AIAA JOURNAL
Vol. 45, No. 8, August 2007

Direct Treatment of State Constraints in Aerodynamic Shape Optimization Using Simultaneous Pseudo-Time-Stepping

Subhendu Bikash Hazra
University of Trier, 54286 Trier, Germany

DOI: 10.2514/1.28560

The paper deals with numerical method for constrained aerodynamic shape optimization problems. Pseudostationary system of state, costate, and design equations are solved using simultaneous pseudo-time-stepping. The method requires no additional globalization in design space and it blends in nicely with existing pseudo-time-stepping method for state and costate equations in this problem class. For convergence acceleration, a reduced sequential quadratic programming methods-based preconditioner is used. Additional state constraints are treated directly using the ideas of partially reduced sequential quadratic programming methods. The unconstrained design velocity is projected onto the tangent space of the state constraints by solving a quadratic programming problem involving the reduced Hessian and the reduced gradients. Computational examples are provided for drag reduction with constant lift and constant pitching moment for an RAE2822 airfoil at different discretization levels using different design parameter spaces and drag reduction with constant lift for a Supersonic Cruise Transport aircraft wing with geometrical constraints. Faster convergence is observed for finer design parameter space using the current method. The overall cost of computation is about seven times that of the forward simulation runs on a coarse grid, about 11 times that of the forward simulation runs on a fine grid for two additional state constraints in 2-D, and less than six times that of the forward simulation runs for single additional state constraint in 3-D.

Nomenclature

A	=	approximate Jacobian
B	=	reduced Hessian
B_F	=	far-field boundary
B_S	=	solid body
C_D	=	drag coefficient
C_L	=	lift coefficient
C_p	=	pressure coefficient
E	=	total energy
g	=	reduced gradient
H	=	total enthalpy
J	=	Jacobian
L	=	Lagrangian
$\boldsymbol{n}$	=	unit outward normal
p	=	pressure
q	=	vector of design variables
S_{ref}	=	reference area
(u_1, u_2, u_3)	=	velocity components
w	=	vector of state variables
(x, y, z)	=	Cartesian coordinates
α	=	angle of attack
γ	=	ratio of specific heats
$\partial\Omega$	=	flowfield boundary
λ	=	vector of adjoint variables
(ξ_1, ξ_2, ξ_3)	=	generalized coordinates
ρ	=	density
Ω	=	flowfield domain

I. Introduction

COMPUTATIONAL fluid dynamics (CFD) has made considerable progress so that it is being used extensively for analysis of any prescribed aerodynamic shape. Recently, a combination of CFD and numerical optimization method is being used to determine an optimum aerodynamic shape automatically for intended applications. The cost of computation is much cheaper than the cost of wind-tunnel experiments. This has proved to be extremely valuable in practice because it provides multiple alternatives to the designer. The best designs identified by numerical computations are confirmed for a decision by the wind-tunnel experiment. Mathematically, such shape design problems can be formulated as a control problem for systems governed by partial differential equations (PDEs) [1].

In control-theory-based design methods, the problem is regarded as optimal control of the flow equations by changing the shape of the boundary to achieve the goal. Gradient-based optimization methods to solve such problems proved to be most effective. In this method, a direction of descent is determined using the gradient informations of the cost function with respect to the design parameters. Once the direction is found, a step, determined by some means (which is known as line-search or globalization strategy), towards this direction is taken in each optimization iteration. This process is continued until convergence.

For computation of gradients or sensitivity derivatives one can use, for example, finite-difference method in which a small variation is introduced in each design parameter and the flow is recalculated to obtain the variation in the objective function. This is repeated for all the design parameters. The disadvantage of this method is that the number of flow calculations needed to estimate the gradient is proportional to the number of design parameters [2]. Continuous adjoint method has advantage over the finite-difference method. In this method, another set of partial differential equations, known as adjoint or costate equations, are to be solved. The cost of solving these equations is comparable to the cost of solving the state (flow) equations. In each iteration of a traditional gradient method, for example, steepest descent or conjugate gradient or quasi-Newton method, one has to solve the state and costate equations with sufficient accuracy. This has to be repeated several times until the convergence of the optimization algorithm. Therefore, in spite of using high-fidelity CFD, the overall cost of computation is quite high to get an optimal solution. Computational results based on these methodologies have been presented, among others, in [3–7] on structured grids. An application of this method on unstructured grids has been presented in [8].

In [9] we proposed a new method for solving such problems using simultaneous pseudo-time-stepping. This formulation is

Received 26 October 2006; revision received 19 February 2007; accepted for publication 19 February 2007.

advantageous because the steady-state flow (as well as adjoint) solution is obtained by integrating the pseudounsteady Euler (or Navier–Stokes) equations in this problem class. Therefore, one can use the same time-stepping philosophy for the whole set of equations and reduced sequential quadratic programming (SQP) methods based preconditioner can be used to accelerate the convergence. In [10–12] we have implemented that method for shape design examples using Euler equations. The method is "one-shot" because we perform one time step for each design update. It differs from the one-shot method of [13], in which the design variables are updated in a hierarchical manner. Simultaneous approach is also proposed in [14], which is based on successive linear programming involving solution of a linear adjoint system. In [15] all-at-once approach is proposed in the SQP scheme where simplification is introduced in the reduced Hessian so that the method requires no adjoint solution. Both the methods of [14,15] require line search and applied to 1-D problems.

Practical shape design problems involve additional state constraints, for example, shape optimization of aircraft for drag reduction with constant lift. Effectiveness of the optimization methods for aerodynamic shape design depends crucially on proper choice of the cost function, the constraints, and their treatment during the optimization. One can treat the constraints "indirectly," for example, by making some kind of transformation so that the constraints are added to the objective function (with some weighting) and the constrained problem is reduced to an unconstrained one. This kind of treatment is termed as "soft" constraints in [16]. It is well known (see, for example, [17], p. 144) that the reduced problem may not correspond always to the original problem and the solution process may not be an efficient one. Indirect treatment of constraints using traditional gradient methods can be found in [6,7,18].

Direct treatment of state constraints using traditional gradient methods to such problems in 2-D are carried out, among others, in [3,19]. The computational effort required there is about 40 forward simulation runs and 27 adjoint runs (nine adjoint runs for each of the objective function and the state constraints). Direct treatment of a single state constraint using our one-shot pseudo-time-stepping method has been carried out in [20,21] for 2-D problems. Here we apply the method to a 3-D problem. Additionally, we extend the method for two state constraints. The basic solution strategy is based on projecting the unconstrained design velocity onto the tangent space of the state constraints by solving a quadratic programming (QP) problem involving the reduced Hessian and the reduced gradients. Application examples for drag reduction with constant lift and constant pitching moment for an RAE2822 airfoil are included. In [22] we have used the indirect treatment of additional state constraint (as in [6]) using pseudo-time-stepping method. Here, we present the results of direct treatment of the state constraint for the same application example. The results show clear evidence of the advantages of direct treatment of additional state constraints.

The paper is organized as follows: We explain the unconstrained simultaneous pseudo-time-stepping method in the next section. Section III presents the problem with additional state constraints and its solution strategy using simultaneous pseudo-time-stepping. In Sec. IV, we present the state, costate, and design equations. Numerical results are presented in Sec. V. We draw our conclusions in Sec. VI.

II. Optimization Problem and Pseudounsteady Formulation of the Karush–Kuhn–Tucker Conditions

The optimization problem that we are dealing with in this study can be written (see, e.g., [9,12]) in abstract form as

$$\min I(w,q) \qquad \text{subject to } c(w,q)=0 \tag{1}$$

where $(w,q)\in X\times P$ (X, P are appropriate Hilbert spaces), I: $X\times P\to\mathbb{R}$ and c: $X\times P\to Y$ are twice Frechet-differentiable (with Y an appropriate Banach space). The Jacobian, $J=\frac{\partial c}{\partial w}$, is assumed to be invertible. Here, the equation $c(w,q)=0$ represents the steady-state flow equations (in our case, the Euler equations) together with the boundary conditions. The objective $I(w,q)$ is the drag of an airfoil/wing for the purposes of this paper. In what follows, we denote by the notation $(\,)^\top$ the transpose of the vector or the operator. In an optimal control setting, adjoint variables are from the dual space and the notation should reflect this. However, because ultimately we are dealing with discretized magnitudes, we do not distinguish between finite and infinite dimensional adjoints and use always the same notation.

The necessary optimality conditions for problem (1) can be formulated using the Lagrangian functional

$$L(w,q,\lambda)=I(w,q)-\lambda^\top c(w,q) \tag{2}$$

where λ is the Lagrange multiplier or the adjoint variable from the dual Hilbert space. If $\hat{z}=(\hat{w},\hat{q})$ is a minimum, then there exists a $\hat{\lambda}$ such that

$$\nabla_z L(\hat{z},\hat{\lambda})=\nabla_z I(\hat{z})-\hat{\lambda}^\top\nabla_z c(\hat{z})=0 \tag{3}$$

Hence, the necessary optimality conditions, known as Karush–Kuhn–Tucker (KKT) conditions, are

$$c(w,q)=0 \qquad \text{(state equation)} \tag{4a}$$

$$\nabla_w L(w,q,\lambda)=0 \qquad \text{(costate equation)} \tag{4b}$$

$$\nabla_q L(w,q,\lambda)=0 \qquad \text{(design equation)} \tag{4c}$$

Traditional so-called black-box methods, for example, steepest-descent method, conjugate gradient method, etc., involve solution of the state and the costate equations at each update of the design variables. These methods only act in the design space and assume that the state and the costate equations are solved sufficiently accurately. This leads to very high computational cost of these methods.

One can use, for example, reduced SQP (rSQP) method to solve the preceding system of nonlinear equations. A step of this method can also be interpreted as an approximate Newton step for the necessary conditions of finding the extremum of problem (1), because the updates of the variables are computed according to the linear system

$$\begin{pmatrix} 0 & 0 & A^\top \\ 0 & B & \left(\frac{\partial c}{\partial q}\right)^\top \\ A & \frac{\partial c}{\partial q} & 0 \end{pmatrix}\begin{pmatrix}\Delta w\\ \Delta q\\ \Delta\lambda\end{pmatrix}=\begin{pmatrix}-\nabla_w L\\ -\nabla_q L\\ -c\end{pmatrix} \tag{5}$$

This is the basic formulation of the inexact rSQP methods that we are using in the subsequent sections, especially in the context of preconditioning.

To determine the solution of this system using a simultaneous pseudo-time-stepping method, we look for steady-state solutions of the following pseudotime embedded evolution equations:

$$\frac{\mathrm{d}w}{\mathrm{d}t}+c(w,q)=0, \qquad \frac{\mathrm{d}\lambda}{\mathrm{d}t}+\nabla_w L(w,q,\lambda)=0$$
$$\frac{\mathrm{d}q}{\mathrm{d}t}+\nabla_q L(w,q,\lambda)=0 \tag{6}$$

This formulation is advantageous because the steady-state flow (as well as adjoint) solution is obtained by integrating the pseudounsteady Euler/Navier–Stokes (adjoint) equations in this problem class. However, this pseudotime embedded system, after semidiscretization, leads to a stiff system of ordinary differential equations (ODEs). Therefore, explicit time-stepping schemes (which are used in most applications of this problem class) may converge very slowly or may even diverge. To accelerate convergence, this

system needs preconditioning. The preconditioner that we use stems from rSQP methods as discussed in detail in [9,12].

In our implementation, we use the inverse of the matrix in Eq. (5) as a preconditioner for the time-stepping process. The pseudotime embedded system that we consider is

$$\begin{pmatrix} \dot{w} \\ \dot{q} \\ \lambda \end{pmatrix} = K \begin{pmatrix} -\nabla_w L \\ -\nabla_q L \\ -c \end{pmatrix} \tag{7}$$

where

$$K = \begin{bmatrix} 0 & 0 & A^\top \\ 0 & B & \left(\frac{\partial c}{\partial q}\right)^\top \\ A & \frac{\partial c}{\partial q} & 0 \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1}\frac{\partial c}{\partial q}B^{-1}\left(\frac{\partial c}{\partial q}\right)^\top A^{-\top} & -A^{-1}\frac{\partial c}{\partial q}B^{-1} & A^{-1} \\ -B^{-1}\left(\frac{\partial c}{\partial q}\right)^\top A^{-\top} & B^{-1} & 0 \\ A^{-\top} & 0 & 0 \end{bmatrix}$$

This seems natural because Eq. (5) can be considered as an explicit Euler discretization for the corresponding time stepping that we envision. Also, due to its block structure, the preconditioner is computationally inexpensive. The preconditioner employed is similar to the preconditioners for the KKT systems discussed in the context of Krylov subspace methods [23,24] and in the context of Lagrange–Newton–Krylov–Schur methods [25].

Within the inexact rSQP preconditioner, one has to look for an appropriate approximation of the reduced Hessian B. In particular, when dealing with partial differential equations constituting the state equations, the reduced Hessian can often be expressed as a pseudodifferential operator, the symbol of which can be computed and exploited for preconditioning purposes as in [9]. Here we use an update formula involving second-order information similar to Broyden–Fletcher–Goldfarb–Shanno (BFGS) updates as explained in Sec. V. The block matrices A and $A^\top$, corresponding to the state and costate equations in the preconditioner, are just identity matrices in the current implementation.

III. Scalar State Constraints

Additional state constraints are treated following the ideas of [20,21]. Adding two scalar state constraints to problem formulation (1) results in

$$\begin{aligned} &\min I(w,q) \qquad \text{subject to } c(w,q) = 0 \\ &h_1(w,q) = 0, \qquad h_2(w,q) = 0 \end{aligned} \tag{8}$$

Typically, in practical applications, additional constraints are of the form

$$h_i(w,q) \geq 0$$

and represent the validity region of the model or the design construction. For the sake of simplicity in presentation we discuss only equality constraints, which is the case in our applications. Inequality constraints can be handled by an active set strategy, which is trivial for scalar inequalities. More difficult constraints, for example, constraints in contact problems, which are not scalar valued, are out of the scope of this paper.

One approximate Newton step for necessary conditions of problem (8) corresponding to an rSQP method is derived from the following system of equations:

$$\begin{pmatrix} 0 & 0 & \left(\frac{\partial h_1}{\partial w}\right)^\top & \left(\frac{\partial h_2}{\partial w}\right)^\top & A^\top \\ 0 & B & \left(\frac{\partial h_1}{\partial q}\right)^\top & \left(\frac{\partial h_2}{\partial q}\right)^\top & \left(\frac{\partial c}{\partial q}\right)^\top \\ \frac{\partial h_1}{\partial w} & \frac{\partial h_1}{\partial q} & 0 & 0 & 0 \\ \frac{\partial h_2}{\partial w} & \frac{\partial h_2}{\partial q} & 0 & 0 & 0 \\ A & \frac{\partial c}{\partial q} & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta w \\ \Delta q \\ \Delta\mu_1 \\ \Delta\mu_2 \\ \Delta\lambda \end{pmatrix} = \begin{pmatrix} -\nabla_w L \\ -\nabla_q L \\ -h_1 \\ -h_2 \\ -c \end{pmatrix} \tag{9}$$

where μ_1 and μ_2 are the Lagrange multipliers corresponding to the additional state constraints $h_1(w,q)$ and $h_2(w,q)$, respectively.

A. Partial Reduction of the Problem

In the following we use Gaussian elimination for partial reduction of the system of Eqs. (9) by eliminating the variables Δw and $\Delta\lambda$ from the system. To do that, we consider the last equation and solve for

$$\Delta w = A^{-1}\left(-c(w,q) - \frac{\partial c}{\partial q}\Delta q\right)$$

A substitution of this for Δw in the fourth equation results in

$$\left(\frac{\partial h_2}{\partial q} - \frac{\partial h_2}{\partial w}A^{-1}\frac{\partial c}{\partial q}\right)\Delta q = -h_2 + \frac{\partial h_2}{\partial w}A^{-1}c$$

If we define

$$g_{h_2} := \left(\frac{\partial h_2}{\partial q} - \frac{\partial h_2}{\partial w}A^{-1}\frac{\partial c}{\partial q}\right)^\top = \begin{bmatrix} -A^{-1}\frac{\partial c}{\partial q} \\ I \end{bmatrix}^\top \begin{bmatrix} \nabla_w h_2 \\ \nabla_q h_2 \end{bmatrix}$$

as the reduced gradient of $h_2(w,q)$, then we get

$$g_{h_2}^\top \Delta q = -h_2 + \frac{\partial h_2}{\partial w}A^{-1}c \tag{10}$$

Similarly, a substitution of Δw in the third equation results in

$$g_{h_1}^\top \Delta q = -h_1 + \frac{\partial h_1}{\partial w}A^{-1}c \tag{11}$$

where g_{h_1} is the reduced gradient of $h_1(w,q)$. Then we consider the first equation and solve for

$$\Delta\lambda = A^{-\top}\left[-\nabla_w L - \left(\frac{\partial h_1}{\partial w}\right)^\top \Delta\mu_1 - \left(\frac{\partial h_2}{\partial w}\right)^\top \Delta\mu_2\right]$$

A substitution of this for $\Delta\lambda$ in the second equation results in

$$\begin{aligned} B\Delta q + \left[\left(\frac{\partial h_1}{\partial q}\right)^\top - \left(\frac{\partial c}{\partial q}\right)^\top A^{-\top}\left(\frac{\partial h_1}{\partial w}\right)^\top\right]\Delta\mu_1 + \left[\left(\frac{\partial h_2}{\partial q}\right)^\top \right. \\ \left. - \left(\frac{\partial c}{\partial q}\right)^\top A^{-\top}\left(\frac{\partial h_2}{\partial w}\right)^\top\right]\Delta\mu_2 = -\nabla_q L + \left(\frac{\partial c}{\partial q}\right)^\top A^{-\top}\nabla_w L \end{aligned}$$

This can be written in terms of the reduced gradients as

$$B\Delta q + g_{h_1}\Delta\mu_1 + g_{h_2}\Delta\mu_2 = -\nabla_q L + \left(\frac{\partial c}{\partial q}\right)^\top A^{-\top}\nabla_w L \tag{12}$$

Writing Eqs. (10–12) in matrix-vector notation results in the reduced system

$$\begin{pmatrix} B & g_{h_1} & g_{h_2} \\ g_{h_1}^\top & 0 & 0 \\ g_{h_2}^\top & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta q \\ \Delta\mu_1 \\ \Delta\mu_2 \end{pmatrix} = \begin{pmatrix} -\nabla_q L + \left(\frac{\partial c}{\partial q}\right)^\top A^{-\top}\nabla_w L \\ -h_1 + \frac{\partial h_1}{\partial w}A^{-1}c \\ -h_2 + \frac{\partial h_2}{\partial w}A^{-1}c \end{pmatrix} \tag{13}$$

B. Solution Strategy of the Constrained Problem

We use the aforementioned reduction strategy to solve problem (8) using simultaneous pseudo-time-stepping. The preconditioned pseudotime embedded nonstationary system to be solved reads as

$$\begin{pmatrix} 0 & 0 & \left(\frac{\partial h_1}{\partial w}\right)^\top & \left(\frac{\partial h_2}{\partial w}\right)^\top & A^\top \\ 0 & B & \left(\frac{\partial h_1}{\partial q}\right)^\top & \left(\frac{\partial h_2}{\partial q}\right)^\top & \left(\frac{\partial c}{\partial q}\right)^\top \\ \frac{\partial h_1}{\partial w} & \frac{\partial h_1}{\partial q} & 0 & 0 & 0 \\ \frac{\partial h_2}{\partial w} & \frac{\partial h_2}{\partial q} & 0 & 0 & 0 \\ A & \frac{\partial c}{\partial q} & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \dot{w} \\ \dot{q} \\ \dot{\mu_1} \\ \dot{\mu_2} \\ \dot{\lambda} \end{pmatrix} = \begin{pmatrix} -\nabla_w L \\ -\nabla_q L \\ -h_1 \\ -h_2 \\ -c \end{pmatrix} \tag{14}$$

For the solution of this problem, we first solve system (7) for the design velocity $\dot{q}$. This design velocity is then projected onto the tangent space of the state constraints through the solution of the following QP:

$$\begin{aligned} &\min \frac{1}{2}\dot{q}^\top B\dot{q} + g^\top \dot{q} \\ &\text{subject to } g_{h_1}^\top \dot{q} = -h_1(w,q) + \frac{\partial h_1}{\partial w}A^{-1}c \\ &\qquad g_{h_2}^\top \dot{q} = -h_2(w,q) + \frac{\partial h_2}{\partial w}A^{-1}c \end{aligned} \tag{15}$$

where $g = [\nabla_q L - (\frac{\partial c}{\partial q})^\top A^{-\top}\nabla_w L]$ is the reduced gradient of the Lagrangian. It is to note that the first-order necessary conditions for solving (15) leads to the same system as in (13) with the unknowns replaced by $\dot{q}$ and $\dot{\mu}_1, \dot{\mu}_2$ [which are the Lagrange multipliers for the system (15)], respectively.

Therefore, this reduction can be interpreted as a projection of the design velocity from Eq. (7) towards the linearized state constraints $h_1(w,q)$ and $h_2(w,q)$, thus resembling dynamic projection strategies onto invariants as in [26]. For the construction of the reduced gradients g , g_{h_1}, and g_{h_2}, one has to solve one adjoint problem (approximately) for g and one each for g_{h_1} and g_{h_2}.

C. Back Projection

Because of nonlinearity of the problem, there is some deviation from the additional state constraints. Therefore, we use the following correction strategy in each optimization step to avoid this deviation: We minimize the distance between the point q_0 and the manifold $\mathcal{S}$ of constant lift and constant pitching moment $\mathcal{S} = \{q | h_1[w(q)] = l_0^1, h_2[w(q)] = l_0^2\}$. This is done by solving ideally the problem

$$\begin{aligned} &\min \frac{1}{2}||q - q_0||^2 \qquad \text{subject to } h_1[w(q)] - l_0^1 = 0 \\ &\qquad\qquad h_2[w(q)] - l_0^2 = 0 \end{aligned} \tag{16}$$

We use one step of a generalized Gauss–Newton method to solve this problem. Because the stiffness matrix of the flow equations is approximated by A when forming the reduced gradients g_{h_1} and g_{h_2}, we compute the step Δq from

$$\begin{aligned} &\min \frac{1}{2}||\Delta q||^2 \qquad \text{subject to } g_{h_1}^\top \Delta q = -h_1(w,q) + \frac{\partial h_1}{\partial w}A^{-1}c \\ &\qquad\qquad g_{h_2}^\top \Delta q = -h_2(w,q) + \frac{\partial h_2}{\partial w}A^{-1}c \end{aligned} \tag{17}$$

The necessary optimality conditions to solve this problem will lead to the system of equations

$$\begin{pmatrix} I & g_{h_1} & g_{h_2} \\ g_{h_1}^\top & 0 & 0 \\ g_{h_2}^\top & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta q \\ \nu_1 \\ \nu_2 \end{pmatrix} = \begin{pmatrix} 0 \\ -\left[h_1(w,q) - \frac{\partial h_1}{\partial w}A^{-1}c\right] \\ -\left[h_2(w,q) - \frac{\partial h_2}{\partial w}A^{-1}c\right] \end{pmatrix}$$

where ν_1 and ν_2 are the Lagrange multipliers for the system (17). Solving this system gives Δq, which is used to get the corrected step given by

$$q^{k+1} = q_0^{k+1} - \Delta q \tag{18}$$

The overall algorithm reads as follows:

Algorithm 1: The simultaneous pseudo-time-stepping for the preconditioned system.

0) Set $k := 0$; start at some initial guess w_0, λ_0, and q_0.

1) Compute λ^{k+1} marching one step in time for the adjoint equations.

2) Compute sensitivities using state and adjoint solutions.

3) Determine some approximation B_k of the projected Hessian of the Lagrangian.

4) Solve the quadratic subproblem Eq. (15) to get $\dot{q}$.

5) March in time one step for the design equation as follows: $q_0^{k+1} = q^k + \Delta t \cdot \dot{q}$.

6) Use the correction step Eq. (18) for the new step.

7) Compute w^{k+1} marching one step in time for the state equations.

8) Set $k := k + 1$; go to step 1 until convergence.

In the current implementation, $A^{-1}c$ in the right-hand side of the constraint in Eq. (15) is approximated by (scaled) $\dot{w}$ value from the previous iteration as it is updated only at step 7 of the preceding algorithm. Δt in step 5 of the algorithm is the minimum time-step length of the forward solver. Instead of the exact reduced gradient of the Lagrangian [e.g., in Eq. (15)], we use an approximation of it. It is a one-shot method because we perform one time step for each design update.

IV. Detailed Equations of the Aerodynamic Shape Optimization Problem

In this section we explain briefly the state, costate, and design equations represented in Eqs. (4) for the shape optimization problem in 3-D (as in [22]). Similar equations for 2-D can be found in [12,20].

State equations: Because we are interested in the steady flow, a proper approach for numerical modeling is to integrate the unsteady Euler equations in time until a steady state is reached. These equations in Cartesian (x, y, z) coordinates for a 3-D flow domain Ω with boundaries $\partial\Omega$ can be written in integral form as

$$\frac{\partial}{\partial t}\int_\Omega w \, \mathrm{d}\Omega + \int_{\partial\Omega} F \cdot \boldsymbol{n} \, \mathrm{d}s = 0 \tag{19}$$

where

$$w := \begin{bmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho u_3 \\ \rho E \end{bmatrix}$$

$$F := [f_1, f_2, f_3], \quad \text{with} \quad f_i := \begin{bmatrix} \rho u_i \\ \rho u_i u_1 + p\delta_{i1} \\ \rho u_i u_2 + p\delta_{i2} \\ \rho u_i u_3 + p\delta_{i3} \\ \rho u_i H \end{bmatrix}$$

where δ_{ij} is the Kronecker delta. For a perfect gas, the pressure and total enthalpy are given by

$$p = (\gamma - 1)\rho\left\{E - \frac{1}{2}\left(u_1^2 + u_2^2 + u_3^2\right)\right\}, \qquad H = E + \frac{p}{\rho}$$

respectively. The boundary conditions used to solve these equations

are zero normal velocity on the solid wall B_s, and the far-field boundary B_F is treated by considering the incoming and outgoing characteristics based on the 1-D Riemann invariants.

The cost function that we choose in the present optimization problem is for drag reduction. Hence, the cost function reads as

$$I(w,q) := C_D = C_A \cos\alpha + C_N \sin\alpha = \frac{1}{S_{\text{ref}}} \iint_{B_S} C_p (S_x \cos\alpha + S_y \sin\alpha)\, \mathrm{d}s \tag{20}$$

where S_x and S_y define projected surface areas and C_p is defined by

$$C_p := \frac{2(p - p_\infty)}{\gamma M_\infty^2 p_\infty} \tag{21}$$

The additional state constraint of constant lift is given by

$$h(w,q) := C_L - C_{L0} = C_N \cos\alpha - C_A \sin\alpha - C_{L0} = \frac{1}{S_{\text{ref}}} \iint_{B_S} C_p (S_y \cos\alpha - S_x \sin\alpha)\, \mathrm{d}s - C_{L0} \tag{22}$$

where C_{L0} is a given constant value.

Costate equations: The problem of derivative computation is solved by using the continuous adjoint approach [1]. The drag minimization is carried out at a constant lift. The adjoint Euler equations and corresponding boundary conditions for this problem are described next.

The costate or adjoint Euler equations are given by

$$\frac{\partial}{\partial t} \int_\Omega \lambda \, \mathrm{d}\Omega + \int_{\partial\Omega} \bar{F} \cdot \boldsymbol{n} \, \mathrm{d}s = 0 \tag{23}$$

where the vector λ contains the components of the adjoint variable and $\bar{F}$ is the matrix of adjoint flux densities, defined as

$$\lambda := \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \lambda_5 \end{bmatrix}, \qquad \bar{F} := \left[\left(\frac{\partial f_1}{\partial w}\right)^T \lambda, \left(\frac{\partial f_2}{\partial w}\right)^T \lambda, \left(\frac{\partial f_3}{\partial w}\right)^T \lambda \right]$$

The boundary conditions for the adjoint Euler equations on the solid body are given by

$$S_x\lambda_2 + S_y\lambda_3 + S_z\lambda_4 = \text{RHS} \quad \text{on } B_S \tag{24}$$

For the aforementioned cost function, the right-hand side (RHS) is given by

$$\text{RHS} = -\frac{2}{\gamma M_\infty^2 S_{\text{ref}}} \{S_x \cos\alpha + S_y \sin\alpha\} \tag{25}$$

and for the state constraint of constant lift, the RHS is given by

$$\text{RHS} = -\frac{2}{\gamma M_\infty^2 S_{\text{ref}}} \{S_y \cos\alpha - S_x \sin\alpha\} \tag{26}$$

The treatment of the far-field boundary is based on freestream conditions.

Design equation: For the design equation, we need an expression for the derivative of the Lagrangian with respect to the design parameters. All the computations are carried out in a generalized coordinate system. Therefore, a transformation is used to transform the physical (x, y, z) domain to the computational (ξ_1, ξ_2, ξ_3) domain. In the computational domain, the variation of the objective function is given by (see, for example, in [7])

$$\delta I = \frac{1}{S_{\text{ref}}} \iint_{B_S} C_p (\delta S_x \cos\alpha + \delta S_y \sin\alpha)\, \mathrm{d}\xi_1\, \mathrm{d}\xi_2 + \int_D \lambda^T \frac{\partial}{\partial \xi_i} (\delta Q_{ij} f_j)\, \mathrm{d}\xi_k \tag{27}$$

where the transformation matrices Q_{ij} are given by

$$Q = \tilde{J} K^{-1}, \quad \text{with} \quad \tilde{J} = \det(K), \qquad K_{i,j} = \left[\frac{\partial x_i}{\partial \xi_j}\right]$$

Here the second integral on the right-hand side is a field integral and this might be costly to compute. As an alternative, surface formulation of [27,28] is used for computations.

The details of discretization of the equations, gradient computation, surface parameterization, and grid perturbation strategies can be found in [22] (for 3-D) and in [12] (for 2-D).

V. Numerical Results and Discussion

The optimization method is applied to test cases in 2-D as well as in 3-D. The complete optimization cycle is performed under the optimization platform SynapsPointerPro [29]. The FLOWer code [30,31] of the German Aerospace Center (DLR) is used for solving the forward and adjoint equations. This code is very robust (in multigrid modes) and tested in many applications including flow computations for the company AIRBUS-Germany. It is enhanced, with minor modification, for the one-shot optimization method.

The design equation is integrated in time using an explicit Euler scheme and the state and costate equations are integrated in time using a five-stage Runge–Kutta scheme. Therefore, the time steps used for the three sets of equations are not the same. In the current implementation of FLOWer the time steps are not same even in each discretization cell because they are determined independently according to the local stability. However, this has no effect on the final solution at steady state.

One of the main issues of using this kind of preconditioned pseudo-time-stepping is the approximation of the reduced Hessian. As we have shown in [10], reduced Hessian approximation based on the current gradient and parameter update informations is good enough for this problem class, so we use the same update strategy here. We define $s_k := (q_{k+1} - q_k)$ and $z_k := (g_{k+1} - g_k)$, where k represents the iteration number. Then the reduced Hessian approximation is determined based on the sign of the product $(z_k^T s_k)$. If the sign is positive, the reduced Hessian is approximated by

$$B_k = \bar{\beta} \frac{z_k^T s_k}{z_k^T z_k} \delta_{ij} \tag{28}$$

where $\bar{\beta}$ is a constant. Otherwise, it is approximated by $\beta\delta_{ij}$, where β is another constant. Additionally, we impose upper and lower limits on the factor so that

$$\beta_{\min} < \bar{\beta} \frac{z_k^T s_k}{z_k^T z_k} < \beta_{\max}$$

This prevents the optimizer from taking too small or too large steps. The constants $\beta_{\min}$ and $\beta_{\max}$ can be chosen, for example, depending on the accuracy achieved in one time step by the forward and adjoint solver.

A. Applications in 2-D

The optimization method is applied to test cases of an RAE2822 airfoil for drag reduction with constant lift and constant pitching moment together with geometric constraint of constant thickness. The flow conditions are described at Mach number 0.73 and angle of incidence 2 deg. The physical domain is discretized using an algebraically generated (193 × 33) C-grid. The airfoil is decomposed into thickness and camberline distributions for parameterization purposes. The parameters corresponding to the thickness have not been changed during optimization to satisfy the geometric constraint.

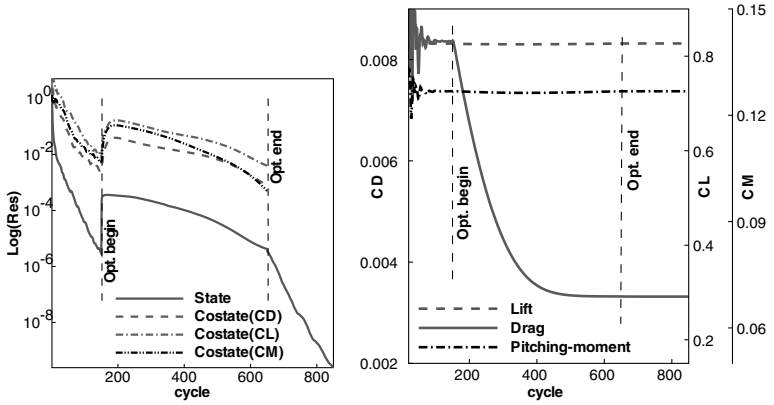

Fig. 1 Convergence history of the optimization problem of case 1.

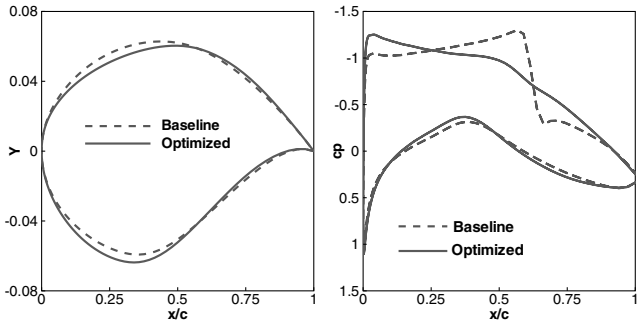

Fig. 2 Comparison of initial and final airfoils and surface pressure distributions of case 1.

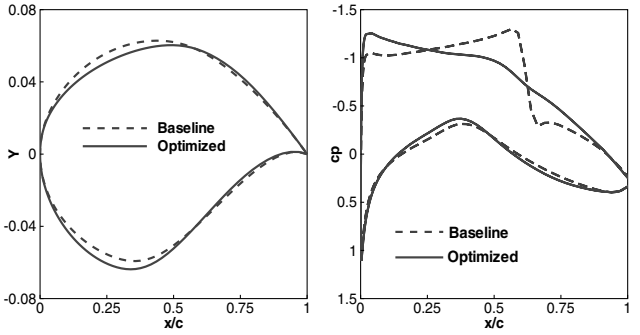

Fig. 4 Comparison of initial and final airfoils and surface pressure distributions of case 2.

The camberline has been parameterized by 21 Hicks–Henne [2] parameters.

The forward and adjoint solutions are computed using the multigrid solvers in FLOWer. Optimization iterations start with the initial solutions obtained after 150 time steps of the state equations w_0 and of the costate equations λ_0. The optimization iterations stopped when $||\Delta q||_\infty < 0.0008$. After the convergence of optimization iterations, another 200 time steps are performed for the state equation on the optimized geometry to reduce its residual further so that the force coefficients and surface pressure distribution can be compared with those obtained by other methods.

In *case 1, drag reduction with constant lift and constant pitching moment with 21 design parameters*, 500 iterations are required for the convergence of the optimization problem. Optimization convergence histories are presented in Fig. 1. Figure 2 presents a comparison of initial and final airfoils and surface pressure distributions. As we see in this figure, the optimization results in a shock-free airfoil in this case. This is due to the fact that in the inviscid transonic regime, a major amount of drag is caused by the shock. Hence, drag reduction results a shock-free airfoil. Table 1 presents a comparison of the force coefficients for baseline and optimized geometries. The drag reduction is 60.30%, the lift is well maintained with an increase by only 0.016%, and the pitching moment is also well maintained with a decrease by 0.016%.

The forward solver requires about 350 time steps to produce "well" converged solution. The total number of time steps required for this optimization is $(150 + 500 + 200)$ (before optimization, during optimization, and after optimization, respectively) for the state solver, and three times $(150 + 500)$ (before optimization, and during optimization, respectively) for the adjoint solver. These together are approximately eight times that of the forward simulation runs. The effort to solve the design equation and gradient computations is almost negligible in comparison to state or costate solutions. Additionally, the simultaneous pseudotime method needs a new grid, obtained by grid perturbation, after each optimization iteration. Additional time is required to write the output after every iteration and read the same before each iteration, as the iterations start with solution values from the previous iteration. If we add on everything, the total effort is less than 10 times that of the forward simulation runs.

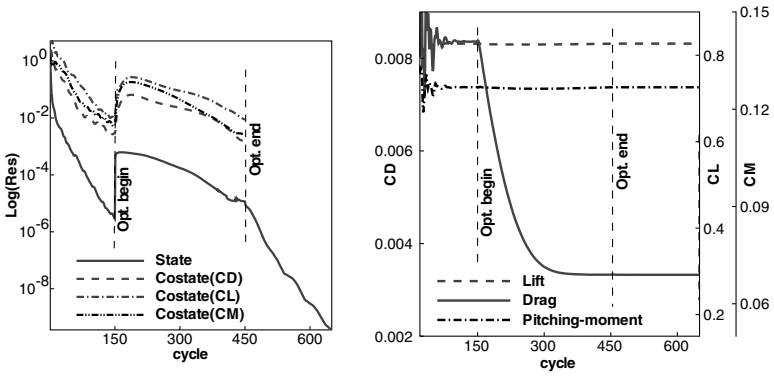

Fig. 3 Convergence history of the optimization problem of case 2.

Table 1 Comparison of number of iterations and force coefficients for baseline and optimized airfoil for different number of design parameters (on 193 × 33 grid)

Geometry	Iterations	C_D	ΔC_D	C_L	ΔC_L	C_M	ΔC_M
Baseline	——	0.836150E − 02	——	0.826810	——	0.12679	——
Opt. (case 1)	500	0.331930E − 02	60.30%	0.826940	−0.016%	0.12677	0.016%
Opt. (case 2)	300	0.332590E − 02	60.22%	0.826770	0.005%	0.12678	0.008%

In *case 2, drag reduction with constant lift and constant pitching moment with 40 design parameters*, we use 40 design parameters to parameterize the camberline of the airfoil. The convergence of the optimization problem is faster, requiring only 300 iterations. Faster convergence of the method can be explained from the physics of the problem as in [20] and repeated here. The optimization problem deals with drag reduction with additional state constraints. As mentioned, in inviscid transonic regime, major drag is caused by the shock jump. The task of the optimizer is to change the design parameters in such a way that the drag disappears. In case of finer parameterization, a small change in parameters has more effect on the shock than that of a coarser parameterization. To achieve the same effect in coarser parameterization, one has to have larger change in the design parameters, which causes constraint violation. That is why in this particular problem class finer parameterization leads to faster convergence.

Because we use continuous adjoint method, the cost of gradient computation is independent of number of design variables. Therefore, it is advantageous to use finer design space in the context of one-shot pseudotime method. Optimization convergence histories are presented in Fig. 3. Figure 4 presents a comparison of the initial and final airfoils and surface pressure distributions. A comparison of force coefficients for the baseline and optimized geometries are presented in Table 1. The drag reduction is 60.22%, the lift is well maintained by a decrease of only 0.005%, and the pitching moment is also well maintained by a decrease of only 0.008%. The optimized force coefficients are almost the same in both cases. Figure 5 presents a comparison of airfoils and surface pressure distributions obtained in case 1 and case 2. There is no noticeable difference in them as well. However, the number of optimization iterations are little more than half of that of case 1 and the total effort in this case is less than seven times that of the forward simulation runs.

In *case 3, drag reduction with constant lift and constant pitching moment with 40 design parameters on* 321×57 *grid*, we study the fine grid optimization with computational grid around the airfoil being of size 321×57. Here also the optimization is started with initial state and costate solutions (w_0, λ_0) obtained after 150 time steps. The optimization requires 800 iterations to converge. After the convergence of the optimization, another 200 time steps are carried out for the state solution to achieve sufficiently accurate force coefficients. Figure 6 presents the optimization convergence histories of this case. Figure 7 presents a comparison of the initial and final airfoils and surface pressure distributions. A comparison of the force coefficients for the baseline and optimized geometries are presented in Table 2. The drag reduction is 61.80, which is a bit more than the last two cases, the lift is again well maintained with an increase by only 0.004%, and the pitching moment is also well maintained with an increase by only 0.02%. In this case the steady-state forward solution is reached by 450 multigrid iterations and thus, if we add on everything, the total effort of the optimization problem is less than 11 forward simulation runs.

Optimization of the same geometry has been carried out in [19] using traditional gradient methods. The drag reduction there is about 60.00% and the constraint violations are about 0.1%. The total effort required there consist of 40 forward simulation runs and 27 adjoint runs. In comparison to that, the current effort is a reduction of about 83%.

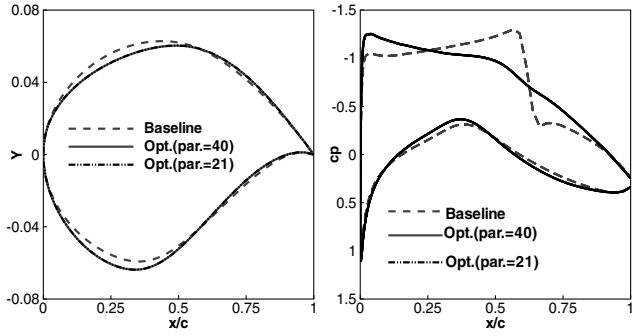

Fig. 5 Comparison of initial and final airfoils and surface pressure distributions of case 1 and case 2.

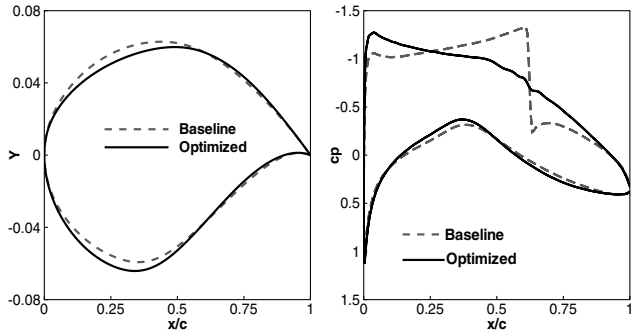

Fig. 7 Comparison of initial and final airfoils and surface pressure distributions of case 3.

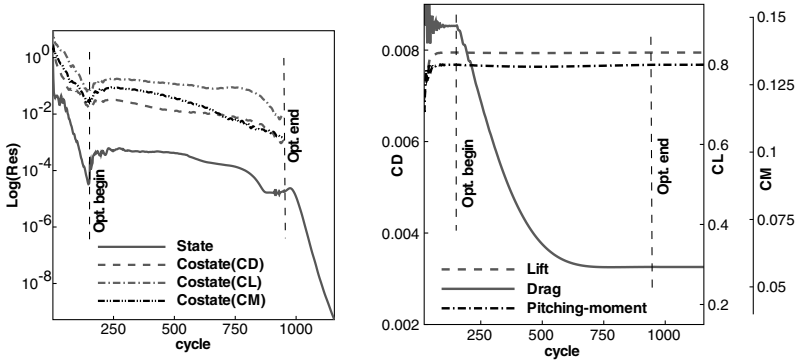

Fig. 6 Convergence history of the optimization problem of case 3.

Table 2 Comparison of force coefficients for baseline and optimized airfoil in case 3 computation (on 321 × 57 grid)

Geometry	Iterations	C_D	ΔC_D	C_L	ΔC_L	C_M	ΔC_M
Baseline	——	0.852910E − 02	——	0.829500	——	0.12920	——
Opt. (case 3)	800	0.325770E − 02	61.80%	0.829530	−0.004%	0.12923	−0.02%

Table 3 Comparison of force coefficients for baseline and optimized wing

Geometry	Iterations	C_D	ΔC_D	C_L	ΔC_L
Baseline	——	0.972837E − 02	——	0.120000E + 00	——
Optimized	300	0.806945E − 02	17.05%	0.120019E + 00	−0.016%

B. Application in 3-D

In this case we apply our method to a 3-D problem of wing optimization at supersonic Mach number 2.0 and angle of incidence 3.22949 deg. The physical domain is discretized into a grid of C-H topology consisting of $(97 \times 17 \times 25)$ grid points. The wing is decomposed into thickness, camberline, and twist distributions for parameterization purposes as described in [22]. The resulting wing is constructed by linear lofting of the modified wing sections. The thickness deformation has been based on B-splines, which set free the range and the chordwise position of the maximum thickness, the leading-edge radius, and the trailing-edge angle at eight wing sections. The position of these sections are chosen according to the spanwise distribution of the geometrical constraints on maximum thickness. The camberline has been modified by adding 10 Hicks–Henne functions at eight wing sections. The twist distribution has been described by a Bezier curve defined by 10 nodes. The center of rotation for the twist has been set at the leading-edge of the wing. A total of 122 design variables are used to change the twist, the thickness, and the camberline at specific wing sections.

The optimization in *case 4, drag reduction with constant lift for a Supersonic Cruise Transport aircraft wing*, is performed with initial state solution w_0 obtained after 150 time steps and initial costate solutions λ_0 obtained after 200 time steps. The optimization needs 300 iterations to converge. The convergence history of the optimization iterations are presented in Fig. 8. Table 3 presents a comparison of the baseline and optimized force coefficients. The drag reduction in this case is 17.05% and the lift is well maintained with an increase by only 0.016%. Figure 9 presents a comparison of the baseline and optimized pressure distributions (left) and geometries (right) at four different spanwise sections. From pressure distributions in the same figure, we see that the pressure peak is reduced almost all over the wing. Figure 10 presents the initial and final Mach contours on the wing.

Optimization of the same geometry in same flow conditions has been carried out using our one-shot method and using the same FLOWer code in [22]. There the additional state constraint has been treated "indirectly" by adding it to the objective function and solving the reduced unconstrained problem. The constant lift is maintained by changing the angle of incidence. There the optimization required 650 iterations to converge. The drag reduction has been 12.59%. In the current computation we could achieve a drag reduction of 17.05% and the total number of optimization iterations required are

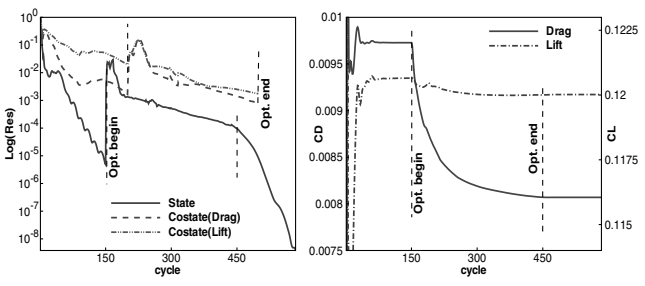

Fig. 8 Convergence history of the optimization of case 4.

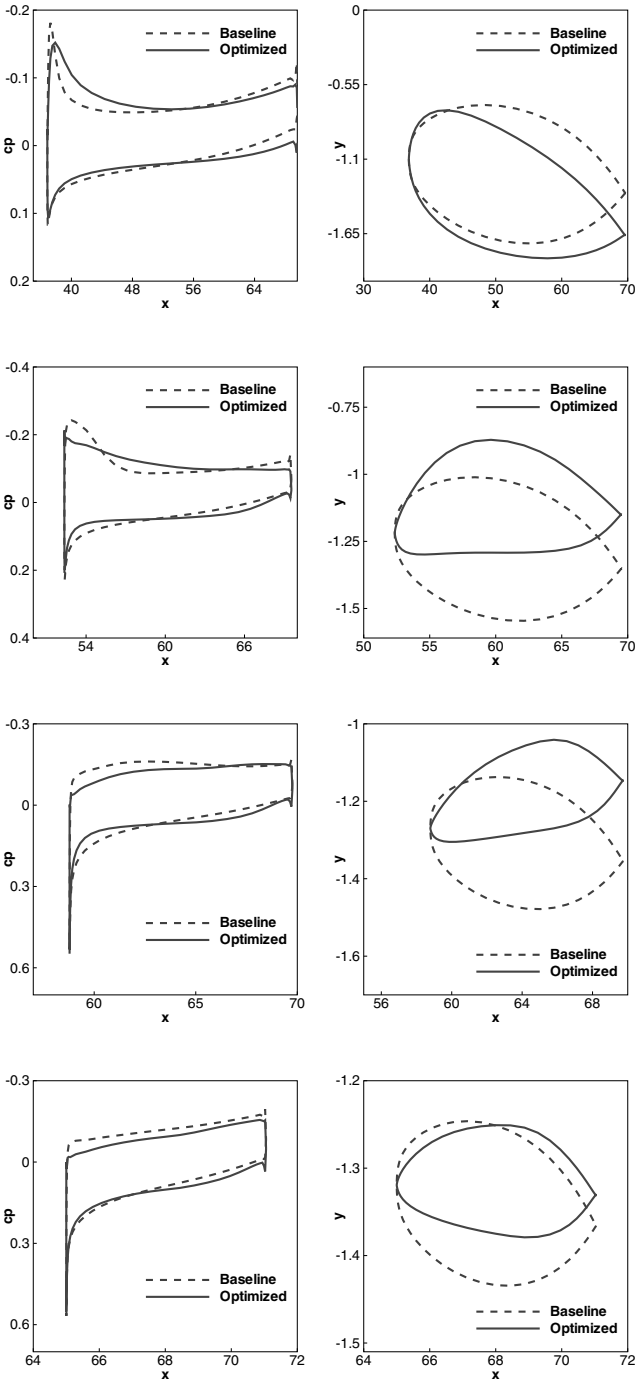

Fig. 9 Comparison of surface pressure distributions (left) and geometries (right) at four sections (from top to bottom) at $\eta = 0.24$, 0.49, 0.70, 0.90 of the wing.

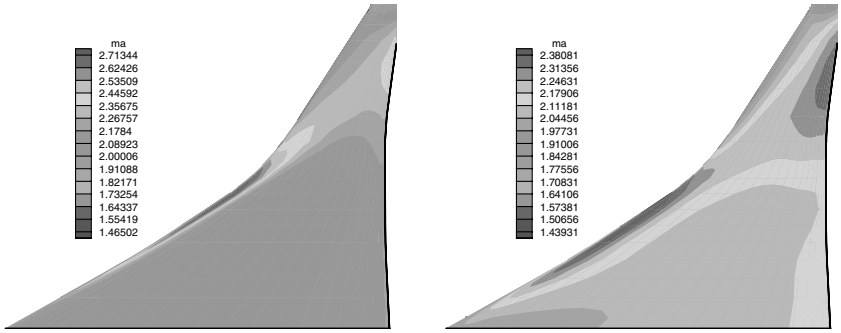

Fig. 10 Baseline (left) and optimized (right) Mach contours on the wing.

300. This confirms that direct treatment of additional state constraints is advantageous. However, in the current computation we need to solve an additional adjoint equation for the lift constraint. If we add on everything, the total computational effort in this case is less than six times that of forward simulation runs. In [32] optimization of the same geometry has been carried out using traditional gradient methods together with indirect treatment of the state constraint (as in [22]). The total effort required there consist of 39 forward simulation runs plus five adjoint runs and the drag reduction has been 12.58%. In comparison to that, the current effort is a reduction of 86%.

VI. Conclusions

Problems in aerodynamic shape optimization with additional state constraints have been solved using simultaneous pseudo-time-stepping. The preconditioned pseudostationary state, costate, and design equations are integrated simultaneously in time until a steady state is reached. The preconditioner used in this study is motivated by a continuous reinterpretation of rSQP methods. The solution strategy consists of projection of unconstrained design velocity onto the tangent space of the constraints. Computational examples, at different discretization levels using different design parameter spaces, in 2-D as well as in 3-D are provided. This kind of "direct" treatment of constraints leads to efficiency in the solution process and to a better optimum of the optimization problem than those of "indirect" treatments. Faster convergence is achieved in finer design parameter space using the current pseudo-time-stepping method. The overall cost of computation is approximately 7–11 times (depending on the grid size and the number of design parameters) that of the forward simulation runs for two additional state constraints in 2-D and less than six times that of the forward simulation runs for a single additional state constraint in 3-D.

Analog to the results presented in our earlier works using one-shot pseudo-time-stepping method, in the convergence histories of all the computations reported here, a linear convergence with respect to the objective is observed. Starting from an almost feasible initial guess, each iteration step in the primal, adjoint, and design space is so small that the process truly reflects a continuous behavior. Each integration step of the preconditioned pseudotime formulation is contracting enough so that no additional globalization strategy is necessary. The preconditioning in the design space gives short enough integration steps (corrected by back projection) so that the iterates "surf" along an almost feasible manifold towards optimality.

Acknowledgments

I would like to thank the German Aerospace Center (DLR), Braunschweig, for giving access to the FLOWer code. Thanks are due to the anonymous referees for their comments and suggestions on this work.

References

[1] Jameson, A., "Aerodynamic Design via Control Theory," *Journal of Scientific Computing*, Vol. 3, No. 3, 1988, pp. 23–260.

[2] Hicks, R. M., and Henne, P. A., "Wing Design by Numerical Optimization," *Journal of Aircraft*, Vol. 15, No. 7, 1978, pp. 407–412.

[3] Gauger, N. R., and Brezillon, J., "Aerodynamic Shape Optimization Using Adjoint Method," *Journal of the Aeronautical Society of India*, Vol. 54, No. 3, 2002, pp. 247–254.

[4] Jameson, A., "Automatic Design of Transonic Airfoils to Reduce Shock Induced Pressure Drag," Mechanical and Aerospace Engineering, Princeton University Report 1881, Princeton, NJ, Feb. 1990.

[5] Jameson, A., "Optimum Aerodynamic Design Using CFD and Control Theory," AIAA Paper 95-1729-CP, June 1995.

[6] Reuther, J., and Jameson, A., "Aerodynamic Shape Optimization of Wing and Wing-Body Configurations Using Control Theory," AIAA Paper 95-0123, Jan. 1995.

[7] Reuther, J., Jameson, A., Farmer, J., Martinelli, L., and Saunders, D., "Aerodynamic Shape Optimization of Complex Aircraft Configurations via an Adjoint Formulation," AIAA Paper 96-0094, Jan. 1996.

[8] Anderson, W. K., and Venkatakrishnan, V., "Aerodynamic Design Optimization on Unstructured Grids with a Continuous Adjoint Formulation," AIAA Paper 97-0643, 1997.

[9] Hazra, S. B., and Schulz, V., "Simultaneous Pseudo-Timestepping for PDE-Model Based Optimization Problems," *BIT Numerical Mathematics*, Vol. 44, No. 3, 2004, pp. 457–472.

[10] Hazra, S. B., "Reduced Hessian Updates in Simultaneous Pseudo-Timestepping for Aerodynamic Shape Optimization," AIAA Paper 2006-933, Jan. 2006.

[11] Hazra, S. B., "An Efficient Method for Aerodynamic Shape Optimization," AIAA Paper 2004-4628, 2004.

[12] Hazra, S. B., Schulz, V., Brezillon, J., and Gauger, N. R., "Aerodynamic Shape Optimization Using Simultaneous Pseudo-Timestepping," *Journal of Computational Physics*, Vol. 204, No. 1, 2005, pp. 46–64.

[13] Ta'asan, S., Kuruvila, G., and Salas, M. D., "Aerodynamic Design and Optimization in One Shot," AIAA Paper 92-0025, Jan. 1992.

[14] Hou, G. W., Taylor, A. C., III, Mani, S. V., and Newman, P. A., "Formulation for Simultaneous Aerodynamic Analysis and Design Optimization," NASA CR-201036, 1993.

[15] Feng, D., and Pulliam, T. H., "An All-at-Once Reduced SQP Scheme for Aerodynamic Design and Optimization," Research Institute for Advanced Computer Science Tech. Rept. 95.19, Moffet Field, CA, Oct. 1995.

[16] Giles, M. B., and Pierce, N. A., "An Introduction to the Adjoint Approach to Design," *Flow, Turbulence and Combustion*, Vol. 65, No. 3/4, 2000, pp. 393–415.

[17] Fletcher, R., *Practical Methods of Optimization*, John Wiley & Sons, New York, 1987.

[18] Kim, H. J., Sasaki, D., Obayahi, S., and Nakahashi, K., "Aerodynamic Optimization of Supersonic Transport Wing Using Unstructured Adjoint Method," *AIAA Journal*, Vol. 39, No. 6, 2001, pp. 1011–1020.

[19] Brezillon, J., and Gauger, N. R., "2D and 3D Aerodynamic Shape Optimization Using the Adjoint Approach," *Aerospace Science and Technology*, Vol. 8/8, No. 8, 2004, pp. 715–727.

[20] Hazra, S. B., and Schulz, V., "Simultaneous Pseudo-Timestepping for Aerodynamic Shape Optimization Problems with State Constraints," *SIAM Journal on Scientific Computing*, Vol. 28, No. 3, 2006, pp. 1078–1099.

[21] Hazra, S. B., and Schulz, V., "Simultaneous Pseudo-Timestepping for State Constrained Optimization Problems in Aerodynamics," *Real-Time PDE-Constrained Optimization*, edited by L. Biegler, O. Ghattas, M. Heinkenschloss, D. Keyes, and B. van Bloemen Waanders, SIAM, Philadelphia (to be published).

[22] Hazra, S. B., Schulz, V., and Brezillon, J., "Simultaneous Pseudo-Timestepping for 3D Aerodynamic Shape Optimization," Dept. of Mathematics/Computer Science, Univ. of Trier Preprint No. 05-2, Trier, Germany, 2005.

[23] Battermann, A., and Sachs, E. W., "Block Preconditioners for KKT Systems in PDE-Governed Optimal Control Problems," *Fast Solution of Discretized Optimization Problems*, edited by K. H. Hoffmann, R. H. W. Hoppe, and V. Schulz, Birkhäuser Verlag, Basel, 2001, pp. 1–18.

[24] Battermann, A., and Heinkenschloss, M., "Preconditioners for Karush-Kuhn-Tucker Systems Arising in the Optimal Control of Distributed Systems," *Optimal Control of PDE, Vorau 1996*, edited by W. Desch, F. Kappel, and K. Kunisch, Birkhäuser Verlag, Basel, 1996, pp. 15–32.

[25] Biros, G., and Ghattas, O., "Parallel Lagrange-Newton-Krylov-Schur Methods for PDE-Constrained Optimization. Part 1: The Krylov-Schur Solver," *SIAM Journal on Scientific Computing*, Vol. 27, No. 2, 2005, pp. 687–713.

[26] Schulz, V. H., Bock, H. G., and Steinbach, M. C., "Exploiting Invariants in the Numerical Solution of Multipoint Boundary Value Problems for DAE," *SIAM Journal on Scientific Computing*, Vol. 19, No. 2, 1998, pp. 440–467.

[27] Weinerfelt, P., and Enoksson, O., "Numerical Methods for Aerodynamic Optimization," *Computational Fluid Dynamics Journal*, Vol. 9, 2001, pp. 758–765.

[28] Weinerfelt, P., and Enoksson, O., "Aerodynamic Shape Optimization and Parallel Computing Applied to Industrial Problems," *Proceedings of the Parallel CFD 2000 Conference*, edited by C. B. Jenssen, Elsevier Science, Ltd., North-Holland, 2001.

[29] Fromman, O., *SynapsPointerPro v2.50*, Synaps Ingenieure Gesellschaft mbH, Bremen, Germany, 2002.

[30] Kroll, N., Rossow, C. C., Becker, K., and Thiele, F., "The MEGAFLOW—A Numerical Flow Simulation System," *21st ICAS Symposium, Melbourne, Australia*, Paper 98-2.7.4, 1998.

[31] Kroll, N., Rossow, C. C., Becker, K., and Thiele, F., "The MEGAFLOW project," *Aerospace Science and Technology*, Vol. 4, June 2000, pp. 223–237.

[32] Brezillon, J., "Application of the Adjoint Technique with the Optimization Framework Synaps Pointer Pro," *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, Vol. 89, edited by N. Kroll and J. Fassbender, Springer–Verlag, Berlin, 2002.

J. Samareh
Associate Editor