haddou, M., Laurent-Varin, J., Talbot, C.: An Interior-Point Approach to Trajectory Optimization. INRIA Research Report RR-5613, www.inria.fr/rrrt/rr-5613.html (2005)
2. Bérend, N., Bonnans, F., Haddou, M., Laurent-Varin, J., Talbot, C.: A Preliminary Interior Point Algorithm For Solving Optimal Control Problems, 5^{th} International Conference on Launcher Technology. Madrid, Spain (2003)
3. Bérend, N., Bonnans, F., Haddou, M., Laurent-Varin, J., Talbot, C. On the refinement of discreization for optimal control problems. 16^{th} IFAC SYMPOSIUM Automatic Control in Aerospace. St. Petersburg, Russia (2004)
4. Betts, J.T.: Survey of Numerical Methods for Trajectory Optimization. AIAA J. of Guidance, Control and Dynamics, **21**, 193–207 (1998)
5. Betts, J.T.: Practical methods for optimal control using nonlinear programming. Society for Industrial and Applied Mathematics (SIAM), Philadelphia (2001)
6. Betts, J.T., Huffman, W.P.: Mesh refinement in direct transcription methods for optimal control. Optimal Control Applications & Methods, **19**, 1–21 (1998)
7. Bonnans, J.F., Launay, G.: Large Scale Direct Optimal Control Applied to a Re-Entry Problem. AIAA J. of Guidance, Control and Dynamics, **21**, 996–1000 (1998)

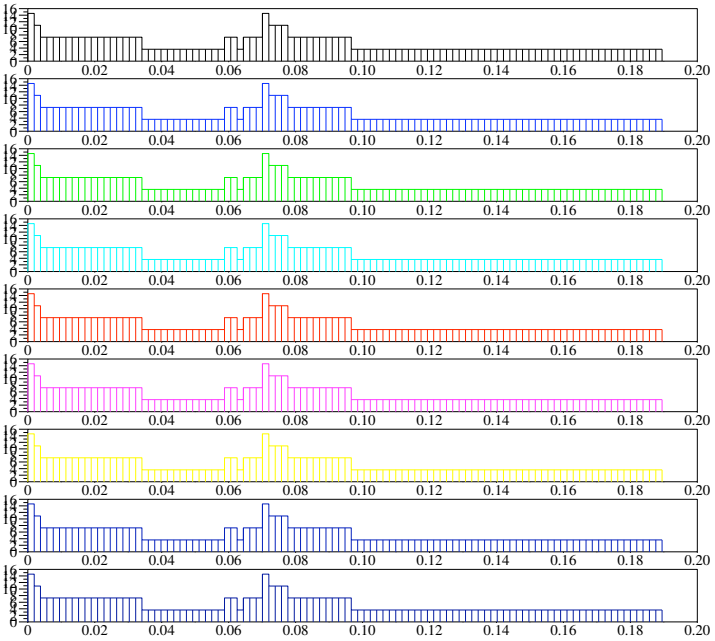


Fig. 5. Mesh

8. Bonnans, J.F., Laurent-Varin, J.: Computation of order conditions for symplectic partitioned Runge-Kutta schemes with application to optimal control. INRIA Research report RR-5398, www.inria.fr/rrrt/rr-5398.html (2004)
9. Bonnans, J.F., Gilbert, J.Ch., Lemaréchal, C., Sagastizábal, C.: Numerical Optimization: theoretical and numerical aspects. Springer-Verlag, Berlin (2003)
10. Bryson, A. E., Ho., Y.-C.: Applied optimal control. Hemisphere Publishing, New-York (1975)
11. Bulirsch, R., Nerz, E., Pesch, H. J., von Stryk, O.: Combining direct and indirect methods in optimal control: range maximization of a hang glider. In "Optimal control", Birkhäuser, Basel, 273–288 (1993)
12. Dontchev, A. L., Hager, W. W.: The Euler approximation in state constrained optimal control, Mathematics of Computation, **70**, 173–203 (2001)
13. Dontchev, A. L., Hager, W. W., Veliov, V. M.: Second-order Runge-Kutta approximations in control constrained optimal control. SIAM Journal on Numerical Analysis, **38**, 202–226 (2000)
14. Hager, W.: Runge-Kutta methods in optimal control and the transformed adjoint system. Numerische Mathematik, **87**, 247–282 (2000)
15. Hairer, E., Lubich, C., Wanner, G.: Geometric numerical integration. Springer-Verlag, Berlin (2002)
16. Hairer, E., Nørsett, S. P., Wanner, G.: Solving ordinary differential equations I. Springer-Verlag, Berlin (1993)

17. Hairer, E., Wanner, G.: Solving ordinary differential equations II. Springer-Verlag, Berlin (1996)
18. Pesch, H. J.: A practical guide to the solution of real-life optimal control problems. Control and Cybernetics, **23**, 7–60 (1994)
19. von Stryk, O., Bulirsch, R.: Direct and indirect methods for trajectory optimization. Annals of Operations Research, **37**, 357–373 (1992)
20. Tsiotras, P., Kelley, H.J.: Drag-law effects in the Goddard problem. Automatica, **27**, 481–490 (1991)