2005 American Control Conference
June 8-10, 2005. Portland, OR, USA

Second Order Adjoint-based Optimization of Ordinary and Partial Differential Equations with Application to Air Traffic Flow*

Robin L. Raffard† and Claire J. Tomlin‡

Abstract— **We present an algorithm to implement the second order Newton method on ordinary differential equation (ODE) and partial differential equation (PDE) optimization programs. The algorithm is based on the direct computation of the Newton step without explicitly calculating the second derivative (Hessian) of the objective function. The method poses the search for the Newton step as a convex quadratic optimization program. We apply our method to (a) dynamical systems driven by ODEs and to (b) constrained PDE optimization programs in the context of air traffic flow. In both cases, our implementation of the Newton method shows much faster convergence than first order algorithms, while not significantly increasing computational time.**

I. Introduction

Optimal control of ordinary differential equations (ODEs) and partial differential equations (PDEs) includes, among others, applications in trajectory planning [6], aerodynamic design [9], [10], [4], turbulent flow control [3], [1] and air traffic flow control [2]. For such problems, the decision variables are continuous functions – as the geometric shape of an airfoil in aerodynamics or the velocity of the air traffic flow – and thus, when discretized, could result in a very large set of discrete design variables. Therefore, from an optimization standpoint, control of ODEs and PDEs generally consists of a high dimensional optimization program. For such programs, first order methods such as the gradient algorithm, the smoothed gradient method, or the steepest descend method have proved to give particularly low performance when the condition number of the Hessian of the objective function is large [5].

A standard method to overcome the difficulties of first order methods is the Newton method. In this approach, the descent direction is chosen as the minimizer of the second order approximation of the objective function. The drawback of the Newton method is that the second derivative of the objective function (the Hessian) needs to be computed – and inverted. For an objective function whose variables are subject to PDEs constraints, there exists no systematic procedure to derive the Hessian. Therefore, only algorithms approximating the Hessian are currently used. The major example is the quasi Newton method [7] which builds up second derivative information by estimating the curvature along a sequence of search directions. However, the length of the sequence is proportional to the number of variables [11], which is problematic in high dimensional optimization.

In this paper, we propose a method to compute the Newton step directly – without having to compute explicitly the Hessian of the objective function. We pose the search for the Newton step as an auxiliary convex quadratic optimization program, in which we minimize the norm of the first order approximation of the gradient. The constraints of this auxiliary problem are the linearized forms of the original PDE and its adjoint equation; they represent the dependence of the first order variation of the state and the costate on the first order variation of the input. This quadratic program is solved using standard PDE optimization techniques such as the conjugate gradient method if the Hessian is positive definite, or the gradient algorithm if the Hessian is not. In the particular case of ODEs, the variables can be easily discretized to obtain either a non-singular linear program or a standard discrete quadratic program.

In Section II, we introduce the class of problems we propose to investigate and we shed some light on the difficulties of first order methods. In Section III, we present our method to implement the Newton method. In Section IV, we illustrate this approach through applications in trajectory planning, and applications in air traffic flow control. We compare the performance of our Newton step approach with that of the gradient algorithm and the smoothed gradient algorithm. We show that the Newton method performs much better, particularly when the decision variables are constrained.

II. Optimization of ODEs/PDEs: Presentation and Previous Work

A. Problem formulation

We will consider optimization problems in which variables are constrained by differential equations.

$$\begin{array}{ll} \text{Minimize} & J(u,x) \\ \text{Subject to} & \mathcal{N}(u,x)=0 \\ & r(u,x)\leq 0 \end{array} \qquad \text{(P1)}$$

J is a twice differentiable real value function; u is the continuous control variable and is, in most engineering applications, a function of time and/or space; x is the continuous state variable and is also generally a function of time and/or space. $\mathcal{N}$ is an operator, possibly nonlinear, which represents the governing differential equations of the system. Finally, r is a twice differentiable function, and represents constraints on the state and the input variables.

*Research supported by NASA under Grant NCC 2-5422 and NAG 2-1564 and by the ONR MURI award "Computational Methods for Collaborative Motion".

†Ph.D Student, Aeronautics and Astronautics, Stanford University. Corresponding author. rraffard@stanford.edu.

‡IEEE Member, Associate Professor, Aeronautics and Astronautics, Courtesy Associate Professor, Electrical Engineering.

Example 1: **Dynamical systems driven by ODEs.** We first consider an example in which $\mathcal{N}(u,x) = 0$ encodes an ordinary differential equation (ODE) of the form $\dot{x}(t) = f(x(t),u(t))$. Furthermore, we write the problem as a standard optimal control problem of a dynamical system:

$$\begin{array}{ll} \text{Minimize} & J(u,x) = \int_0^T h(x(t),u(t))\,dt \\ \text{Subject to} & \dot{x}(t) = f(x(t),u(t))\,,\ \ x(0) = x_0 \\ & r(u(t),x(t)) \leq 0 \end{array} \tag{P2}$$

The constraint $r(u(t),x(t)) \leq 0$ represents bounds on the admissible input and the authorized state of the system. The particular case of aircraft trajectory planning will be chosen for the numerical demonstration of our algorithm.

Example 2: **Transport PDEs for Air Traffic flow.** In the second example, $\mathcal{N}(u,x) = 0$ encodes transport PDEs involved in air traffic flow. It has been demonstrated [2] that flow of aircraft can be analyzed and controlled using an Eulerian viewpoint of the airspace. In this formulation, the state variable of the system is the density of aircraft $\rho(s,t)$ (s being the position), which represents the number of aircraft per unit length of jetway. The control variable is the velocity $v(s,t)$ which the air traffic controller can prescribe to the aircraft located at position s and time t.

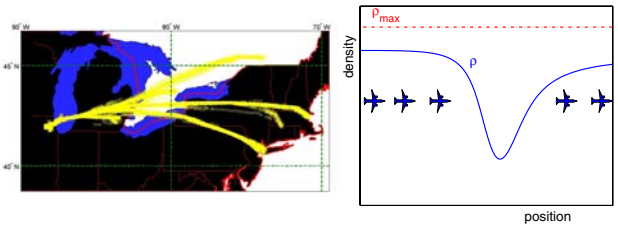

Fig. 1. *Left*: Air traffic flow from New-York, Boston and Montreal landing in Chicago, over a 24 hour period. *Right*: Given a spatial distribution of aircraft, the density $\rho(s,t)$ is defined as the number of aircraft per unit length of jetway.

Given a velocity field $v(s,t)$, the density of aircraft satisfies the continuity equation [2]:

$$\frac{\partial \rho(s,t)}{\partial t} + \frac{\partial \rho(s,t)v(s,t)}{\partial s} = 0 \tag{1}$$

The problem we propose to solve is the following: We would like to determine the velocity field which maximizes the number of aircraft landing at the destination airport under the constraint that the density does not exceed the safety density $\rho_{\max}$.

$$\begin{array}{ll} \text{Minimize} & J(\rho,v) = -\int_0^T \rho(L,t)v(L,t)\,dt \\ \text{Subject to} & \frac{\partial \rho}{\partial t} + \frac{\partial \rho v}{\partial s} = 0 \\ & \rho \leq \rho_{\max}\,, \quad v_{\min} \leq v \leq v_{\max} \end{array} \tag{P3}$$

$-J(\rho,v)$ represents the total number of aircraft landing at the final destination $s = L$. $v_{\min}$ and $v_{\max}$ are bounds on the authorized aircraft velocity.

B. Descent Algorithms

Problems (P1)-(P3) are subject to both inequality and equality constraints. We first eliminate the inequality constraints on (P1), using a logarithmic barrier

$$\begin{array}{ll} \text{Minimize} & J(u,x) - \epsilon \mathbf{1}^T \log(-r(u,x)) \\ \text{Subject to} & \mathcal{N}(u,x) = 0 \end{array} \tag{P4}$$

(P4) and (P1) are equivalent when $\epsilon \to 0$. The problem is now to find a descent direction for the control input u. For convenience, let us note $I(u) = J(u,x) - \epsilon \mathbf{1}^T \log(-r(u,x))$, with x such that $\mathcal{N}(u,x) = 0$. Work by Lions [12] and then Jameson [9] has shown that the gradient $\nabla I(u)$ can be derived using the adjoint method. This method returns the gradient of I as a function of the control input u, the state x and a costate p: $\nabla I(u) = \mathcal{G}(u,x,p)$. The costate q solves a backward PDE, called the adjoint equation: $\mathcal{N}^*(u,x,p) = 0$.

C. Disadvantanges of first order methods

First order methods such as the gradient algorithm or the smoothed gradient method perform poorly when the condition number of the Hessian of I is high (say, 100). In these cases, some components of the gradient highly fluctuate as a function of u whereas some other components vary slowly with u. Components with high amplitude and high fluctuations are updated while the other components remain almost unchanged during the algorithm. This phenomenon is illustrated in Figure 2, which shows the control variable as well as the gradient of a dynamical system for which the control variable is upper bounded. This example has been borrowed from the trajectory planning problem discussed in Section 4. The control variable $u(t)$ corresponds to the turning rate angle $w(t)$. At points at which the constraint is active, the gradient is "noisy" and is high in amplitude, and at points at which the input variable is far from the constraint, the gradient is not updated, although it should be since the gradient at these points is non zero.
Other methods such as the steepest descent method are problem dependent and need a good choice of preconditioner, *i.e.* a good change of coordinates, which may not be easy to find. In contrast, the Newton method automatically rescales the variables of the problem in Hessian norm which considerably descreases the condition number of the Hessian near optimum, leading to robust and very fast convergence [5]. In next section, we will propose a method to derive the Newton step for optimization programs of ODEs/PDEs.

III. Second Order Method

A. Newton step

For any control input u, the Newton step Δu_{nt} is defined as the input that should be added to u in order to set the first order approximation of the gradient to 0. Therefore,

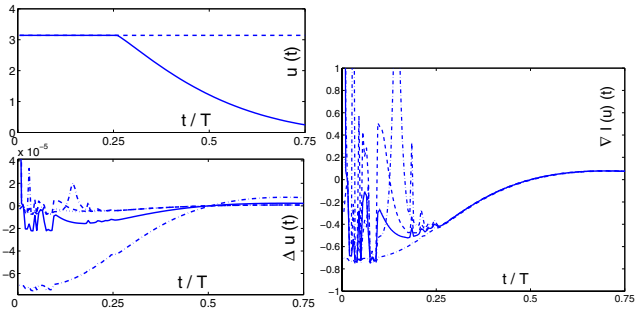

Fig. 2. Example of gradient descent iterations, for a problem in which the control variable $u(t)$ is subject to the constraint $u(t) \leq u_{\max}$. *Top left*: 5 updates of the control variable which are almost indetectable from each other. The 5 updates, $u(t)\,,\ t \in [0, 3T/4]$ are gathered on the solid line and the constraint is represented by the dashed line. The gradient algorithm is unable to update $u(t)$, although it is far from being optimum. *Bottom left*: Difference between the 5 updates, $u(t)\,,\ t \in [0, 3T/4]$. The difference is very small (of the order of 10^{-5}). *Right*: 5 updates of the gradient. The gradient is updated for the times at which the contol variable is almost optimum: $t \in [0, T/4]$, and not for the others: $t \in [T/4, T]$.

using $\widehat{\nabla I}(u+\delta u)$ to denote the first order approximation of $\nabla I(u+\delta u)$, we have:

$$\widehat{\nabla I}(u+\Delta u_{\text{nt}}) = \nabla I(u) + \nabla^2 I(u)\Delta u_{\text{nt}} = 0 \qquad (2)$$

$\nabla^2 I(u)$ denotes the second order derivative of I, and is also referred to as the Hessian, noted $\mathcal{H}$. Now, in order to compute Δu_{nt}, we need to give an explicit expression for $\mathcal{H}$. For this purpose, we write the first order variation of the gradient $\nabla I(u)$. Recall that the gradient is a function of the input, the state, and the costate: $\nabla I(u) = \mathcal{G}(u,x,p)$; thus

$$\begin{aligned}\nabla I(u+\delta u) - \nabla I(u) \quad = \quad & \nabla_u \mathcal{G}(u,x,p)\ \delta u \\ & + \nabla_x \mathcal{G}(u,x,p)\ \delta x \\ & + \nabla_p \mathcal{G}(u,x,p)\,\delta p + \mathcal{O}||\delta u||^2\end{aligned} \qquad (3)$$

Keeping only the first order terms, we obtain

$$\begin{aligned}\mathcal{H}\delta u \quad = \quad & \nabla_u \mathcal{G}(u,x,p)\ \delta u + \nabla_x \mathcal{G}(u,x,p)\ \delta x \\ & + \nabla_p \mathcal{G}(u,x,p)\ \delta p\end{aligned} \qquad (4)$$

δx and δp are the first order variation of the state and the costate generated by a first order variation of the input δu. Therefore $x+\delta x$ and $p+\delta p$ solve the state and the costate PDEs to the first order:

$$\begin{aligned}&\mathcal{N}(u+\delta u, x+\delta x) = \mathcal{O}||\delta u||^2 \\ &\mathcal{N}^*(u+\delta u, x+\delta x, p+\delta p) = \mathcal{O}||\delta u||^2\,,\end{aligned} \qquad (5)$$

which can be written as

$$\widehat{\mathcal{N}}(\delta u, \delta x) = 0\,, \quad \widehat{\mathcal{N}^*}(\delta u, \delta x, \delta p) = 0 \qquad (6)$$

in which $\widehat{\mathcal{N}}$ represents the linearized form of $\mathcal{N}$ and $\widehat{\mathcal{N}^*}$ represents the linearized form of $\mathcal{N}^*$. Finally, using (4), (6) and (2), the differential equation satisfied by the Newton step Δu_{nt} reads

$$\begin{aligned}&\nabla_u \mathcal{G}(u,x,p)\ \Delta u_{\text{nt}} + \nabla_x \mathcal{G}(u,x,p)\ \delta x \\ &\qquad + \nabla_p \mathcal{G}(u,x,p)\ \delta p = -\mathcal{G}(u,x,p) \\ &\widehat{\mathcal{N}}(\Delta u_{\text{nt}}, \delta x) = 0\,, \quad \widehat{\mathcal{N}^*}(\Delta u_{\text{nt}}, \delta x, \delta p) = 0\end{aligned} \qquad (7)$$

In the case in which $\mathcal{N}(u,x)$ encodes an ordinary differential equation, this system of equations is a *linear* system of ODEs and can be solved numerically using semi-implicit linear discretization schemes. In other cases, this system is a *linear* system of PDEs and we propose the following auxiliary optimization program to compute Δu_{nt}:

$$\begin{aligned}&\Delta u_{\text{nt}} = \underset{\delta u}{\text{argmin}}\{||\mathcal{H}\delta u + \nabla I(u)||^2\} \\ &\text{subject to } \widehat{\mathcal{N}}(\delta u, \delta x) = 0 \\ &\qquad\qquad \widehat{\mathcal{N}^*}(\delta u, \delta x, \delta p) = 0\end{aligned} \qquad \text{(P5)}$$

where $\mathcal{H}$ is given by (4).

In this optimization program, the Newton step Δu_{nt} is expressed as the minimizer of the the first order approximation of the gradient. Mathematically, the constraints of this problem are homogeneous linear PDEs and the objective is a positive definite quadratic form. Therefore, the problem is a (convex) quadratic program, which certifies that a descent algorithm will return a global optimal solution, *i.e.* the Newton step. We can solve this optimization program for PDEs in the following manner. (a) If $\mathcal{H}$ is positive definite, we use a conjugate gradient method [8]. The computation of Δu_{nt} is then easy and very fast. (b) If $\mathcal{H}$ is not positive definite, we use a gradient descent method and the complexity of this procedure is roughly the same as the complexity of the original problem. Therefore, we could argue that, for this last case, a gradient algorithm might fail to converge in implementation and therefore not return the exact Newton step. This issue is resolved in practice. For most cases, an estimate for the Newton step is sufficient and the one given by the auxiliary optimization problem is usually a good descent direction, much better than the gradient one. This will be illustrated in the framework of the air traffic flow example.

B. Application to Dynamical Systems

We will now show how this approach can be applied to dynamical systems driven by ODEs, as introduced in example 1 of Section II-A.

1) Perturbation of the gradient: The gradient of the objective function I can been derived as $\nabla I(u) = \mathcal{G}(u,x,p) = \nabla_u f(x,u)\,p + \nabla_u h(x,u)$ [6]. Therefore a first order variation of $\nabla I(u)$ leads to

$$\begin{aligned}\nabla I(u+\delta u) - \nabla I(u) = & \nabla_u f(x,u)\ \delta p \\ & + \nabla_u^2\ p^T f(x,u)\ \delta u + \nabla_{xu}^2\ p^T f(x,u)\ \delta x \\ & + \nabla_u^2\ h(x,u)\ \delta u + \nabla_{xu}^2\ h(x,u)\ \delta x + \mathcal{O}||\delta u||^2\end{aligned} \qquad (8)$$

where $\nabla_u f$ is the matrix whose ith row and jth column is $\frac{\partial f_j}{\partial u_i}$, and where $\nabla^2_{xu} h$ is the matrix whose ith row and jth column is $\frac{\partial^2 h}{\partial u_i \partial x_j}$. Therefore,

$$\begin{aligned}\mathcal{H}\delta u &= \underbrace{(\,\nabla^2_u\, p^T f(x,u) + \nabla^2_u\, h(x,u)\,)}_{\mathcal{H}_1}\,\delta u \\ &+ \underbrace{(\,\nabla^2_{xu}\, p^T f(x,u) + \nabla^2_{xu}\, h(x,u)\,)}_{\mathcal{H}_2}\,\delta x \\ &+ \underbrace{\nabla_u f(x,u)}_{\mathcal{H}_3}\,\delta p\end{aligned} \tag{9}$$

The variation of the state and the costate: δx and δp satisfy the following linear ordinary differential equation:

$$\begin{aligned}\begin{bmatrix}\dot{\delta x}\\ \dot{\delta p}\end{bmatrix} &= \underbrace{\begin{bmatrix}\nabla_x f^T & 0\\ -\nabla^2_x h - \nabla^2_x p^T f & -\nabla_x f\end{bmatrix}}_{A(t)}\begin{bmatrix}\delta x\\ \delta p\end{bmatrix} \\ &+ \underbrace{\begin{bmatrix}\nabla_u f^T\\ -\nabla^2_{ux} h - \nabla^2_{ux}\, p^T f\end{bmatrix}}_{B(t)}\delta u\end{aligned} \tag{10}$$

2) Derivation of the Newton step via linear programming: We can now discretize this linear ODE accurately using semi-implicit numerical schemes. Posing $y = (x, p)$, we discretize (10) as follows

$$\begin{aligned}&\delta y(k+1) - \delta y(k) = \tfrac{1}{2}\big(A(k+1)\,\delta y(k+1)\\ &+A(k)\,\delta y(k)\big) + \tfrac{1}{2}\big(B(k+1)\,\delta u(k+1) + B(k)\,\delta u(k)\big)\end{aligned} \tag{11}$$

for $k = 0, \ldots, N-1$, where $Ndt = T$.
Finally, the Newton step can be found by solving the following linear program of variables δu, δx and δp.

$$\begin{aligned}&\mathcal{H}_1(k)\delta u(k) + \mathcal{H}_2(k)\delta x(k)\\ &\qquad +\mathcal{H}_3(k)\delta p(k) = -\nabla I(u(k))\\ &\delta x(0) = 0\,, \quad \delta p(N) = 0\\ &\delta y(k) = (\delta x(k), \delta p(k))\\ &\delta y(k+1) - \delta y(k) = \tfrac{1}{2}\big(A(k+1)\delta y(k+1)\\ &+A(k)\delta y(k)\big) + \tfrac{1}{2}\big(B(k+1)\delta u(k+1) + B(k)\delta u(k)\big)\end{aligned} \tag{12}$$

where the first and fourth equations have to be satisfied for $k = 0, \ldots, N$, and the last equation has to be satisfied for $k = 0, \ldots, N-1$.
Remark: In the case in which the linear system (12) is singular, the quadratic program (P5) has to be solved instead. It will return the descent direction δu which minimizes the Euclidean norm of the first order approximation of the gradient.

C. Application to Air Traffic Flow

We will now show how to derive the Newton step in the case of the air traffic flow control problem, introduced in example 2 of Section II-A.

1) Derivation of the auxiliary optimization program: The gradient of the cost function with respect to the control input can be derived as [2]

$$\nabla I(v) = \rho\frac{\partial p}{\partial s} + \frac{1}{M}\Big(\frac{1}{v_{\max} - v} - \frac{1}{v - v_{\min}}\Big) \tag{13}$$

where the costate p satisfies the following backward PDE

$$\frac{\partial p}{\partial t} + v\frac{\partial p}{\partial s} = \frac{1}{\rho - \rho_{\max}}\,, \quad p(s,T) = 0\,, \quad p(L,t) = -1 \tag{14}$$

Perturbing $\nabla I(v)$ to the first order, we get the expression for the Hessian in terms of the variations of the input, of the state and of the costate.

$$H\delta v = \delta\rho\,\frac{\partial p}{\partial s} + \rho\frac{\partial\,\delta p}{\delta s} + \frac{1}{M}\Big(\frac{\delta v}{(v_{\max} - v)^2} + \frac{\delta v}{(v - v_{\min})^2}\Big) \tag{15}$$

A first order perturbation of the state and the costate equations gives:

$$\begin{aligned}\tfrac{\partial\,\delta\rho}{\partial t} + \tfrac{\partial\,\delta\rho\; v}{\partial s} &= -\tfrac{\partial\rho\,\delta v}{\partial s}\\ \tfrac{\partial\,\delta p}{\partial t} + \delta v\,\tfrac{\partial p}{\partial s} + v\tfrac{\partial\,\delta p}{\partial s} &= -\tfrac{\delta\rho}{M(\rho-\rho_{max})^2}\end{aligned} \tag{16}$$

All boundary conditions are trivial and have not been stated because of space constraint. We are now in a position to form the cost function for the auxiliary optimization program:

$$\begin{aligned}J_A &= \int_0^L\int_0^T (\mathcal{H}\delta v + \nabla I(v))^2\,dt\,ds\\ &= \int_0^L\int_0^T (\nabla I(v) + \delta\rho\tfrac{\partial p}{\partial s} + \rho\tfrac{\partial p}{\partial s} + \tfrac{\delta v}{M}k)^2\,dt\,ds\end{aligned} \tag{17}$$

where, $k = \frac{1}{(v_{\max}-v)^2} + \frac{1}{(v-v_{\min})^2}$. The problem is to find the minimizer of this cost function given the system of PDEs (16).

2) Solving the auxiliary optimization program: In order to solve the auxiliary optimization problem, we compute the gradient of J_A with respect to δv given the system (16). For this purpose, we perturb both J_A and (16) to the first order. Note that v, ρ and p are fixed parameters of the problem.

$$\delta J_A = \int_0^L\int_0^T a(\delta^2\rho\frac{\partial p}{\partial s} + \rho\frac{\partial\delta^2 p}{\partial s} + \frac{\delta^2 v}{M}k)dt\,ds \tag{18}$$

where $a = 2(\nabla I(v) + \mathcal{H}\delta v)$.

$$\begin{aligned}\tfrac{\partial\,\delta^2\rho}{\partial t} + \tfrac{\partial\,\delta^2\rho\; v}{\partial s} &= -\tfrac{\partial\rho\,\delta^2 v}{\partial s}\\ \tfrac{\partial\,\delta^2 p}{\partial t} + \delta^2 v\,\tfrac{\partial p}{\partial s} + v\tfrac{\partial\,\delta^2 p}{\partial s} &= -\tfrac{\delta^2\rho}{M(\rho-\rho_{max})^2}\end{aligned} \tag{19}$$

In order to eliminate the terms both in $\delta^2\rho$ and $\delta^2 p$ in δJ_A, we introduce two new costate variables $q(s,t)$ and $r(s,t)$. Multiplying respectively the PDE satisfied by $\delta^2\rho$ by q and the PDE satisfied by $\delta^2 p$ by r; integrating twice by parts and making the right choice for the adjoint variables q and r, we can replace the terms in $\delta^2\rho$ and $\delta^2 p$ by terms in q and r, and we get the following expression for the variation of J_A:

$$\delta J_A = \int_0^L \int_0^T \underbrace{(\frac{a\ k}{M} - \rho\frac{\partial q}{\partial x} + r\frac{\partial p}{\partial s})}_{\nabla J_A} \delta^2 v\, dt\, ds \qquad (20)$$

provided that

$$\begin{array}{l} \frac{\partial r}{\partial t} + \frac{\partial rv}{\partial s} = -a\frac{\partial \rho}{\partial s} - \rho\frac{\partial a}{\partial s} \\ \frac{\partial q}{\partial t} + v\frac{\partial q}{\partial s} = a\frac{\partial p}{\partial s} + \frac{r}{M(\rho-\rho_{\max})^2} \end{array} \qquad (21)$$

3) Algorithm: The algorithm to find the Newton step is, in the general case, a gradient descent on J_A.

Newton step search Algorithm

Given (u, ρ, p), form $\nabla I(v)$ and k. Set the starting value for δv as $-\nabla I(v)M/k$

Repeat:

1. *Solve for $\delta\rho$ and δp (16).* Form a (18).
2. *Solve for r (21), then for q (21) and form ∇J_A* (20).
3. *Line search: Compute $\beta > 0$ such that $J_A(\delta v - \beta\nabla J_A)$ is minimized.*
4. *Update:* $\delta v := \delta v - \beta\nabla J_A$.

Terminate when ∇J_A is small.

Return $\Delta v_{\text{nt}} = \delta v$.

IV. Results

In this section, we will compare the performance of the gradient algorithm, the smoothed gradient algorithm, and the Newton method in the context of (a) trajectory planning, which consists of a nonlinear dynamical system optimization problem and (b) air traffic flow, which represents a nonlinear PDE optimization problem. The smoothed gradient will be taken as the projection of the gradient on the space of the Legendre polynomials of degree less than 6.

A. Trajectory planning

We implement the Newton descent algorithm in the context of trajectory planning. An aircraft pursuer has an objective to minimize its distance from another aircraft leader. The dynamics of the aircraft is the traditional 2D kinematic nonlinear model.

$$\left\{ \begin{array}{rcl} \dot{x}_i(t) &=& v_i(t)\cos(\phi_i(t)) \\ \dot{y}_i(t) &=& v_i(t)\sin(\phi_i(t)) \\ \dot{\phi}_i(t) &=& \omega_i(t) \end{array} \right. \qquad (22)$$

Both $\omega(t)$ and $v(t)$ have saturation: $\omega_{\min} \le \omega(t) \le \omega_{\max}$ and $v_{\min} \le v(t) \le v_{\max}$.
Therefore, the problem reads

$$\begin{array}{ll} \text{Minimize} & J = \int_0^T (x(t) - x_{\text{leader}}(t))^2 \\ & \qquad +(y(t) - y_{\text{leader}}(t))^2\ dt \\ \text{Subject to} & (22) \\ & \omega_{\min} \le \omega(t) \le \omega_{\max}\,,\ v_{\min} \le v(t) \le v_{\max} \end{array} \qquad \text{(P6)}$$

The scenario is chosen so as to fully exploit the nonlinearity of the dynamics of the aircraft as well as the constraints on the input variables. For instance, the heading angle is expected to range from $\pi/2$ to $-\pi/2$ and the velocity and heading angle rate inputs are expected to reach saturation. For both the gradient descent and the smoothed gradient descent, we used 50 descent iterations. For the Newton descent algorithm, we only used 6 descent iterations. Figure 3 displays the computed trajectories. The gradient algorithm and the smoothed gradient method both return a similar trajectory which is far from the trajectory computed by the Newton method. Figure 4 shows the input variables for the 3 different methods. Unlike the Newton method, the first order methods are unable to push the input variables to saturation when it should do so. Table I returns the costs, where the Newton method is shown to give the best result.

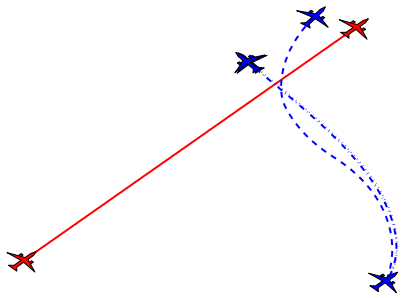

Fig. 3. Trajectories of the aircraft leader (solid line) and the aircraft pursuer given by the Newton method (dashed line), the gradient algorithm (dashed dotted line) and the smoothed gradient method (dotted line). The dashed dotted and dotted lines are almost on top of each other. The Newton method drives the pursuer much closer to the leader than the first order methods do.

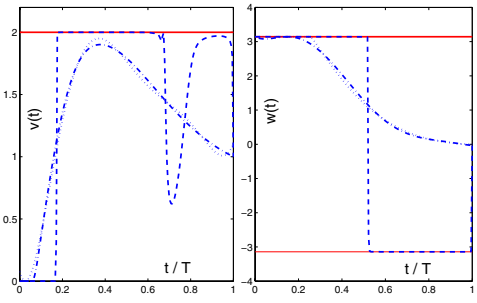

Fig. 4. Input variables as a function of time as given by the Newton method (dashed line), the gradient algorithm (dashed dotted line) and the smoothed gradient method (dotted line). The Newton algorithm exploits full potential of the dynamics of the aircraft. On the contrary, the first order methods drive only the turning angle rate to saturation for $t \in [0, 0.25T]$. This example is the same as the one displayed in Fig. 2, in which we had also displayed the gradient of the objective function with respect to w for the gradient algorithm.

	Gradient	Smoothed gradient	Newton
Final Cost (J)	0.5967	0.6003	0.5571
Computational Time	396 sec.	330 sec.	149 sec.

TABLE I. Performance of the different methods for the trajectory planning problem. The Newton method returns the lowest cost and therefore the best solution.

B. Air traffic flow control

In this section, we implement the Newton descent algorithm in the framework of air traffic flow. We consider the case in which a set of aircraft are landing at a destination airport located at $s = L$. The problem is to compute the velocity open loop controller which maximizes the number of aircraft landing at $s = L$, for a given time interval $t \in [0, T]$. Furthermore, the density $\rho(s,t)$ is subject to the constraint $\rho(s,t) \leq \rho_{\max}$. The initial density is chosen low enough, so that no matter how fast ($v = v_{\max}$) the aircraft travel, the maximum density will never be exceeded. Therefore, an optimal control is simply $v(s,t) = v_{\max}$. This scenario is a good test case (a) because the optimal cost for the problem is known and (b) because most algorithms will perform poorly when v approaches $v_{\max}$. The results are displayed in Figure 5 and TABLE II. The solution returned by the Newton method makes the aircraft travel faster than the first order methods, which leads to a lower cost and therefore a better solution.

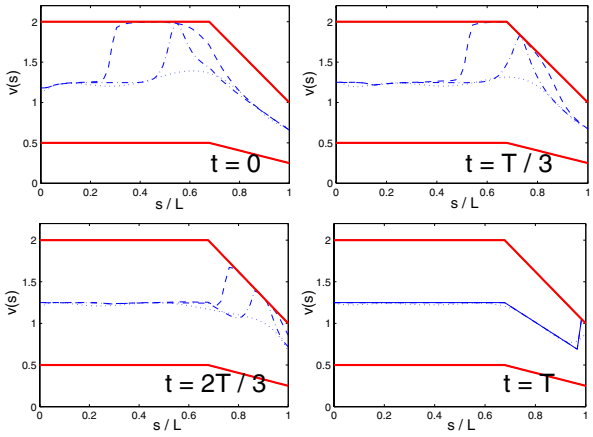

Fig. 5. *Air traffic flow control (ATC) low density case.* Velocity profile returned by the Newton method (dashed line), the gradient algorithm (dashed dotted line) and the smoothed gradient method (dotted line), at 4 different times. $v_{\min}$ and $v_{\max}$ are indicated by thick solid lines. With the Newton method, aircraft initially far from the airport (located at s=0.25 at t=0) speed all the way and reach the airport by $t = T$, thus contributing to a high payoff, *i.e.* a low cost (See TABLE II).

V. Conclusion

We have proposed an original method to implement the Newton method in the context of optimization for systems driven by ordinary differential equations and partial differential equations. Applications of the method to trajectory planning for aircraft of nonlinear dynamics and to air traffic flow control modeled by propagation PDEs show that this algorithm returns better results than first order methods. The discrepancy between the performance of the methods is exacerbated in particular when the input variable is subject to inequality constraints. Finally, in the context of PDE optimization, solving the auxiliary optimization programs slows down the algorithm – typically the Newton method is twice as slow as the first order methods. However, the improvement in the objective function is substantial enough to make the method desirable.

	Gradient	Smoothed gradient	Newton
Final Cost (J)	-3.604	-2.748	-3.999
Computational Time	296 sec.	291 sec.	491 sec.

TABLE II. *ATC low density case*: Performances of the different algorithms. The Newton method almost returns an optimal solution, for which the cost is -4. The computations have been stopped when the algorithms have converged *i.e* when the value of the cost function has become stationary. The algorithms could have run indefinitely, the objective functions would not have been improved more.

References

[1] O.M. Aamo and M. Krstic. *Flow Control by Feedback.* Springer, 2002.

[2] A. M. Bayen, R. L. Raffard, and C. J. Tomlin. Adjoint-based constrained control of Eulerian transportation networks: Application to air traffic control. *In the Proceedings of the American Control Conference, Boston, June 2004.*

[3] T.R. Bewley. Flow control: New challenges for a new renaissance. *Progress in Aerospace Science, 37:21-58, 2001.*

[4] J.T. Borggaard and J.A. Burns. A PDE Sensitivity Equation Method for Optimal Aerodynamic Design. *Journal of Computational Physics, Vol. 136, pages 366-384, 1997.*

[5] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2003.

[6] A. E. Bryson, Jr. and Y.-C. Ho. *Applied Optimal Control.* Hemisphere Publishing Corporation. Washington, D.C., 1975.

[7] P. E. Gill and M. W. Leonard. Reduced-hessian quasi-newton methods for unconstrained optimization. *SIAM J. Optim. Vol. 12, No 1, pp.209-237.*

[8] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization.* Academic Press. Harcourt Brace and Company, 1999.

[9] A. Jameson. Aerodynamic design via control theory. *Princeton University Report MAE 1824, ICASE Report No. 88-64, November 1988, also, J. of Scientific Computing, Vol. 3, 1988, pp. 233-260.*

[10] A. Jameson, Sriram, and L. Martinelli. A continuous adjoint method for unstructured grids. *16th AIAA Computational Fluid Dynamics Conference, AIAA Paper AIAA-2003-3955, Orlando, FL, June 23-26, 2003.*

[11] A. Jameson and J. C. Vassberg. Studies of alternative numerical optimization methods applied to the brachistochrone problem. *Computational Fluid Dynamics JOURNAL, vol.9 no.3, October 2000.*

[12] J.-L. Lions. Optimal control of systems governed by partial differential equations. *translated by S.K. Mitter, Springer Verlag, New York, 1971.*