Mathematical Programming manuscript No.
(will be inserted by the editor)

François Courty · Alain Dervieux

A SQP-like one shot algorithm for Optimal Control problems

Received: date / Revised version: date

Abstract. We propose an extension of the SQP algorithm to an optimization algorithm dealing with equality constraints under the form of large scale state equations. The specificity of this algorithm is that it relies on an iteration which does not need a complete linear system resolution. Instead, it uses a partial or progressive resolution of the state and costate that is much more efficient. As in Trust-Region SQP, the Wolfe condition and the constraint satisfaction are controlled by the algorithm. An application to a large scale optimal shape design is presented.

Key words. Sequential Quadratic Programming – trust regions – optimal design – state equation

1. Introduction

The tools applying to smooth optimization have significantly improved over the last two decades, replacing gradient based methods with Sequential Quadratic ones.

François Courty: INRIA, BP 93, 06902 Sophia Antipolis, France e-mail: `francois.courty@sophia.inria.fr`

Alain Dervieux: INRIA, BP 93, 06902 Sophia Antipolis, France

Mathematics Subject Classification (1991): 20E28, 20G40, 20C20

The initial target of SQP methods was the minimization of a functional under a set of equality constraints. A particular case arises when the constraints can be used to eliminate a part of the unknowns. This is the case when an Optimal Control problem involving a state equation is addressed. SQP methods are designed to solve the three Karush-Kuhn-Tucker (KKT) conditions, namely the state equation, the adjoint equation and the stationarity condition. At the same time, they keep the idea of decreasing some functional close to the original one. Two steps are applied in a typical SQP algorithm. A restoration step yields a new value of the state variables satisfying better the constraints. A minimization step updates the control in order to decrease the criterion to minimize. The two steps are of the Newton type solved using an exact Hessian, or more frequently a quasi-Newton algorithm made possible, for example, by the BFGS formula [16, 3].

Newton methods are subject to attraction region conditions and SQP algorithms became significantly more robust with the addition of methods such as the Trust-Region controlling the convergence.

A representative and popular algorithm combining these methods is the Byrd-Omojokun algorithm [16,3]. It involves the above steps and a Trust Region solving progressively the state equation at the same time as the functional stationarity equation.

The sequel of this introduction concentrates on this algorithm, in order to show how it can be transformed to better apply to a particular class of problems. The disadvantages that we shall point out also exist in other SQP algorithms.

One advantage of SQP is that the state equation is not necessary solved at each iteration. Instead, it can be advanced progressively towards a solution which will satisfy the state equation only at full convergence of the SQP algorithm. This may be important if the solution of state equation is very expensive and would render each SQP iteration so costly that only a few iterations would be possible (this is the case when the previous generation of algorithms - gradient algorithms - are used). However, SQP efficiency requirements more or less assume that a Newton step for the state equation, i.e. an adhoc linear system solution, that can be solved at low cost. Further, it seems paramount in the mechanism (and convergence proof) of the Byrd-Omojokun (and most other SQP) algorithms that the *adjoint state* equation be completely solved at each SQP iteration. Last but not least, a third linearized state equation is also solved in the minimization phase. In summary, a standard SQP method will solve the equivalent of three linearized systems derived from state equation at each main iteration. This approach is not well suited to the case of large scale state systems as those arising in many engineering applications of optimal design, and, typically in aerodynamics. Let us recall some reasons for this:

- Newton methods for solving stiff nonlinear state equations are not the most robust and efficient algorithms. For the particular case of aerodynamics, pseudo-time solvers are more satisfactory, and already exist in many codes.

- Linearized systems obtained by exact differentiation are not always easy to solve. For compressible flows, they are generally not solved (but replaced by simpler ones), and when they are solved, the matrix conditioning is generally improved by a mass matrix multiply (from pseudo-time iteration). We can however imagine using a simplified Jacobian matrix inside the SQP restoration. Solving the system accurately may remain expensive in cpu time (it has often to be solved with a matrix-free iteration). Would the simplified system be accurate enough for replacing the linear system solved in the minimization phase?

- The adjoint system can be even more complex than the state equation, particularly when matrices cannot be stored, and then equation assembly is dominant in solution complexity. A linearized PDE often can have more terms than the nonlinear original PDE. This can be due to the complex nonlinear devices introduced by sophisticated approximations (TVD schemes and Riemann solvers in Computational Fluid Dynamics). The situation can be even worse when the derivation is not optimally programmed in the software, as happens frequently if an Automatic Differentiation tool is used.

These remarks tend to show that in many large scale applications, solving the three above linear systems is as complex as solving the state equation itself.

The impact of this on the global complexity of the algorithm can be evaluated by the following model: let us assume that the complexity of solving the linear systems is N and that the number of SQP iterations is approximatively equal to the number of control parameters, let us say n. Then the complexity is $N \times n$, i.e. quadratic. When N is large, the overall complexity is too large when n is not small, and it is compulsory to replace the existing approach by an algorithm that solves simultaneously or progresssively the set of KKT equations.

In order to avoid quadratic complexity, Shlomo Ta'asan introduced this type of algorithm under the term of “one shot” in a pioneering work [18]. An algorithm with a similar behaviour was introduced in [2]. Many authors dealing with particular applications proposed similar ideas. See for example [11,8].

These works demonstrated important efficiency improvements but need to be revisited from the standpoint of optimization theory. Indeed, in practice, these methods are not always sufficiently robust. They converge only after some complex trial and error for tuning some parameters. They should involve some robustness heuristics, and even some superlinear convergence mechanism. For those improvements, most of the missing ingredients exist in today's optimization theory and methods. The purpose of this work is to introduce a new generation of algorithms, not belonging to the SQP class, but similar, which will incorporate some of the modern properties of SQP while retaining the benefits of the one shot formulation.

In order to achieve this, we first consider the Byrd-Omojokun SQP algorithm. This version will be useful for two purposes:

- firstly for efficiency and robustness comparisons with the new proposed algorithm,
- secondly for presenting the new algorithm as resulting from a relatively minor modification of the Byrd-Omojokun SQP algorithm.

The basic assumption in the proposed method is that we have at our disposal a **progressive** iterative solver for the state and co-state. By progressive, we mean in particular that one step of this solver is much less costly than solving the linearized system. Such solvers have been designed and are routinely used in many applications, and particularly in Aerodynamics.

The basic principle for controlling the convergence is, like in SQP algorithm, the control of the decreasing of a functional. The second principle is to consider the adjoint equation as a constraint in the same way as the state equation. Both are advanced in a preliminary restoration phase which, in contrast to usual SQP, cannot be invalidated, and therefore progression in the resolution of these two systems is a natural consequence. This results in a Wolfe-type condition guaranteeing the quality of the descent direction and of the step length.

The efficiency of the new algorithm will be checked and compared with the efficiency of the Byrd-Omojokun algorithm. A first example will be a model problem inspired by both optimal control and least square identification. A second example will rely on one of today's challenge in Computational Fluid Dynamics, the solution of an optimal shape problem.

The plan is the following:

- Section 2 : Description of main features of a particular version of the Byrd-Omojokun algorithm,

- Section 3 : one shot algorithm,

- Section 4 : Application to a model problem.

- Section 5 : Application to sonic boom minimization.

- Section 6 : Conclusions and future directions.

2. Description of main features of a particular version of the Byrd-Omojokun algorithm

The algorithm proposed by Byrd [4] and Omojokun[17] can be classed as a SQP method with a trust region [3]. This algorithm is in the same line as [19], [6], [10], and [5].

This algorithm is in the sequel of those proposed by Vardi [19], Celis, Dennis and Tapia [6], Gilbert [10], Byrd, Gilbert and Nocedal [5].

2.1. Main principles

We shall use the version proposed in [14] rewritten in the so-called **Reduced-Hessian** version. We consider the minimization problem:

$$\min \quad J(u, Y) \tag{1}$$

$$\text{subject to } \Psi(u, Y) = 0 \ .$$

The minimum is taken with respect to the composite variable x:

$$x = (u, Y) \ .$$

We assume an equality contraint written as:

$$\Psi(u, Y) = 0$$

We also assume that the Jacobian

$$A = \frac{\partial \Psi}{\partial Y} \tag{2}$$

is always invertible.

Starting from the iterate $x_k = (u_k, Y_k)$, we define $\Psi_k = \Psi(x_k)$ and $A_k = \dfrac{\partial \Psi}{\partial Y}(x_k)$.

The Lagrange multiplier Π_k is a solution of

$$\Pi_k = \left(\frac{\partial \Psi}{\partial Y}(x_k)\right)^{-T} \frac{\partial J}{\partial Y}(x_k) \ . \tag{3}$$

We denote by $g_k = g(x_k, \Pi_k)$ with:

$$g(x_k, \Pi_k) = \frac{\partial J}{\partial u}(x_k) - < \Pi_k \ , \ \frac{\partial \Psi}{\partial u}(x_k) > \tag{4}$$

2.1.1. State restoration phase Given a trust region Δ_k and a relaxation parameter $\xi \in (0,1)$, we compute a step v_k which solves the vertical or normal subproblem:

$$\min_{v \in \mathbb{R}^n} \ \|A_k v + \Psi_k\| \tag{5}$$

$$\text{under condition} \quad \|v\| \leq \xi \Delta_k$$

This problem is generally solved with the help of a Newton step ($v^n = -A_k^{-1}\Psi_k$) corrected in order to satisfy the trust region constraint. This correction can be done with a dogleg method, see [16,3]. The combination of Trust Region and dogleg methods ia a way for controlling the convergence of the Newton iteration for the state equation. The linear vertical problem may have many solutions, but Lalee, Nocedal and Plantenga have shown in [14] that it has always a unique solution in the image space of A_k. This allows a complete decoupling between this subproblem and the second subproblem.

2.1.2. Minimization phase We first compute the step δu

$$\min_{\delta u} \; {g_k}^T \delta u + \frac{1}{2} \delta u^T W_k \delta u \tag{6}$$
$$\textit{under the contraint} \quad \|\delta u\| \leq \sqrt{\Delta_k^2 - \|v_k\|^2}.$$

where W_K is a BFGS-approximation (see [16,3]) of the reduced hessian computed from the successive g_k and δu_k.

Introducing

$$h_k \;=\; (-\; A_k^{-1} \frac{\partial \Psi}{\partial u}(x_k)\delta u \;,\; \delta u) \tag{7}$$

the final step is defined by

$$d_k \;=\; (v_k, 0) \;+\; h_k \tag{8}$$

2.2. Byrd-Omojokun algorithm overview

The complete algorithm writes as follows

Initialization of constants $\varepsilon > 0$ and $\eta \in (0,1)$

Initialization of x_0 and $\delta_0 > 0$.

For k=0,...

Compute $J_k, \Psi_k, A_k, \Pi_k, g_k$ from (2),(3) and (4)

If $\|g_k\|_\infty < \varepsilon$ **et** $\|\Psi_k\|_\infty < \varepsilon$ **Then stop** (Problem is solved).

Restoration phase

Compute v_k by solving the vertical subproblem (5)

Minimization phase

Compute W_k from BFGS approximation.

Compute h_k by solving the horizontal subproblem (6).

Let $d_k = v_k \ + \ h_k$.

Update Δ_k with trust region heuristics.

End of loop : next k.

2.3. Trust region heuristics

The Trust Region method is explained in details in ([16], 18.9) and [3]. We recall here some useful informations for understanding the present work. The simultaneous decreasing of state residual and objective functional is controlled with the merit function of Han: $\Phi(x) = J(x) + \mu\|\Psi(x)\|_2$.

The algorithm writes:

Compute parameter μ^+ (See [14]).

Compute the **actual reduction**:

$$\text{a\_red} \ = \ J(x_k) - J(x_k + d_k) + \mu^+(\|\Psi(x_k)\| - \|\Psi(x_k + d_k)\|). \tag{9}$$

Compute the **predicted reduction**

$$\text{p\_red} \ = \ -d_k^T g_k - \frac{1}{2} d_k^T W_k d_k + \mu(\|\Psi(x_k)\| - \|A_k^T v_k + \Psi(x_k)\|). \tag{10}$$

If $\frac{\text{a\_red}}{\text{p\_red}} \geq \eta$

Then $x_{k+1} = x_k + d_k$, $\mu_{k+1} = \mu^+$, and enlarge the trust radius.

Else $x_{k+1} = x_k$, $\mu_{k+1} = \mu_k$, and reduce the trust radius.

End of loop : next k.

2.4. Some comments

The minimisation phase involves a correction of the state variable computed with the third linear problem to solve (7). This correction uses a first-order Taylor formula that, for small steps, ensures that the state equation residual does not increase much.

The control for the algorithm relies on two main parameters, the Trust Region radius, and the penalty coefficient.

We denote by $\Psi(\varepsilon) = \Phi(x + \varepsilon.d)$.

The Trust Region radius ε controls the local correspondance between the quadratic model and the fonctional to minimize. More precisely, this accordance is equivalent to the classical *Wolfe condition*: $(0 < \omega_1 < \omega_2 < 1)$

$$\begin{cases} \Psi(\varepsilon) \leq \Psi(0) + \omega_1 \varepsilon \Psi'(0) \\ \Psi'(\varepsilon) \geq \omega_2 \Psi'(0). \end{cases}$$

To satisfy it, it is sufficient to reduce the radius ε since the usual gradient satisfies (11).

The penalty parameter μ^+, via the merit function, controls the simultaneous convergence of functional (to minimum) and of constraints (to zero).

3. One-Shot-Trust-Region algorithm

3.1. Main assumption: pseudo-Newton iteration

The restoration step (5) for the state equation will be replaced by a *pseudo-Newton* step:

$$v = -S\ \Psi(x_k)$$

relying on a particular operator S that is reasonably inexpensive to evaluate and that approximates the inverse of the Jacobian $\dfrac{\partial \Psi}{\partial Y}$. Typically, it can be the result of a few Gauss-Seidel sweeps or diagonally preconditioned Conjugate Gradient iterations. This simplified solution will be applied in place of both the restoration phase and the computation of h_k in the minimization step (7).

We will do this in a similar way for the adjoint equation. In the proposed algorithm, the adjoint equation is considered as an additional equality constraint. Instead of solving

$$\Pi = (\frac{\partial \Psi}{\partial Y})^{-T} \frac{\partial J}{\partial Y},$$

its restoration will consist in solving the simpler problem:

$$p = S^T \left((\frac{\partial \Psi}{\partial Y})^T \Pi - \frac{\partial J}{\partial Y} \right).$$

It is then natural to require the new algorithm to restore approximatively both the state constraint and the adjoint constraint in two restoration phases before doing the minimization step. By comparison with a classical SQP algorithm, applying only the incomplete solutions will decrease considerably the cost of one global iteration.

The main assumptions concerning the above pseudo-Newton iteration should involve:

- convergence properties of the state iteration,
- convergence properties of the adjoint iteration,
- their complexity,
- restoration of the descent property.

This leads us to introduce the following assumptions:

- **Assumption 1 :** For $u \in I\!R^{N_u}$ and for $p \in I\!R^{N_Y}$

$$\|\Psi(Y - S\Psi(Y,u),u)\| \ \le \ 0.99\ \|\Psi(Y,u)\|$$

and

$$\|\frac{\partial \Psi}{\partial Y}^* (\Pi + p) - \frac{\partial J}{\partial Y}\| \ \le \ 0.99\ \|\frac{\partial \Psi}{\partial Y}^* \Pi - \frac{\partial J}{\partial Y}\|$$

- **Assumption 2 :**

Both quasi-Newton steps are as complex as a few equations assemblies.

- **Assumption 3 :** $\forall\ \ u_0$, after k_1 partial resolution steps, $g(Y_k, u_0, \Pi_k)$ is close enough to the gradient to satisfy a "good" descent direction:

$$\exists\ k_1, \exists\ \rho_{max}\ /\ \forall\ \rho,\ \ 0 < \ \rho\ \le\ \rho_{max},$$

$$\forall\ k \ge k_1\ \ ,\ \ J(Y_{k+k'}, u_0 - \rho W_k^{-1} g(Y_k, u_0, \Pi_k)) - J(Y_k, u_0)\ \le\ -0.95\rho\,\|j'(u_0)\|^2$$

where $\quad \Pi = (\frac{\partial \Psi}{\partial Y})^{-T} \frac{\partial J}{\partial Y}$ and $j'(u) \;=\; \frac{\partial J}{\partial u}(x) \;-\; < \Pi \;,\; \frac{\partial \Psi}{\partial u}(x) >.$

Assumption 1 is a convergence assumption concerning the pseudo-Newton iteration. **Assumption 2** is a natural consequence of the smoothness of the state equation and the cost function. The values of the positive constants .99 and .95 is not important as long as they are smaller than unity. For **Assumption 3** we observe that for $k - - > +\infty$, Y_k converges towards the solution of the state equation, Π_k converges towards the solution of the adjoint state equation and then g is the gradient of the functional j, $j : u \;\rightarrow\; J(Y(u), u)$.

Assumption 3 is a smoothness assumption and is a consequence of assumption 1 if the functional J and the operator Y are sufficiently smooth.

3.2. Post-update of the state variable

The minimization phase of the reduced-Hessian BO algorithm of Section 2 requires in (7) a complete solution of the linearized state system. As noted below, this linearised state update insures that the SQP minimization step maintains the residual of the state equation at the previous level attained in the restoration step.

In the new algorithm, a good update of the state variable will also be necessary, in particular in order to be able to search (by a *linear search*) a correction which satisfies the Wolfe condition.

We cannot, due to efficiency reasons, apply a linearized resolution for updating the state. Instead, we shall use the quasi-Newton operator S. But this operator would not allow a good evaluation of δY. We shall then avoid seeking for a good approximation of the correction δY. Instead, we choose to compute directly a new state variable by applying a “post-restoration phase” similar to the previous restoration phase.

The state equation is thus solved iteratively with the help of the operator S. As in SQP, a certain level of convergence must be satisfied. We introduce as iteration test the condition that the final residual of post-restoration is not smaller than the one obtained at the end of the first restoration phase. This criterion is applied to each restoration in the line search (see 3.4, Minimization phase, step (1)).

Finally, in order to have an equilibrated iterative convergence of both state and co-state to their limits, we also update the adjoint state. Two possibilities exist for evaluating δY and $\delta \Pi$, according to choosing either to update the state variable for computing the adjoint one or not. We choose the first option that is *a priori* more stable and converges faster.

3.3. Main algorithm branchings

In transforming a SQP algorithm into the one-shot extension, the management of contraint has to be deeply reconsidered.

- In SQP, a step can be refused (or reduced) by the Trust Region heuristics (essentially because the Wolfe condition is not sufficiently satisfied. In that case, the state variable is reset to its previous value, and CPU effort spent in restoring it is lost).

- In one-shot, the CPU effort for advancing the state variables should not be wasted. When state iteration is not sufficient for satisfying the Wolfe condition, we propose to suppress the minimization update, but not the pre-restoration, which, instead, is re-iterated repeatedly until a minimisation step satisfying the Wolfe condition is obtained.

We also need to control the level of resolution of the state system in the minimization phase. In SQP, the correction does not increase the state equation residual as long as the global minimisation step is small, due to the third system being solved. This property is controlled by a penalty mechanism. Since, in our new algorithm, the third linear system is replaced by progressive updates, we replace the penalty heuristics by a direct control of the residual norm requiring the residual norm, in the minimization step, including line search, to be always lower than the one obtained in the pre-restoration phase.

3.4. Global view of the algorithm

Initialisation

Initialize x_0, Δ_0 and μ_0. $(nstep\ =\ 1)$

For k=0,...

Let $\|G_k\|_\infty \ = \ Max(\|\frac{\partial J}{\partial Y} - \left(\frac{\partial \Psi}{\partial Y}\right)^T \Pi\| \ , \ \|\frac{\partial J}{\partial u} - \left(\frac{\partial \Psi}{\partial u}\right)^T \Pi\| \ , \ \|\Psi(x)\|)$

If $\|G_k\|_\infty \ < \varepsilon$ **then stop.**

Compute the cost function J.

- **State restoration phase .**

Compute a Pseudo-Newton step $v_Y \;=\; -\,S(Y,u)\Psi(Y,u)$

Update the state Y: $\;Y^+ \;=\; Y \;+\; v_Y$

- **Adjoint restoration phase .**

Compute a Pseudo-Newton step $v_\Pi \;=\; -S(Y^+,u)^T \left(\left(\dfrac{\partial \Psi}{\partial Y}(Y^+,u) \right)^T \Pi - \dfrac{\partial J}{\partial Y}(Y^+,u) \right).$

Update the adjoint Π: $\;\Pi^+ \;=\; \Pi \;+\; v_\Pi$

Let $u^+ \;=\; u$.

Compute the gradient $g^+ \equiv g(x^+,\Pi^+) = g(Y^+,u^+,\Pi^+)$

Apply a BFGS or Conjugate Gradient acceleration

Minimization phase

(1) Compute the step δu using a line search and solving the state equation at same level as pre-restoration

(2) If the Wolfe condition is satisfied,

update the control $u' \;=\; u \;+\; \delta u$,

update the state Y: $\;Y' \;=\; Y^+ \;+\; \delta Y$

otherwise reduce the linear search step

(4) If the Wolfe condition not satisfied,

minimisation step is invalidated, go to the beginning of first restoration,

- **Adjoint post restoration phase (without Trust Region).**

$$\delta\Pi \;=\; -S(Y',u')^T \left(\left(\frac{\partial \Psi}{\partial Y}(Y',u') \right)^T \Pi - \frac{\partial J}{\partial Y}(Y',u') \right). \qquad (nstep)$$

Update the adjoint Π: $\Pi^{+} = \Pi + v_{\Pi}$

3.5. Complexity of the one shot algorithm

The cost of the new algorithm is much lower since at each iteration we have just to assemble the equations and to apply the pseudo-Newton steps, which is of similar complexity (Assumption 2). We estimate that the new algorithm is able to reach an optimal complexity of $N_u + 2N_Y$, or in other words, that the solution is reached with an effort of k times a single state system resolution, k being independent from the number of unknowns n.

3.6. Some properties of the algorithm

We have not a proof of the convergence of the algorithm. However some simple properties can be stated. Let us assume that on a particular example, this algorithm has a fixed point, that is

$$u_n \to u$$

$$Y_n \to Y$$

$$\Pi_n \to \Pi$$

then the following properties hold:

- the fixed point is a point where state and adjoint are completely solved since the control is tending to a fixed value and thus the state and adjoint are necessarily progressively solved in the proposed iteration,

– it is not a point where the gradient is not a descent direction. Otherwise, the minimization step would not be valid and convergence of the state and adjoint would be pushed further, then by assumption (2) the gradient would become a descent direction.

– it is not a point where the step length of the linear search tends to zero, since the pseudo-gradient satisfies the Wolfe condition.

Then the only possible scenario is that Y_n and Π_n converge to the state and adjoint solutions and that u_n converges towards a solution of $j' = 0$.

Another evident property of the algorithm is that (u_n, Y_n, Π_n) is a sequence that makes the functional $J(u_n, Y_n, \Pi_n)$ decrease.

4. Application to a model problem

4.1. Theoretical problem

Let us consider a quadratic problem of distributed optimal control.

We denote by Ω the unit interval, $\Omega = [0, 1]$.

We denote by $J(Y, u)$ the cost function

$$J(Y) = \frac{1}{2} \int_{\Omega} (Y - Y_d)^2 dx \ . \tag{11}$$

We want to solve the associated *minimization problem*

$$\min_{v \in \mathcal{U}} \quad \frac{1}{2} \int_{\Omega} (Y(v) - Y_d)^2 dx \tag{12}$$

where $Y(v)$ is the solution of the following problem:

$$\begin{cases} -\Delta Y(v) & = & f \ + \ v \\ Y(v) & \in & H_0^1(\Omega) \end{cases} \tag{13}$$

We consider a finite element discretisation of the continuous problem (13). For the sake of simplicity, we keep the same notations. We denote by A the stiffness matrix of the state system

$$AY \ = \ f \ + \ u \tag{14}$$

The adjoint system writes

$$A^* \Pi \ = \ Y(u) \ - \ Y_d \tag{15}$$

The gradient is

$$g(Y, u) = \Pi(u) \tag{16}$$

4.2. Coarse calculation

We first apply the Byrd-Omojokun algorithm to a discretization with 50 control parameters and 50 state variables. The linear state equation is converged twice at each iteration, with at each time a final residual less than 10^{-12}. This is obtained in about $30 - 50$ conjugate gradient inner sweeps. We present in Fig.1 the history of the convergence of the optimality condition, together with the residuals of the two state and costate equations. Both state residuals stay small at each iteration. With this small number of unknown, the convergence of the Byrd-Omojokun algorithm is rather fast. It is obtained in 50 optimization iterations. Further, the superlinear behavior appears after 30 iterations, as can be expected with this kind of Quasi-Newton algorithm for a symmetric problem of 50 unknowns.

We turn now to the new algorithm. The BFGS option is allowed by the relatively small number of parameters. State and costate equations are solved with a pseudo-Newton algorithm consisting in exactly 3 conjugate gradient sweeps. The residuals of states start with rather large values and then decrease in a similar war as the optimality equation. Each optimization iteration is three or four times less expensive. The convergence is rather surprisingly fast, in 55 iterations. The superlinear convergence also appears in about 35 iterations.

4.3. Finer calculation

We consider now the case of a larger problem with 500 control unknowns (and 500 state variables). At each state resolution, hundreds sweeps are necessary to drive state residual to machine error. The application of the Byrd-Omojokun algorithm is again very fast, with a convergence in 70 optimization iterations. Due to the large number of unknowns, the superlinear convergence is lost.

The new algorithm is again applied with the BFGS option. The pseudo-Newton iteration consists of 3 CG sweeps. In spite of this, the convergence is fast, 85 iterations. The peaks in the state and costate residuals correspond to the reiteration of the pre-restorations, showing that 3 sweeps is really to small.

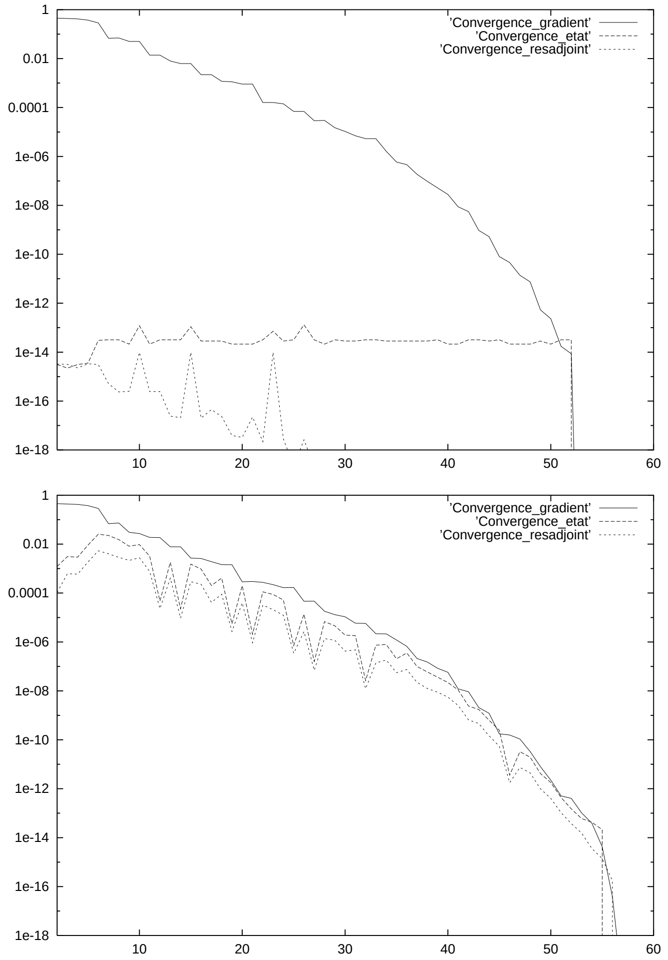

Fig. 1. Model optimal control problem (coarse mesh). Top: Byrd-Omojokun convergence, bottom: one shot convergence

But the computation is robust and self adaptive, while keeping a fair factor in efficiency, with a gain with respect to the BO algorithm that is better than one order of magnitude. The above comparisons are shown in Table 1.

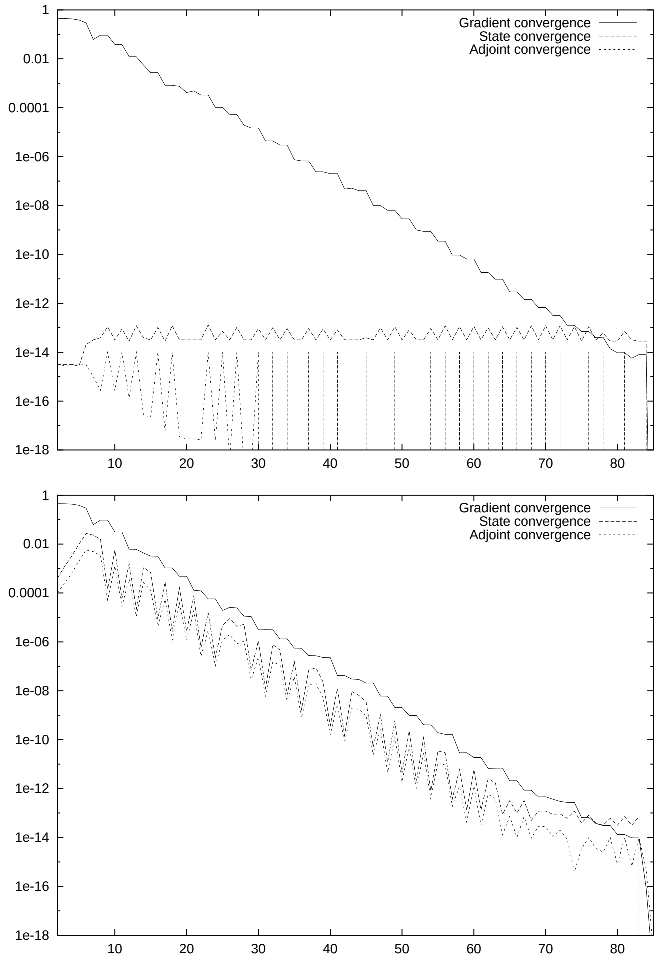

Fig. 2. Model optimal control problem (fine mesh). Top: Byrd-Omojokun convergence, bottom: one shot convergence

Table 1. Efficiency of the different computations measured as the total number of conjugate gradient iterations for a decrease of 10^{-14} of all residuals.

Case	Opt. iterations	Conj. Grad. iterations
Byrd-Omojokun, N=50	50	2000
One shot, N=50	55	660
Byrd-Omojokun, N=500	80	16000
One shot, N=500	85	1020

5. Application to sonic boom minimization

Shape optimization is a very computer intensive application. Two decades ago, only inverse design approaches were possible. They were progressively replaced by control theory based optimisation after the pioneering works of Jameson (see [12,13]). Pseudo unsteady shape optimization algorithms emulating the inverse approach with efficiency comparable to one-shot methods were proposed in [1].

5.1. The optimal shape design problem

For demonstration, we have applied both algorithms to a demonstration problem for optimisation of optimal aerodynamical shapes described in [20,9]. This problem involves the computation of a compressible supersonic flow around a parametrized geometry. A functional is also introduced for evaluating aerodynamical performances and sonic boom strength. The parameters are the positions of all the mesh vertices lying on the shape. Since these parameters are numerous, the modern approach of adjoint is applied. The adjoint software has been built from state routines with the help of reverse Automated Differentiation of assembly loop, we refer to [7,15]. The Euler state equation is solved with an implicit pseudo-time advancing with at each time step the evaluation of first-order Jacobian matrix and the partial resolution of a linear system with a few tens Jacobi sweeps. In the experiment presented here, we use the first-order accurate option for which state and adjoint resolutions are much less cpu consuming (by a factor 4-5). In the initial software, a conjugate gradient was used for optimization. We concentrate on the case of an optimization starting from a

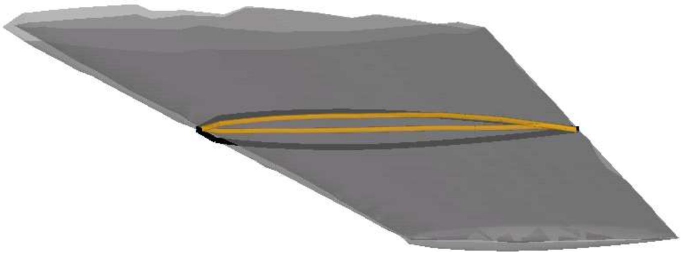

Fig. 3. ONERA M6 optimisation. Initial (black) and optimized (grey) shapes.

ONERA-M6 wing, and with a coarse geometry of 780 shape parameters ($11,000$ flow variables). For this case the cost functional was decreased from 0.06 to less than 0.01. The optimised shape is quite different from the initial one, see Fig.3, and the reduction of the resulting bow shock is demonstrated by a cut of the pressure field below the wing, see Fig.4.

The protocole for our experiments is to keep exactly the specification of test case, while replacing the initial conjugate gradient optimiser by (1) the version of Byrd-Omojokun SQP algorithm described in this paper, and (2) the one-shot extension proposed by this paper.

5.2. Coarse mesh results

We present a first test with the coarse grid of 780 parameters. More realistic experiments would lead to much larger number of parameters and would render the use of an usual BFGS not possible for memory storage reasons. So we have equipped the SQP algorithms with a nonlinear conjugate gradient of Polak-Ribière type, as was used in the original non-SQP gradient loop of [20].

However, with one of our examples, we could try the BFGS option. Unfortu-

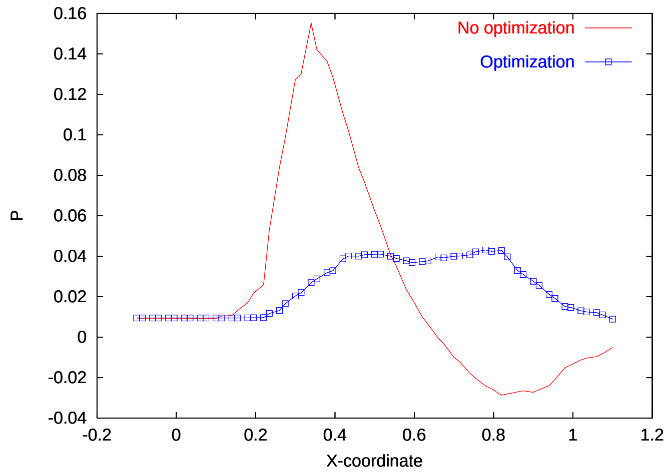

Fig. 4. ONERA M6 optimisation. Pressure along a line below the wing.

nately, this option proved to be less efficient than the old conjugate gradient. We have identified the possible explanation: the so-called "curvature condition", is implemented as a necessary condition before allowing successive gradients to contribute to the approximate BFGS Hessian computation; unfortunately, for this test case, this condition was very rarely satisfyied in the first hundred iterations.

The results we present are obtained with the nonlinear conjugate gradient model inside the SQP algorithm.

The application of the Byrd-Omojokun SQP with complete solution (down to round-off error) of state and adjoint is a mess from the CPU standpoint. Total CPU for 200 iterations is 14 hours of a 1GHz workstation. One state solution consists of about 40 time steps, each including one matrix evaluation and 40 Jacobi sweeps. This results in about 100 seconds CPU time. The adjoint state is only about 100-200 Jacobi sweeps and $5 - 10$ seconds because we have fixed

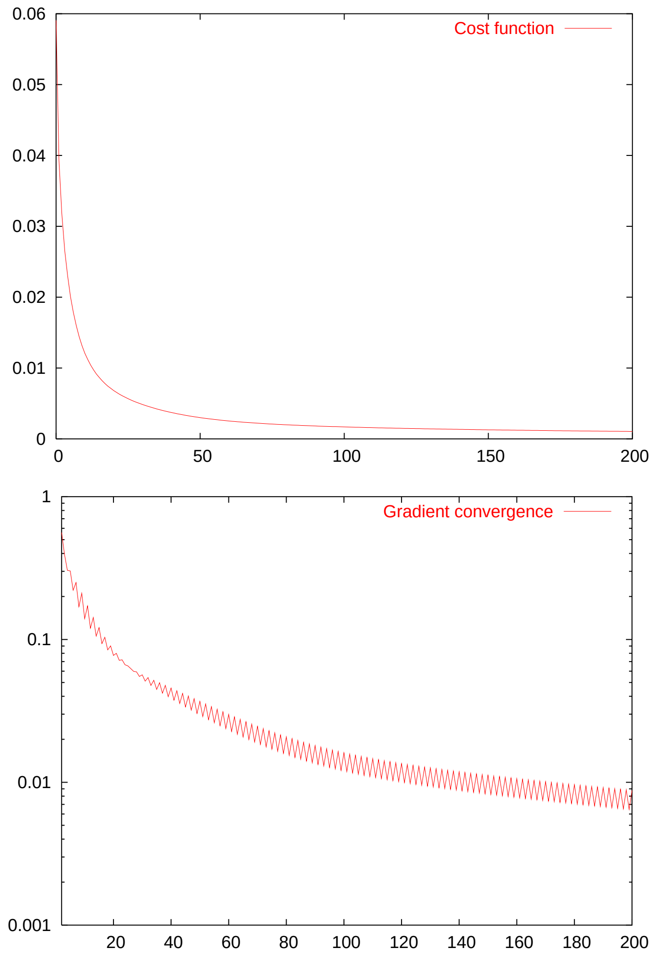

Fig. 5. Application to an optimal shape design problem (780 parameters, 11,000 state variables). Figure displays the convergence of a traditional SQP method, with perfect resolution of all linear systems

the accuracy order of the whole numerical CFD model to first-order in order to reduce the cost of the test. With 200 iterations, see Fig.5, the gradient of functional is decreased by two orders of magnitude. On the cost functional level, it appears that the 100 last iterations are useful only to decrease it of one extra. For practical applications, only a few iterations, typically less than 50 would be

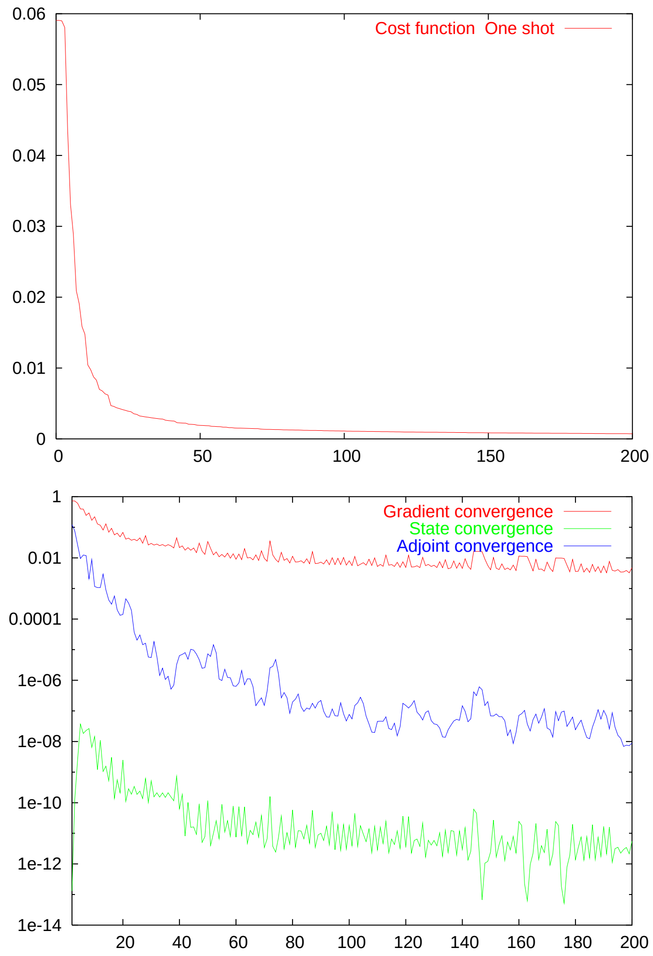

Fig. 6. Application to an optimal shape design problem (780 parameters, 11,000 state variables). One-shot optimiser. Upper figure displays the convergence of cost functional convergence. Lower figure shows the convergence of: functional gradient (upper curve) state residual convergence (lower curve), and in between, adjoint state residual convergence.

sufficient.

For the one shot option, state system partial resolution involves only one time step with one matrix evaluation and 20 Jacobi sweeps. The adjoint system involves only 20 Jacobi sweeps. We present the result of 200 iterations, for a total

CPU of 40 minutes. The gradient decreases at about the same rate as for the case of SQP (Fig.6). In practice, 50-100 steps ($10 - 20$ minutes) will be enough for decreasing significantly the cost.

Since a complete state system resolution is 1.5 minutes, the optimal solution was obtained for a cost of about 15 state plus costate solutions, a figure 40 times smaller than for the SQP option.

5.3. Finer mesh results

The same method is applied to a finer geometry with 3222 shape parameters and 77,315 state variables. For this case, a single resolution of flow takes 23 minutes. One hundred optimization iterations with the SQP method took 44 hours CPU.

With the one-shot algorithm, convergence rates (Fig. 8) are remarkably close to both those of SQP computation and those of the coarse mesh. In practice, 100 steps (6 hours) will be enough for decreasing significantly the cost. That is again for a cost of about 15 state plus costate solutions

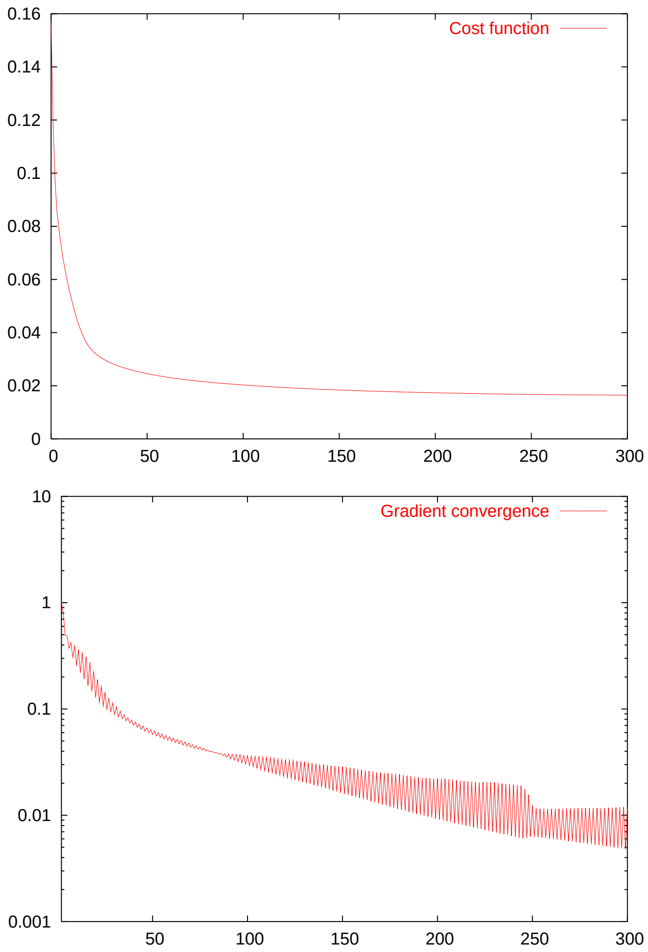

Fig. 7. Application to an optimal shape design problem (3222 parameters, 77,315 state variables). SQP optimiser. Cost functional (upper) and functional gradient (lower) convergence.

6. Conclusion and future directions

We have proposed a method for solving a class of optimization problems with a state equality constraint. We addressed the particular case where the linearized state and costate systems that are very expensive to solve. The key assumption is that a so-called non-expensive pseudo-Newton iteration for the state and costate

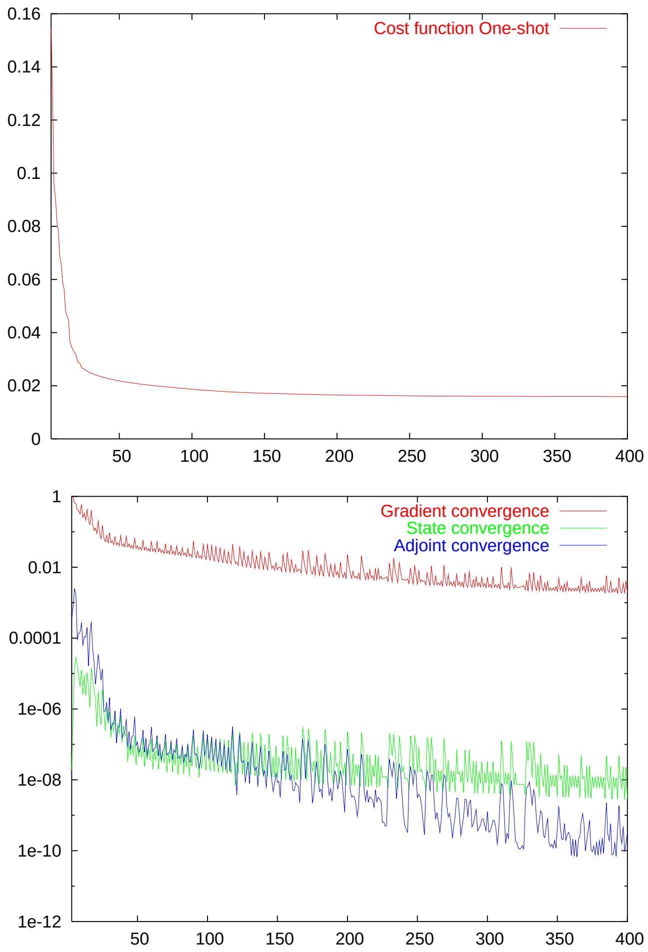

Fig. 8. Application to an optimal shape design problem (3222 parameters, 77,315 state variables). One-shot optimiser. Upper figure displays the convergence of cost functional convergence. Lower figure shows the convergence of: functional gradient (upper curve) state residual convergence (lower curve, for iterations 1 to 100), and in between, adjoint state residual convergence.

are available. This will allow a simultaneous solution of the KKT system. The new algorithm extends ideas of SQP, using in particular the BFGS acceleration. It involves a direct control of the Wolfe condition.

The proposed method works essentially like other one shot methods, but with robustness. This is gained thanks to heuristics needing extra computations that are not expensive.

In the examples we have tested, the global convergence of the minimization algorithm is only slightly degraded while each iteration is much less expensive. More importantly, this holds for various numbers of unknowns.

We think that this kind of algorithm (and its further improvements) qualifies as expected for being the key for direct and robust application to new problems, without divergence risk and parameter-tuning preliminary steps.

Acknowledgments: We thank Jean-Charles Gilbert for fruitful discussions and Stephen Wornom for rereading this paper.

References

1. S. Ta'asan A. Iollo, G. Kuruvila. Pseudo-time method for optimal shape design using Euler equations. Report 95-59, NASA contractor report 198205, ICASE, 1995.
2. H. Attouch, X. Goudou, and P. Redont. The heavy ball with friction method. I: The continuous dynamical system: Global exploration of the local minima of a real-valued function by asymptotic analysis of a dissipative dynamical system. *Commun. Contemp. Math.*, 2(1):1–34, 2000.
3. J.F. Bonnans, J.Ch. Gilbert, C. Lemaréchal, and C. Sagastizábal. *Numerical Optimization – Theoretical and Practical Aspects*. Springer Verlag, Berlin, 2002.
4. R.H. Byrd. Robust trust region methods for constrained optimization. Third SIAM Conference on Optimization, Houston, TX, may 1987.
5. R.H. Byrd, J.Ch. Gilbert, and J. Nocedal. A trust region method based on interior point techniques for nonlinear programming. *Mathematical Programming*, 89:149–185, 2000.
6. M.R. Celis, J.E. Dennis, and R.A. Tapia. A trust region strategy for nonlinear equality constrained optimization. In P.T. Boggs, R.H. Byrd, and R.B. Schnabel, editors, *Numerical Optimization 1984*, pages 71–82. SIAM Publication, Philadelphia, 1985.
7. F. Courty, A. Dervieux, B. Koobus, and L. Hascoet. Reverse automatic differentiation for optimum design: from adjoint state assembly to gradient computation. *accepted in Optimization Methods and Software Journal*, 2002. http://www-sop.inria.fr/tropics/Francois.Courty/Publications.html.
8. A. Dadone and B. Grossman. Progressive optimization of inverse fluid dynamic design problems. *Computer and Fluids*, 29:1–32, 2000.
9. A. Dervieux, F. Courty, M. Vázquez, and B. Koobus. Additive multilevel optimization and its application to sonic boom reduction. In *Numerical Methods for Scientific Computing - JP60 Meeting, Variational Problems and Applications*, Jyvaskyla, Finland, 2002. to be published by CIMNE. http://www-sop.inria.fr/tropics/Francois.Courty/Publications.html.
10. J.Ch. Gilbert. On the realization of the Wolfe conditions in reduced quasi-Newton methods for equality constrained optimization. *SIAM Journal on Optimization*, 7:780–813, 1997.
11. C. Held and A. Dervieux. One shot airfoil optimisation without adjoint. *Computer and Fluids*, 31:1015–1049, 2002.

12. A. Jameson. Aerodynamic design via control theory. *SIAM Journal on Scientific Computing*, 3(3):233–261, 1988.
13. A. Jameson. Aerodynamic design via control theory. Report 91-2, NASA contractor report 187497, ICASE, 1991.
14. M. Lalee, J. Nocedal, and T. Plantenga. On the implementation of an algorithm for large-scale equality constrained optimization. *SIAM Journal on Optimization*, 8:682–706, 1998.
15. B. Mohammadi and O. Pironneau. *Applied shape optimization for fluids.* Clarendon Press - Oxford, 2001.
16. J. Nocedal and S.J. Wright. *Numerical Optimization.* Springer Series in Operations Research. Springer, New York, 1999.
17. E.O. Omojokun. *Trust region algorithms for optimization with nonlinear equality and inequality constraints.* PhD thesis, Department of Computer Science, University of Colorado, Boulder, Colorado 80309, 1991.
18. S. Ta'asan. One shot methods for optimal control of distributed parameter systems i: finite dimensional control. ICASE, Technical Report 91-2, Nasa contractor report, 1991.
19. A. Vardi. A trust region algorithm for equality constrained minimization: convergence properties and implementation. *SIAM Journal on Numerical Analysis*, 22:575–591, 1985.
20. M. Vazquez, A. Dervieux, and B. Koobus. Aerodynamical and sonic boom optimization of a supersonic aircraft. INRIA Report 4520, 2002.