Aerodynamic Optimization
Using the Flow Sensitivity Approach

by

Peterson Wong

A thesis submitted in conformity with the requirements
for the degree of Master of Applied Science
Graduate Department of Aerospace Science and Engineering
University of Toronto

Abstract

Aerodynamic Optimization
Using the Flow Sensitivity Approach

Peterson Wong
Master of Applied Science
Graduate Department of Aerospace Science and Engineering
University of Toronto
2001

The flow sensitivity approach for aerodynamic optimization using computational fluid dynamics is studied. The flow sensitivity equation is solved using matrix-free GMRES in conjunction with first-order Jacobian matrix preconditioning. The accuracy of the gradients calculated using the flow sensitivities is assessed by comparison with finite-difference gradients. Several design problems are studied, including inverse design, drag minimization, and lift enhancement. A wide range of constraints are applied to the design cases, such as specified lift, lower limit on pressure, and geometry constraints. The matrix-free sensitivity approach provides a robust and accurate method for gradient calculation in aerodynamic design optimization.

Acknowledgements

I would like to thank Professor David Zingg for his guidance and supervision throughout my research. His many useful ideas and suggestions helped me a lot in my research.

I would like to thank Marian Nemec for his help in my research. He has excellent knowledge in this topic and helped me a lot with many useful discussions, from CYCLONE to optimization software.

I would like to thank Jason Lassaline for his help with computer problems. He is so genius that he knows everything about computers. Thanks to Mike Sullivan for setting up a nice working environment for us. I would also like to thank Luis Manzano for his help. He has been helping me with everything from ARC1D to LaTeX.

It is always a pleasure to work with John Gatsis, for his humour and his excellence in math, and Peter Blaser, Ian Sigal as well as Edward Wehner.

Finally I would like to thank my girlfriend, my parents and my brother for their support, patience and understanding.

This research is supported by the Natural Sciences and Engineering Research Council of Canada.

Contents

List of Figures

List of Tables

List of Symbols

$\hat{E}, \hat{F}$	convective flux
$\hat{H}$	pressure source term
$\mathcal{Q}$	continuous flow variables
Q	discrete flow variables
$\hat{S}$	viscous flux
D	dissipation

ρ	density
a	speed of sound
e	energy
p	pressure
u, v	Cartesian velocities
H	enthalpy
M	Mach number
Pr	Prandtl number
Pr_t	turbulent Prandtl number
Re	Reynolds number
tr	laminar to turbulent transition
U, V	contravariant velocities
V_n	normal velocity component
V_t	tangential velocity component
μ	laminar viscosity
μ_t	turbulent viscosity
C_p	pressure coefficient
C_l	lift coefficient
C_d	drag coefficient

ξ, η	curvilinear coordinates
J^{-1}	metric Jacobian

$\mathcal{F}$	continuous steady flow equations
R	discrete steady flow equations
J	objective function
C	constraint
g	design variables
dJ/dg	gradients
dQ/dg	discrete flow sensitivities
ψ	adjoint variables
ϵ	finite-difference stepsize
ϵ_m	machine zero
A_e	exact flow Jacobian
A_2	second order approximate flow Jacobian
A_1	first order approximate flow Jacobian
σ	first order Jacobian weight
n_{dv}	number of design variables
w_T	thickness constraint weights

BFGS	Broyden-Fletcher-Goldfarb-Shanno (optimizer)
KSOPT	constrained optimizer using the KS function
DFP	Davidon-Fletcher-Powell
KS	Kreisselmeier-Steinhauser
RCM	Reverse Cuthill-McKee reordering
GMRES	generalized minimum residual method
BFILU	block-fill incomplete LU factorization

Chapter 1

Introduction

1.1 Motivation

Recent development of robust, accurate and efficient flow solvers, with the rapid growth of computer power, have motivated research in CFD optimization. The significance of this work is the potential to design an optimum wing according to user specifications in an automated manner. Current research areas include lift enhancement, drag minimization and preventing separation. The goal is to develop a reliable and systematic design procedure which is suitable for industrial applications.

1.2 Literature Review

Hicks, Murman and Vanderplaats [1] developed an airfoil design algorithm more than twenty years ago. They used a gradient-based method with gradients calculated using finite-differences and a simple transonic small-disturbance fluid flow model. The topic of aerodynamic design was further pioneered by Jameson [2], who first applied control theory to aerodynamic optimization problems.

Design optimization can be done using several methods. Gunzburger [3] summarized different approaches that can be used to solve the optimization problem. Frank and Shubin [4] compared gradient-based methods for the one-dimensional nozzle design problem. The inverse problem was solved and the optimizer successfully determined the nozzle shape that matched the target velocity distribution.

For gradient-based methods, the reliability of the optimization depends on the

accuracy of the calculated gradients. There are several ways to calculate the gradients. The adjoint method is most efficient because the computational cost to calculate the gradients does not depend on the number of design variables used. The adjoint gradients can be calculated using a discrete or a continuous formulation. Anderson and Venkatakrishnan [5] solved the inverse problem and the drag minimization problem for a two-dimensional airfoil. The continuous adjoint formulation was used on unstructured grids. Preconditioned GMRES coupled with an implicit multigrid solver was used to solve the adjoint equations and the KSOPT code [6] was used as the optimizer. Nielsen and Anderson [7] used the discrete adjoint method to perform aerodynamic optimization for multi-element airfoils and wings. They used the three-dimensional Reynolds-averaged Navier-Stokes equations on unstructured grids. Elliott and Peraire [8] solved constrained and multi-point shape optimization problems for three-dimensional bodies using a subspace BFGS method.

An alternative method of calculating the gradients is to use the flow sensitivity equations. This method is less efficient than the adjoint approach because the number of sensitivity equations to be solved depends on the number of design parameters used. However, there are some advantages to be considered:

- The sensitivity approach is advantageous when flow constraints are applied to the design problem because flow sensitivities can be reused.
- The flow sensitivities may give the designer more insight about how the physical system responds to different design variables.
- The sensitivity equations can be solved using the matrix-free GMRES approach, which reduces memory use in the procedure, and can avoid difficulties in linearization.

Turgeon and Pelletier [9] applied the continuous sensitivity approach to optimal design in mixed convection. The geometry of a mixed convection system was designed to achieve the best heat transfer property.

The third method to calculate the gradients is to use finite differences. This method is most computationally expensive. However, it is accurate and can be easily extended to multidisciplinary and general engineering designs. Newman et al. [10] studied multidisciplinary sensitivity derivatives using complex variables. An aero-structural problem was analyzed. Anderson et al. [11] performed the sensitivity analysis for the Navier-Stokes equations on unstructured meshes using complex variables. They reported that

complex sensitivity derivatives are not prone to errors caused by subtractive cancellation. However, the complex approach suffers from twice the memory requirement and three times the algorithm running time.

1.3 Objectives

The objective of the project is to implement the flow sensitivity approach for aerodynamic optimization problems. We focus on some advantages of the sensitivity approach to assess the feasibility of the method in optimization. The algorithm is tested on a one-dimensional nozzle problem and various two-dimensional aerodynamic design problems with constraints.

Chapter 2

The Nozzle Design Problem

This chapter presents several gradient-based methods: finite difference, flow sensitivity, discrete and continuous adjoint. One-dimensional nozzle design as in reference [12] is considered. The inverse design problem is studied for both subsonic and transonic cases, with two and ten design variables.

2.1 Governing Equations

The governing equations are the quasi-one-dimensional Euler equations.

$$\frac{\partial \hat{\mathcal{Q}}}{\partial t} + \frac{\partial \hat{E}}{\partial x} = \hat{H} \tag{2.1}$$

The steady solution is obtained by setting the time derivative term to zero. The steady flow equations used in the optimization problem are

$$\mathcal{F}(\hat{\mathcal{Q}}) = -\frac{\partial \hat{E}}{\partial x} + \hat{H} = 0 \tag{2.2}$$

where $\mathcal{F}$ denotes the 3×1 vector of continuous flow equations. The conservative variables are given by

$$\mathcal{Q} = \begin{bmatrix} \rho \\ \rho u \\ e \end{bmatrix} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{2.3}$$

and

$$\hat{\mathcal{Q}} = \mathcal{Q}S \tag{2.4}$$

where S is the cross-sectional area of the nozzle. The inviscid fluxes are given by

$$E = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ u(e+p) \end{bmatrix} = \begin{bmatrix} q_2 \\ \frac{q_2^2}{q_1} + (\gamma - 1)(q_3 - \frac{1}{2}\frac{q_2^2}{q_1}) \\ \frac{q_2}{q_1}[q_3 + (\gamma - 1)(q_3 - \frac{1}{2}\frac{q_2^2}{q_1})] \end{bmatrix} \tag{2.5}$$

and

$$\hat{E} = ES \tag{2.6}$$

The source terms are given by

$$H = \begin{bmatrix} 0 \\ p \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ (\gamma - 1)(q_3 - \frac{1}{2}\frac{q_2^2}{q_1}) \\ 0 \end{bmatrix} \tag{2.7}$$

and

$$\hat{H} = H\frac{dS}{dx} \tag{2.8}$$

2.2 Design Variables

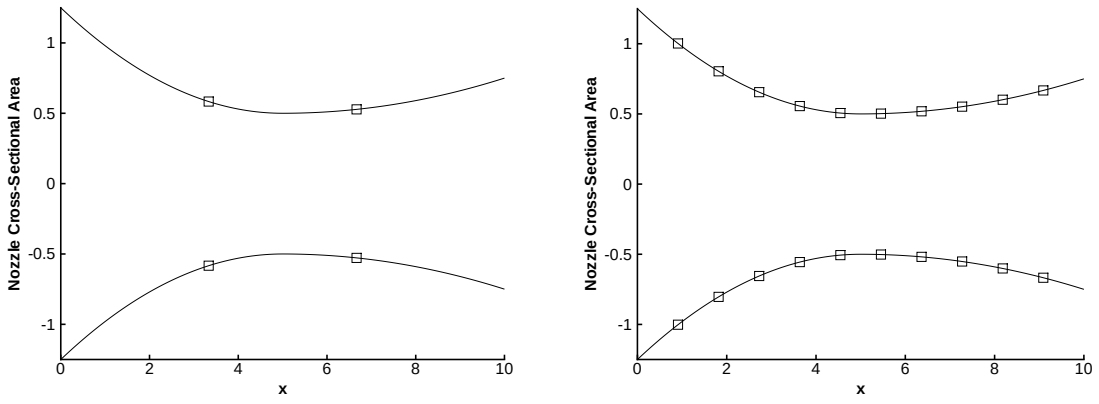
The nozzle shape is controlled by the design variables g. Figure 2.1 shows the control points of the nozzle design problem. The nozzle geometry is formed by cubic spline interpolation through the control points. Hence, the nozzle area S is a function of the design variables. As a result, $\mathcal{F}$ is a function of both the conservative flow variables and the design variables.

2.3 Flow Solver

We follow the methology of reference [13] to solve the flow equations. The spatial derivatives are discretized on a uniform grid. Spatial discretization is based on second-order centered differences with a nonlinear artificial dissipation model. This leads to a system of nonlinear algebraic equations

$$R(Q, g) = 0 \tag{2.9}$$

where R is the $3(j_{max}) \times 1$ vector of discrete flow equations, Q is the $3(j_{max}) \times 1$ vector of discrete flow variables, and j_{max} is the number of grid nodes. The discrete flow equations are solved using an implicit time marching method to steady state.



(a) 2 design variables

(b) 10 design variables

Figure 2.1: Design variables of the nozzle problem.

2.4 Gradient Calculation

We use a gradient-based optimization algorithm to search for the optimum design. This section summarizes four different methods for gradient calculations [3].

2.4.1 Black Box/Finite-Difference Approach

The easiest method for gradient calculation is the finite-difference approach. No modification is needed to the flow solver. The gradients of the objective function are obtained by perturbing the design variables in the design space. Using forward differences, the gradient with respect to the first design variable is

$$\frac{dJ}{dg_1} = \frac{J(Q(g_1+\epsilon, g_2, \cdots), g_1+\epsilon, g_2, \cdots) - J(Q(g_1, g_2, \cdots), g_1, g_2, \cdots)}{\epsilon} \tag{2.10}$$

The objective function $J(Q, g)$ is a function of both the flow variables and the design variables. The flow variables are functions of the design variables because they must satisfy the flow equations $R(Q, g) = 0$. The cost of gradient calculation using the finite-difference approach is M flow solves, where M is the number of design variables.

2.4.2 Flow Sensitivity Approach

Another method to calculate the gradients is the flow sensitivity approach, using the following equation:

$$\frac{dJ}{dg} = \frac{\partial J}{\partial g} + \frac{\partial J}{\partial Q}\frac{dQ}{dg} \tag{2.11}$$

The partial derivatives $\partial J/\partial g$ and $\partial J/\partial Q$ are usually known. There are two methods to calculate the flow sensitivities dQ/dg. The first way is to use finite differences. This method is expensive with the cost of M flow solves. The other method is to differentiate the flow equations $R(Q, g) = 0$ to obtain the sensitivity equations

$$\frac{\partial R}{\partial Q}\frac{dQ}{dg} = -\frac{\partial R}{\partial g} \tag{2.12}$$

The flow sensitivities can be solved from the sensitivity equations. The cost of gradient calculation using this approach is M linear solves.

2.4.3 Discrete Adjoint

The gradients can also be calculated using the adjoint equations. The adjoint equations are formed using the Lagrange multiplier method. The Lagrangian is formed from the objective function and the flow equations:

$$L(Q, g, \psi) = J(Q, g) - \psi^T R(Q, g) \tag{2.13}$$

The adjoint equation is obtained by setting the derivative of the Lagrangian with respect to the flow variables to zero:

$$\frac{\partial R}{\partial Q}^T \psi = \frac{\partial J}{\partial Q}^T \tag{2.14}$$

After solving for the adjoint variables ψ from the above equation, the gradients are then calculated using

$$\frac{dJ}{dg} = \frac{\partial J}{\partial g} - \psi^T \frac{\partial R}{\partial g} \tag{2.15}$$

The cost of calculating the gradients using the discrete adjoint approach is one linear solve, independent of the number of design variables.

2.4.4 Continuous Adjoint

The continuous adjoint equations are formed by differentiation of the continuous Lagrangian with respect to the flow variables. The derivation is shown in Appendix B. See reference [14] for a comparison between the continuous and discrete adjoint approaches.

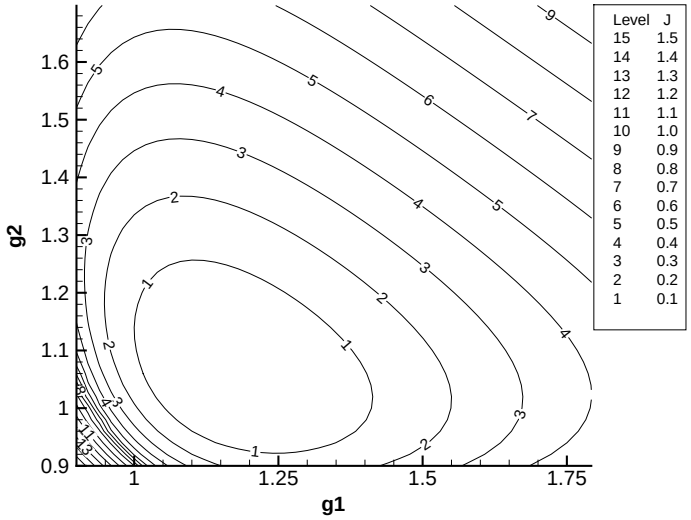


Figure 2.2: Contour plot of the objective function for subsonic case.

2.5 Optimization Results

The optimization problem is solved using KSOPT [6]. The inverse design problem is considered. A target velocity distribution is specified, and the optimizer searches for the nozzle shape that matches the velocity distribution. The objective function is given by

$$J(Q) = \frac{1}{2} \sum_j \left(u_j - u_j^* \right)^2 \tag{2.16}$$

Figures 2.2 and 2.3 show contour plots of the objective functions in the design space for two-design-variable subsonic and transonic cases. The objective function is more sensitive to the second design variable in the transonic case.

The partial derivatives of the objective function with respect to the flow variables at node j are

$$\left(\frac{\partial J}{\partial Q} \right)_j = \left[\begin{array}{ccc} \frac{\partial J}{\partial q_1} & \frac{\partial J}{\partial q_2} & \frac{\partial J}{\partial q_3} \end{array} \right]_j \tag{2.17}$$

with

$$\begin{aligned} \frac{\partial J}{\partial q_1} &= \left(\frac{q_2}{q_1} - u_j^* \right) \left(-\frac{q_2}{q_1^2} \right) \\ \frac{\partial J}{\partial q_2} &= \left(\frac{q_2}{q_1} - u_j^* \right) \left(\frac{1}{q_1} \right) \\ \frac{\partial J}{\partial q_3} &= 0 \end{aligned} \tag{2.18}$$

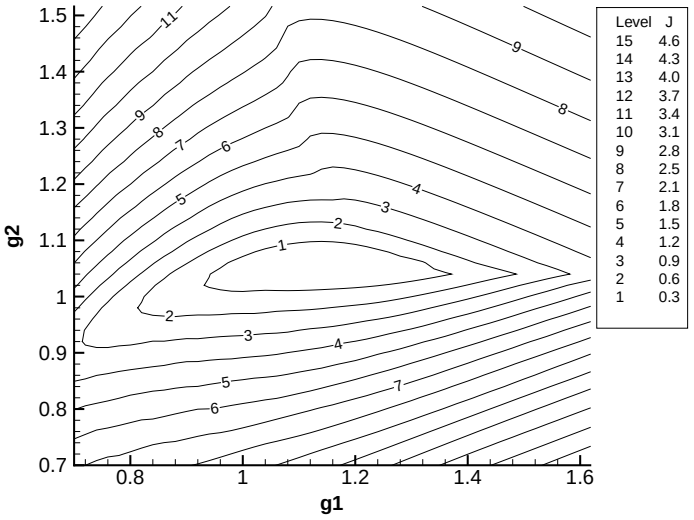


Figure 2.3: Contour plot of the objective function for transonic case.

The partial derivatives of the objective function with respect to all flow variables are

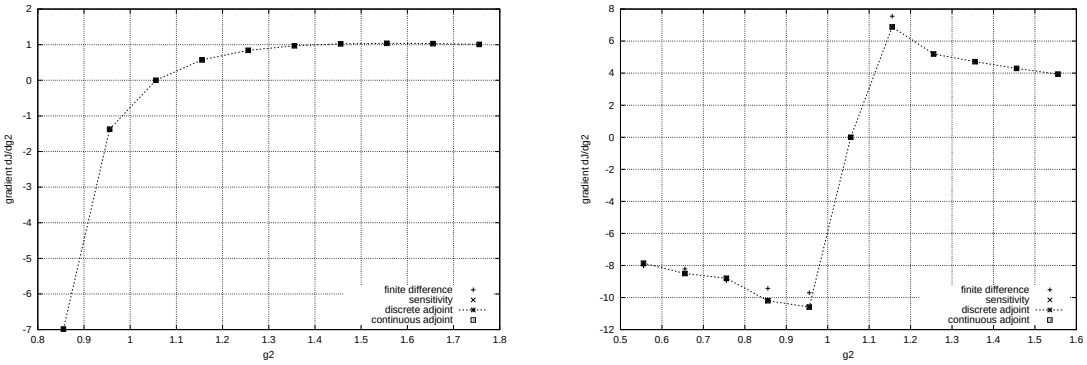
$$\frac{\partial J}{\partial Q} = \left[\begin{array}{ccc} \left(\frac{\partial J}{\partial Q}\right)_1 & \cdots & \left(\frac{\partial J}{\partial Q}\right)_{j_{max}} \end{array} \right] \tag{2.19}$$

In this case, the objective function does not depend explicitly on the design variables. The partial derivatives of the objective function with respect to the design variables are thus zero:

$$\frac{\partial J}{\partial g} = 0 \tag{2.20}$$

The flow Jacobian $\partial R/\partial Q$ can be calculated using equation A.1 in Appendix A. The derivatives of the flow equations with respect to the design variables $\partial R/\partial g$ are calculated using finite differences. The gradients of the objective function are calculated using the methods described in section 2.4. Figure 2.4 shows the calculated gradients with respect to g_2. For the subsonic case, the gradients calculated using the four methods agree well. For the transonic case, the gradient calculated using finite differences is slightly different from those calculated using the other methods. This difference produces no significant effect on either optimizer accuracy and performance.

Figures 2.5 to 2.8 show the results of the inverse design problem using the flow sensitivity approach. The nozzle geometries that correspond to the target velocities are calculated accurately. The numbers of gradient and function evaluations used for each case are recorded in table 2.1.



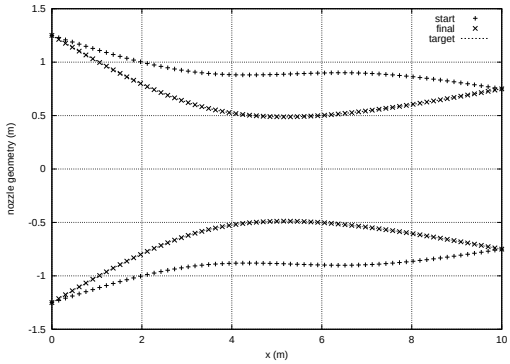
(a) Subsonic

(b) Transonic

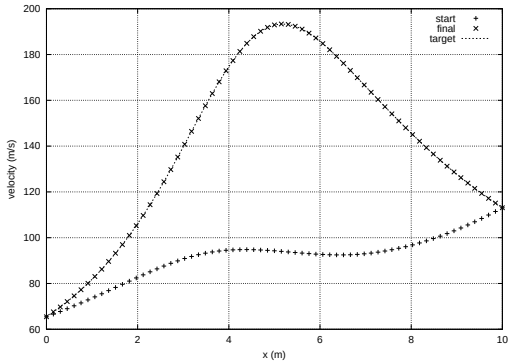
Figure 2.4: Comparing gradients using different methods (g_1 fixed at 1.166667).

Case		Design Variables	Gradient Evaluations	Function Evaluations
1	Subsonic	2	7	50
2	Transonic	2	10	72
3	Subsonic	10	22	145
4	Transonic	10	39	283

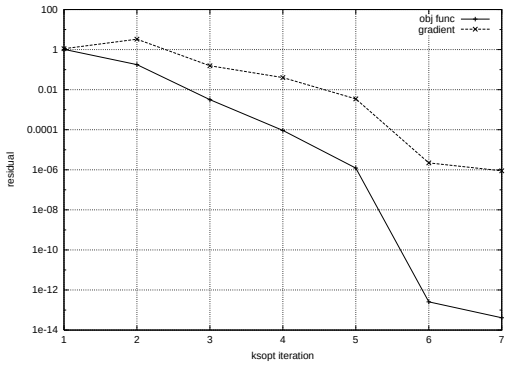
Table 2.1: Gradient and function evaluations for the four design cases.



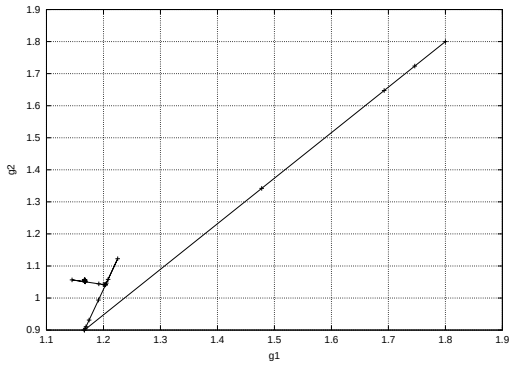
(a) Nozzle geometry



(b) Velocity distribution

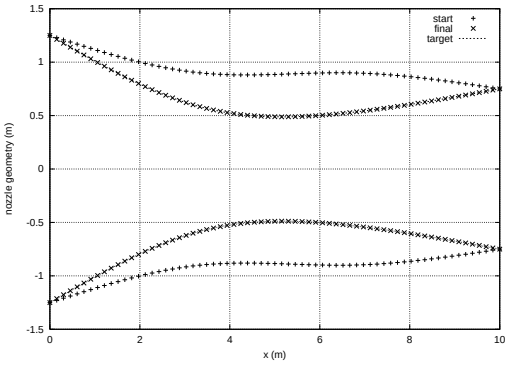


(c) Design history

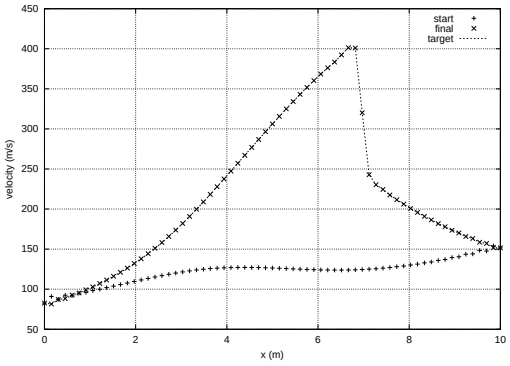


(d) Design path

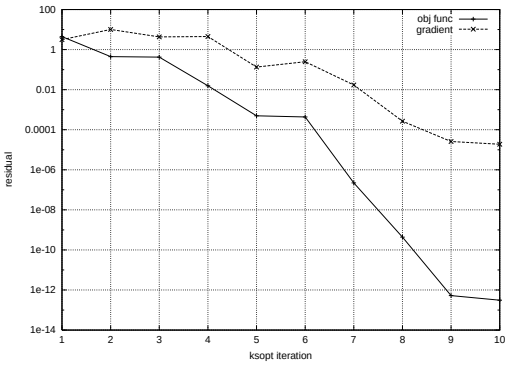
Figure 2.5: Subsonic inverse design with 2 design variables.



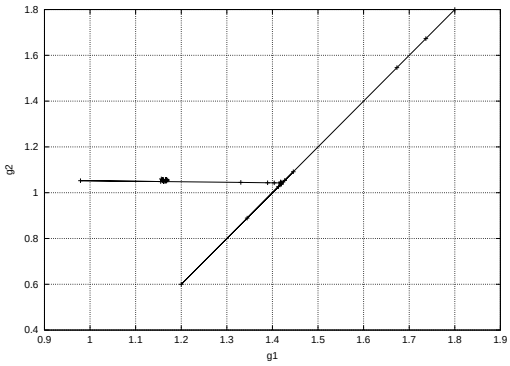
(a) Nozzle geometry



(b) Velocity distribution

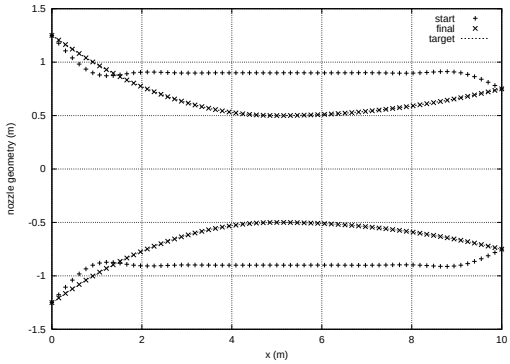


(c) Design history

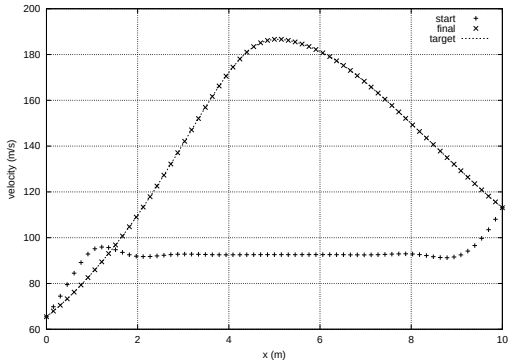


(d) Design path

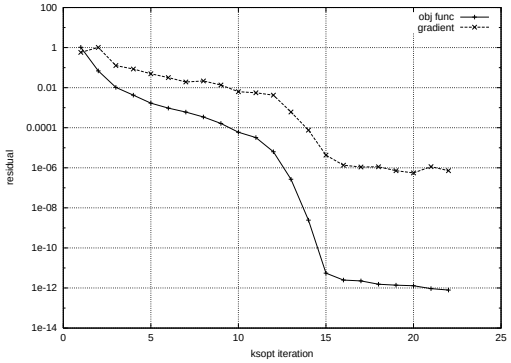
Figure 2.6: Transonic inverse design with 2 design variables.



(a) Nozzle geometry

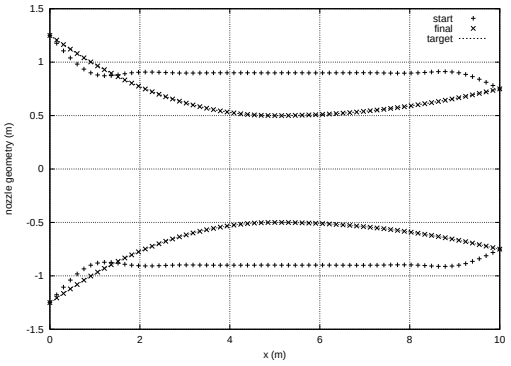


(b) Velocity distribution

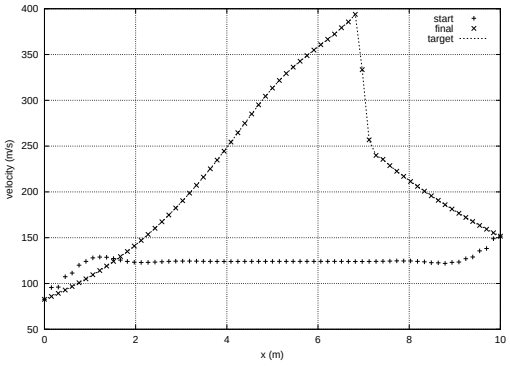


(c) Design history

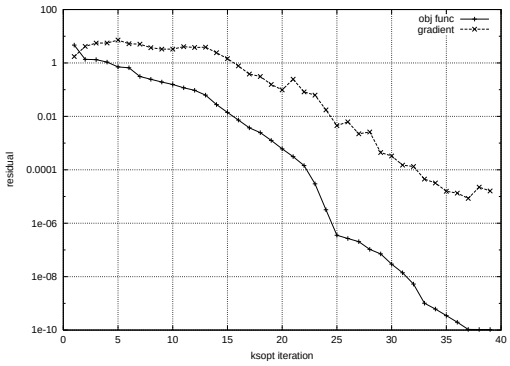
Figure 2.7: Subsonic inverse design with 10 design variables.



(a) Nozzle geometry



(b) Velocity distribution



(c) Design history

Figure 2.8: Transonic inverse design with 10 design variables.

Chapter 3

The Airfoil Design Problem

3.1 Overview

The airfoil design problem consists of design objectives, constraints, design variables and the flow equations. Some examples of design objectives are drag reduction and increased lift-to-drag ratio. Design objectives are written as objective functions. Examples of design constraints include target lift specification, pressure constraints and geometry constraints. Constraints can be posed as equality constraints or inequality constraints. The design variables are the airfoil geometry and the angle of attack. We modify the design variables to satisfy the design objectives while satisfying the constraints. A set of flow equations are used to govern the physics of the problem. The design problem is to find a set of design variables that minimize the objective function and satisfies the flow equations and constraints.

3.1.1 Governing Equations

The governing equations are the steady two-dimensional thin-layer Navier-Stokes equations. We use the Baldwin and Lomax turbulence model [15] to simulate turbulent flow. The governing equations in generalized curvilinear coordinates are

$$\partial_\xi \hat{E} + \partial_\eta \hat{F} = M_\infty Re^{-1} \partial_\eta \hat{S} \tag{3.1}$$

The inviscid fluxes $\hat{E}$ and $\hat{F}$ are functions of the flow variables $\mathcal{Q}$

$$\mathcal{Q} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ e \end{bmatrix}, \hat{E} = J^{-1} \begin{bmatrix} \rho U \\ \rho u U + \xi_x p \\ \rho v U + \xi_y p \\ U(e+p) \end{bmatrix}, \hat{F} = J^{-1} \begin{bmatrix} \rho V \\ \rho u V + \eta_x p \\ \rho v V + \eta_y p \\ V(e+p) \end{bmatrix} \tag{3.2}$$

where the contravariant velocities are

$$U = \xi_x u + \xi_y v, V = \eta_x u + \eta_y v \tag{3.3}$$

The Jacobian of transformation is given by

$$J^{-1} = x_\xi y_\eta - x_\eta y_\xi \tag{3.4}$$

Pressure is related to the flow variables by the equation of state

$$p = (\gamma - 1)\left(e - \frac{1}{2}\rho(u^2 + v^2)\right) \tag{3.5}$$

where γ is the ratio of specific heats. The viscous flux is

$$\hat{S} = J^{-1} \begin{bmatrix} 0 \\ \eta_x m_1 + \eta_y m_2 \\ \eta_x m_2 + \eta_y m_3 \\ \eta_x(u m_1 + v m_2 + m_4) + \eta_y(u m_2 + v m_3 + m_5) \end{bmatrix} \tag{3.6}$$

where

$$\begin{aligned} m_1 &= (\mu + \mu_t)(4\eta_x u_\eta - 2\eta_y v_\eta)/3 \\ m_2 &= (\mu + \mu_t)(\eta_y u_\eta + \eta_x v_\eta) \\ m_3 &= (\mu + \mu_t)(-2\eta_x u_\eta + 4\eta_y v_\eta)/3 \\ m_4 &= (\mu Pr^{-1} + \mu_t Pr_t^{-1})(\gamma - 1)^{-1}\eta_x \partial_\eta(a^2) \\ m_5 &= (\mu Pr^{-1} + \mu_t Pr_t^{-1})(\gamma - 1)^{-1}\eta_y \partial_\eta(a^2) \end{aligned} \tag{3.7}$$

The speed of sound is $a = \sqrt{\gamma p/\rho}$. The dynamic viscosity is μ and turbulent viscosity μ_t. The Reynolds number is Re and Prandtl number Pr.

	Inviscid	Viscous
Grid size	245 × 41	257 × 57
Cluster points:		
Upper LE	0.001	0.0005
Upper TE	0.01	1.0E-3
Lower TE	0.01	1.0E-3
Number of grid lines:		
Upper	90	100
Lower	90	100
Wakecut	33	29
Chords to far field	12	12
η_{max}	41	57
Offwall spacing	2.0E-3	2.0E-6

Table 3.1: AMBER2D parameters for grid generation.

3.1.2 Flow Solver

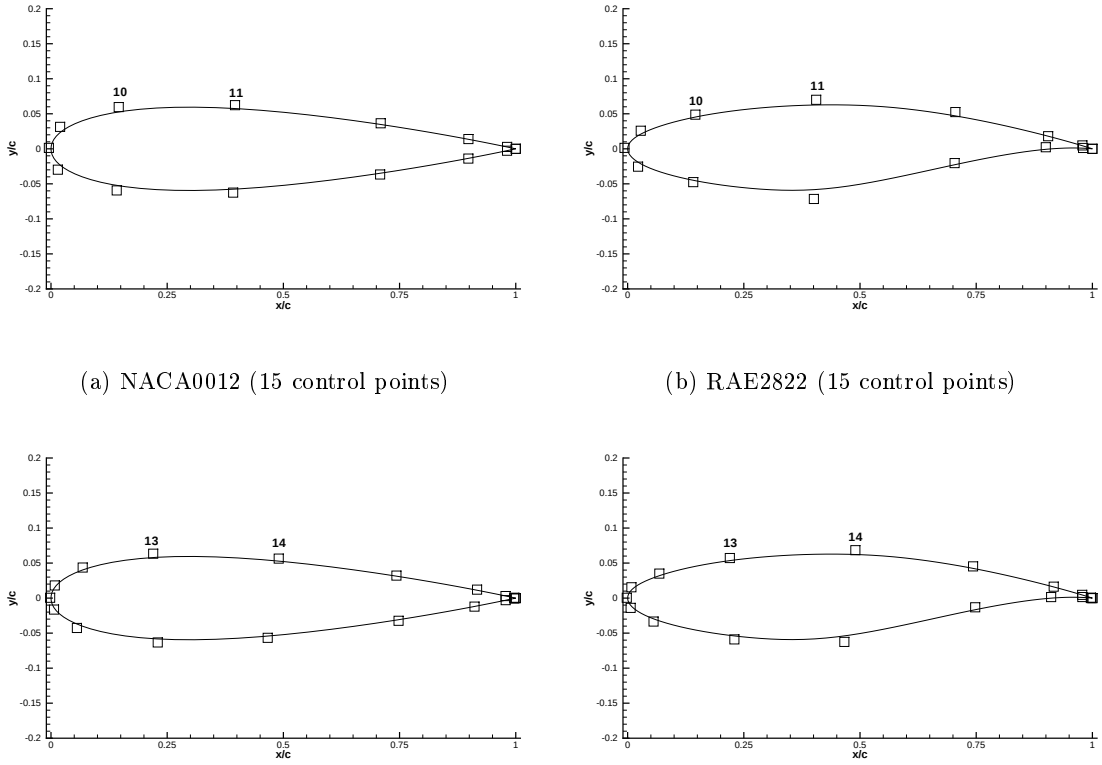
We use the flow solver ARC2D [13] to solve the flow equations. This is a finite-difference scheme on a structured grid. Spatial discretization is based on second order centered differences with a nonlinear dissipation model. Spatial discretization leads to a system of nonlinear algebraic equations

$$R(Q, g) = 0 \tag{3.8}$$

where R is the $4(j_{max}k_{max})\times 1$ vector of discrete flow equations, and Q is the $4(j_{max}k_{max})\times 1$ vector of discrete flow variables. The approximate factorization technique is used to solve the flow equations. A far-field circulation correction is applied.

3.1.3 Grid Generation

The flow equations are discretized on a C-topology structured grid. We use the grid generator AMBER2D [16] to generate a grid from a set of airfoil body points. Table 3.1 summarizes the grid generation parameters used. For viscous flows, more grid lines are required in the η direction, with a smaller offwall spacing.



(a) NACA0012 (15 control points)

(b) RAE2822 (15 control points)

(c) NACA0012 (19 control points)

(d) RAE2822 (19 control points)

Figure 3.1: B-spline control points.

3.1.4 Control Points and B-Spline Interpolation

We use a set of B-spline control points to describe an airfoil. We follow the approach from Nemec and Zingg [17]. Figure 3.1 shows the control points and the airfoil geometry. The airfoil geometry is formed by B-spline interpolation through these control points. Fourth-order B-splines are used. We use 15 control points to describe an airfoil for inviscid design, and 19 points for viscous design.

Some control points may be fixed during a design process, with the remainder assigned as design variables. For example, three control points may be fixed at the leading edge to ensure a well-defined leading edge curvature. We can fix four control points at the trailing edge (two on each surface) to prevent trailing edge cross-over.

3.1.5 Optimizers

Two optimizers are used in this work. Both optimizers use a quasi-newton method to determine a search direction using an approximate Hessian matrix. The first one is the BFGS algorithm from Broyden-Fletcher-Goldfarb-Shanno, together with a cubic line search. The second optimizer is the KSOPT code written by Wrenn [6]. It uses the Davidon-Fletcher-Powell (DFP) algorithm with a quadratic line search. KSOPT is a constrained optimizer. Constraints are incorporated through the Kreisselmeier-Steinhauser (KS) function.

3.2 Gradient Solver

3.2.1 The Flow Sensitivity Approach

The gradients are calculated using the flow sensitivity approach. The flow sensitivity approach consists of two steps. The first step is to solve for the flow sensitivities from the sensitivity equations. Then we calculate the required gradients from the flow sensitivities. To present the sensitivity approach, we start with the flow equations

$$\mathcal{F}(\mathcal{Q}, g) = \partial_\xi \hat{E} + \partial_\eta \hat{F} - M_\infty Re^{-1} \partial_\eta \hat{S} = 0 \tag{3.9}$$

where $\mathcal{F}$ is the 4×1 vector of continuous flow equations. Then we differentiate the flow equations and expand using chain rule

$$\frac{d\mathcal{F}}{dg} = \frac{\partial \mathcal{F}}{\partial g} + \frac{\partial \mathcal{F}}{\partial \mathcal{Q}} \frac{d\mathcal{Q}}{dg} = 0 \tag{3.10}$$

Rearrange to obtain the sensitivity equation

$$\frac{\partial \mathcal{F}}{\partial \mathcal{Q}} \frac{d\mathcal{Q}}{dg} = -\frac{\partial \mathcal{F}}{\partial g} \tag{3.11}$$

The first step is to solve for the flow sensitivities $d\mathcal{Q}/dg$ from the sensitivity equation. There is one sensitivity equation for each design variable

$$\frac{\partial \mathcal{F}}{\partial \mathcal{Q}} \frac{d\mathcal{Q}}{dg_m} = -\frac{\partial \mathcal{F}}{\partial g_m} \tag{3.12}$$

The second step is to calculate the required gradients from the flow sensitivities

$$\frac{dJ}{dg} = \frac{\partial J}{\partial g} + \frac{\partial J}{\partial \mathcal{Q}} \frac{d\mathcal{Q}}{dg} \tag{3.13}$$

There is one gradient for each design variable

$$\frac{dJ}{dg_m} = \frac{\partial J}{\partial g_m} + \frac{\partial J}{\partial \mathcal{Q}} \frac{d\mathcal{Q}}{dg_m} \tag{3.14}$$

For each design problem, we first define an objective function J. Next, we calculate the partial derivatives $\partial J/\partial \mathcal{Q}$ and $\partial J/\partial g$. The required gradients can then be calculated using equation 3.13. We can repeat the same procedure to calculate constraint gradients, by reusing the flow sensitivities:

$$\begin{aligned} \frac{dC_1}{dg} &= \frac{\partial C_1}{\partial g} + \frac{\partial C_1}{\partial \mathcal{Q}} \frac{d\mathcal{Q}}{dg} \\ \frac{dC_2}{dg} &= \frac{\partial C_2}{\partial g} + \frac{\partial C_2}{\partial \mathcal{Q}} \frac{d\mathcal{Q}}{dg} \\ &\vdots \end{aligned} \tag{3.15}$$

3.2.2 Discrete and Continuous Formulations

There are two ways to implement the flow sensitivity approach. For the continuous sensitivity approach, we start with the continuous flow equations 3.9. Then we differentiate to obtain the continuous sensitivity equations 3.11. These equations are then discretized and solved to calculate the flow sensitivities.

The second method is the discrete sensitivity approach. In this formulation, the flow equations 3.9 are discretized to obtain the discrete flow equations

$$R(Q, g) = 0 \tag{3.16}$$

Then we differentiate the discrete flow equations 3.16 to obtain the discrete sensitivity equations

$$\frac{\partial R}{\partial Q} \frac{dQ}{dg} = -\frac{\partial R}{\partial g} \tag{3.17}$$

This is a large sparse linear system. The right-hand-side of the system is the partial derivative of the flow equations with respect to the design variables. This is calculated using centered differences:

$$\frac{\partial R}{\partial g} = \frac{R(Q, g+\epsilon) - R(Q, g-\epsilon)}{2\epsilon} \tag{3.18}$$

The left-hand-side of the system is the flow Jacobian matrix.

We use the discrete flow sensitivity approach. One advantage of the discrete approach is that a large part of the code can be reused from the flow solver.

3.2.3 Discrete Flow Equations

The flow equations are discretized exactly the same as the flow solver. Spatial discretization is based on second-order centered differences with a nonlinear dissipation model.

Finite Differencing

Second-order centered differences are used to discretize the inviscid fluxes on a structured grid

$$\begin{aligned} \partial_\xi \hat{E}_{j,k} &= \frac{\hat{E}_{j+1,k} - \hat{E}_{j-1,k}}{2} \\ \partial_\eta \hat{F}_{j,k} &= \frac{\hat{F}_{j,k+1} - \hat{F}_{j,k-1}}{2} \end{aligned} \tag{3.19}$$

The viscous flux involves second derivatives. Centered differences are evaluated at half nodes

$$\partial_\eta \hat{S}_{j,k} = \hat{S}_{j,k+\frac{1}{2}} - \hat{S}_{j,k-\frac{1}{2}} \tag{3.20}$$

The discrete flow equation at node (j,k) can be written as

$$\begin{aligned} R_{j,k}(Q,g) &= \frac{1}{2}\left[\hat{E}_{j+1,k} - \hat{E}_{j-1,k} + \hat{F}_{j,k+1} - \hat{F}_{j,k-1}\right] \\ &\quad -M_\infty Re^{-1}\left[\hat{S}_{j,k+\frac{1}{2}} - \hat{S}_{j,k-\frac{1}{2}}\right] + D_{j,k} = 0 \end{aligned} \tag{3.21}$$

The terms $D_{j,k}$ are the artificial dissipation terms.

Artificial Dissipation

The scalar nonlinear dissipation model of Jameson et al. [18] is used. The dissipation terms can be written as a sum of second- and fourth-order terms for both directions in the curvilinear mesh

$$D_{j,k} = D_{j,k}^{(2\xi)} + D_{j,k}^{(4\xi)} + D_{j,k}^{(2\eta)} + D_{j,k}^{(4\eta)} \tag{3.22}$$

The second and fourth order dissipation terms in the ξ direction are

$$D_{j,k}^{(2\xi)} = \Delta_\xi\left[\varepsilon^{(2)}_{j+\frac{1}{2},k}\nabla_\xi(J_{j,k}\hat{Q}_{j,k})\right] \tag{3.23}$$

$$D_{j,k}^{(4\xi)} = \Delta_\xi\left[\varepsilon^{(4)}_{j+\frac{1}{2},k}\nabla_\xi\Delta_\xi\nabla_\xi(J_{j,k}\hat{Q}_{j,k})\right] \tag{3.24}$$

Here Δ and ∇ denote first-order backward and forward differences respectively. The second-difference dissipation coefficients are

$$\varepsilon^{(2)}_{j+\frac{1}{2},k} = 2\left(\sigma^{(\xi)} J^{-1} \epsilon^{(2)}\right)_{j+\frac{1}{2},k} \tag{3.25}$$

The term $\sigma^{(\xi)}$ is the spectral radius of the flux Jacobian matrix $\partial\hat{E}/\partial\hat{Q}$

$$\sigma^{(\xi)}_{j,k} = |U| + a\sqrt{\xi_x^2 + \xi_y^2} \tag{3.26}$$

and

$$\epsilon^{(2)}_{j,k} = \kappa_2 \Delta t \max(\Upsilon_{j+1,k}, \Upsilon_{j,k}, \Upsilon_{j-1,k}) \tag{3.27}$$

The value of Δt is set to one in the gradient calculation. The pressure gradient coefficient Υ is defined as

$$\Upsilon_{j,k} = \frac{|p_{j+1,k} - 2p_{j,k} + p_{j-1,k}|}{|p_{j+1,k} + 2p_{j,k} + p_{j-1,k}|} \tag{3.28}$$

The fourth-difference dissipation coefficients are

$$\varepsilon^{(4)}_{j+\frac{1}{2},k} = \max\left[0, \kappa_4 2\left(\sigma^{(\xi)} J^{-1}\right)_{j+\frac{1}{2},k} - 2\epsilon^{(2)}_{j+\frac{1}{2},k}\right] \tag{3.29}$$

The values of κ_2 and κ_4 used are 1.0 and 0.01 respectively.

3.2.4 Boundary Conditions

We follow the approach of Pueyo [19] in the treatment of boundary conditions. There are two types of boundary conditions: body boundary and far field. At body boundaries, flow tangency conditions are imposed for inviscid flow and no-slip conditions are used for viscous flow. The characteristic approach from Riemann is used to determine the far-field boundary conditions. The inviscid body boundary conditions are

$$\begin{aligned} (V_n)_{j,1} &= 0 \\ (V_t)_{j,1} - 2(V_t)_{j,2} + (V_t)_{j,3} &= 0 \\ p_{j,1} - 2p_{j,2} + p_{j,3} &= 0 \\ H_{j,1} - H_\infty &= 0 \end{aligned} \tag{3.30}$$

Here, V_n and V_t are the velocity components normal and tangential to the body surface respectively. The stagnation enthalpy is denoted as $H = (e + p)/\rho$. The viscous body

conditions are

$$\begin{aligned} \rho_{j,1} - \rho_{j,2} &= 0 \\ (\rho u)_{j,1} &= 0 \\ (\rho v)_{j,1} &= 0 \\ p_{j,1} - p_{j,2} &= 0 \end{aligned} \tag{3.31}$$

Far-field boundary conditions are calculated using Riemann invariants. For subsonic outflow at the $k = k_{max}$ boundary, the far-field boundary conditions are

$$\begin{aligned} \left(V_n - \frac{2a}{\gamma - 1}\right)_{j,k_{max}} - \left(V_n - \frac{2a}{\gamma - 1}\right)_{\infty} &= 0 \\ \left(V_n + \frac{2a}{\gamma - 1}\right)_{j,k_{max}} - \left(V_n + \frac{2a}{\gamma - 1}\right)_{j,k_{max}-1} &= 0 \\ S_{j,k_{max}} - S_{j,k_{max}-1} &= 0 \\ (V_t)_{j,k_{max}} - (V_t)_{j,k_{max}-1} &= 0 \end{aligned} \tag{3.32}$$

Here, $S = p^{\gamma}/\rho$ is the entropy. For inflow conditions, S and V_t are set to free stream values. Far-field boundary conditions at $j = 1$ and $j = j_{max}$ can be derived similarly.

3.2.5 Differentiation of the Discrete Flow Equations

We differentiate the discrete flow equations to obtain the discrete sensitivity equations

$$\frac{\partial R}{\partial Q}\frac{dQ}{dg} = -\frac{\partial R}{\partial g} \tag{3.33}$$

The left-hand side is the flow Jacobian matrix. This can be reused from existing flow solvers that use an implicit approach. Many flow solvers do not use the exact flow Jacobian to solve the flow equations. An approximate Jacobain is commonly used and it does not affect the accuracy of the flow solution. However, the accuracy of the flow sensitivities depends on how accurately the flow Jacobian is formed. Recall the discretized flow equations from section 3.2.3

$$\begin{aligned} R_{j,k}(Q,g) = \; & \frac{1}{2}\left[\hat{E}_{j+1,k} - \hat{E}_{j-1,k} + \hat{F}_{j,k+1} - \hat{F}_{j,k-1}\right] \\ & -M_{\infty}Re^{-1}\left[\hat{S}_{j,k+\frac{1}{2}} - \hat{S}_{j,k-\frac{1}{2}}\right] \\ & +D_{j,k}^{(2\xi)} + D_{j,k}^{(4\xi)} + D_{j,k}^{(2\eta)} + D_{j,k}^{(4\eta)} = 0 \end{aligned} \tag{3.34}$$

The linearization of equation 3.34 leads to nine 4×4 blocks. The nine blocks are given by

$$
\begin{aligned}
B_{j,k}^{-2,\xi} &= \frac{\partial D_{j,k}^{(4\xi)}}{\partial \hat{Q}_{j-2,k}} \\
B_{j,k}^{-1,\xi} &= \frac{\partial D_{j,k}^{(4\xi)}}{\partial \hat{Q}_{j-1,k}} + \frac{\partial D_{j,k}^{(2\xi)}}{\partial \hat{Q}_{j-1,k}} - \frac{1}{2}\frac{\partial \hat{E}_{j-1,k}}{\partial \hat{Q}_{j-1,k}} \\
B_{j,k}^{-2,\eta} &= \frac{\partial D_{j,k}^{(4\eta)}}{\partial \hat{Q}_{j,k-2}} \\
B_{j,k}^{-1,\eta} &= \frac{\partial D_{j,k}^{(4\eta)}}{\partial \hat{Q}_{j,k-1}} + \frac{\partial D_{j,k}^{(2\eta)}}{\partial \hat{Q}_{j,k-1}} - \frac{1}{2}\frac{\partial \hat{F}_{j,k-1}}{\partial \hat{Q}_{j,k-1}} - M_\infty Re^{-1}\frac{\partial \hat{S}_{j,k-\frac{1}{2}}}{\partial \hat{Q}_{j,k-1}} \\
B_{j,k}^{0} &= \frac{\partial D_{j,k}^{(4\xi)}}{\partial \hat{Q}_{j,k}} + \frac{\partial D_{j,k}^{(2\xi)}}{\partial \hat{Q}_{j,k}} + \frac{\partial D_{j,k}^{(4\eta)}}{\partial \hat{Q}_{j,k}} + \frac{\partial D_{j,k}^{(2\eta)}}{\partial \hat{Q}_{j,k}} \\
&\quad + M_\infty Re^{-1}\left[\frac{\partial \hat{S}_{j,k-\frac{1}{2}}}{\partial \hat{Q}_{j,k}} + \frac{\partial \hat{S}_{j,k+\frac{1}{2}}}{\partial \hat{Q}_{j,k}}\right] \\
B_{j,k}^{+1,\eta} &= \frac{\partial D_{j,k}^{(4\eta)}}{\partial \hat{Q}_{j,k+1}} + \frac{\partial D_{j,k}^{(2\eta)}}{\partial \hat{Q}_{j,k+1}} - \frac{1}{2}\frac{\partial \hat{F}_{j,k+1}}{\partial \hat{Q}_{j,k+1}} - M_\infty Re^{-1}\frac{\partial \hat{S}_{j,k+\frac{1}{2}}}{\partial \hat{Q}_{j,k+1}} \\
B_{j,k}^{+2,\eta} &= \frac{\partial D_{j,k}^{(4\eta)}}{\partial \hat{Q}_{j,k+2}} \\
B_{j,k}^{+1,\xi} &= \frac{\partial D_{j,k}^{(4\xi)}}{\partial \hat{Q}_{j+1,k}} + \frac{\partial D_{j,k}^{(2\xi)}}{\partial \hat{Q}_{j+1,k}} - \frac{1}{2}\frac{\partial \hat{E}_{j+1,k}}{\partial \hat{Q}_{j+1,k}} \\
B_{j,k}^{+2,\xi} &= \frac{\partial D_{j,k}^{(4\xi)}}{\partial \hat{Q}_{j+2,k}}
\end{aligned}
\tag{3.35}
$$

If all the terms are linearized exactly, we obtain the exact flow Jacobian A_e. However, some terms are difficult to linearize analytically such as equations 3.26 and 3.27. This occurs in the dissipation terms, turbulence model and circulation correction.

One way to avoid this difficulty is to treat the coefficients of the artificial dissipation, as well as μ and μ_t as constants in the linearization. This forms the second-order approximate Jacobian A_2. It is used in the flow solver PROBE [20]. Alternatively, it is possible to obtain the exact Jacobian numerically, at a higher computational cost.

The second-order approximate Jacobian can be approximated further by using only second-difference dissipation terms. This forms the first-order approximate Jacobian A_1 as defined in reference [20]. The dissipation coefficients of the new matrix are given by

$$\varepsilon_2^l = \varepsilon_2^r + \sigma\varepsilon_4^r \tag{3.36}$$

Here the superscript r denotes values from the right-hand side and l for the first-order Jacobian. Fourth-difference dissipation terms are removed. The resulting matrix is more diagonally dominant and it has five blocks per row instead of nine. Less computational cost and memory are required to form this matrix. We use this matrix as a preconditioner in solving the sensitivity equations as shown in the next section.

Besides the interior nodes, the boundary equations are linearized as well when forming the flow Jacobian matrix. Implicit boundary conditions are required to calculate accurate flow sensitivities. Reference [19] provides a detailed description about linearization of boundary conditions.

3.2.6 Solving the Discrete Flow Sensitivity Equations

The sensitivity equation is a large sparse linear system. Solving the system using direct solvers such as LU factorization is computationally expensive. We use the GMRES approach, which is an iterative solver for linear systems. We use the restart version of GMRES, denoted as GMRES(m).

The system needs preconditioning before applying the GMRES approach. This improves the efficiency and robustness of the linear solver. We use a preconditioner based on a first-order Jacobian matrix, following the approach of Pueyo and Zingg [20]. Right preconditioning is used. The preconditioner is decomposed using block-fill incomplete LU factorization with a level of fill, BFILU(n).

Since the GMRES algorithm requires only matrix-vector products, the sensitivity equations can be solved using a matrix-free approach. Matrix-vector products are approximated by second-order Frechet derivatives

$$\frac{\partial R}{\partial Q}v = \frac{R(Q+\epsilon v) - R(Q-\epsilon v)}{2\epsilon} \tag{3.37}$$

The order of the derivatives affects the accuracy of the flow sensitivities. The stepsize is chosen as

$$\epsilon\|v\| \simeq \sqrt{\epsilon_m} \tag{3.38}$$

where $\|v\|$ is the norm of v and ϵ_m is the value of machine zero.

The matrix-free approach reduces memory use, and avoids difficulties in linearization. Figure 3.2 shows the gradient accuracy of the matrix-free and the standard approaches. Matrix-free sensitivity gradients are more accurate when compared to finite-difference gra-

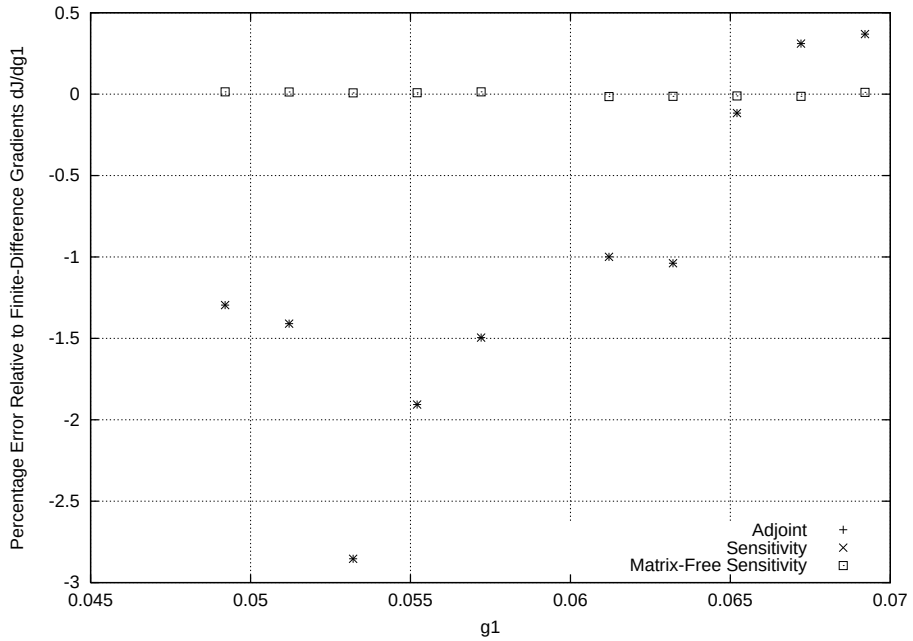


Figure 3.2: Comparing gradient accuracy relative to finite-difference gradients.

dients. The reason is that matrix-free Jacobian are closer to the exact Jacobian A_e, than the second-order Jacobian A_2 used in the matrix approach.

It is possible to obtain adjoint gradients with the same level of accuracy as the matrix-free sensitivity gradients by using the exact Jacobian. One suggestion to form an exact Jacobian in affordable cost is to calculate $\partial\hat{E}/\partial\hat{Q}$, $\partial\hat{F}/\partial\hat{Q}$ and $\partial\hat{S}/\partial\hat{Q}$ analytically, but $\partial D/\partial\hat{Q}$ numerically.

The ordering of unknowns affects the convergence of the preconditioned iterative solver [20]. Figure 3.3 shows the structure of the Jacobian matrix using natural ordering on a 22×8 grid. The off-diagonal elements are caused by the wakecut. Figure 3.4 shows the matrix structure using reverse Cuthill-McKee (RCM) reordering. The resulting matrix has a lower bandwidth.

The flow sensitivity equations are solved using the GMRES approach with the following parameters:

- Matrix-free GMRES(90) with second-order Frechet derivatives
- Preconditioner based on first-order Jacobian matrix with $\sigma = 3.0$

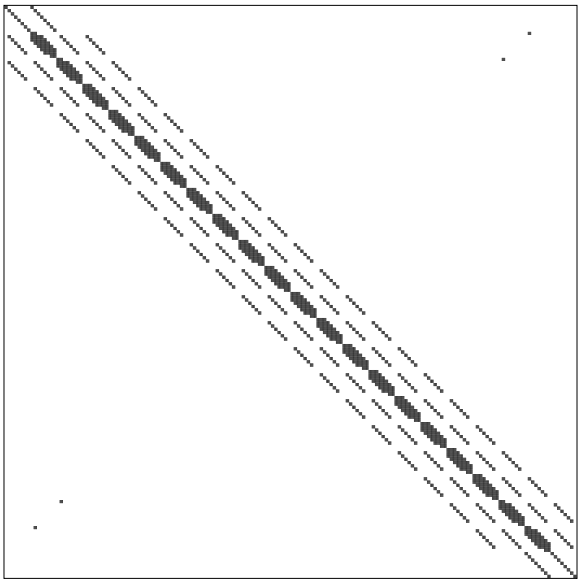


Figure 3.3: Matrix structure of the flow Jacobian matrix using natural ordering.

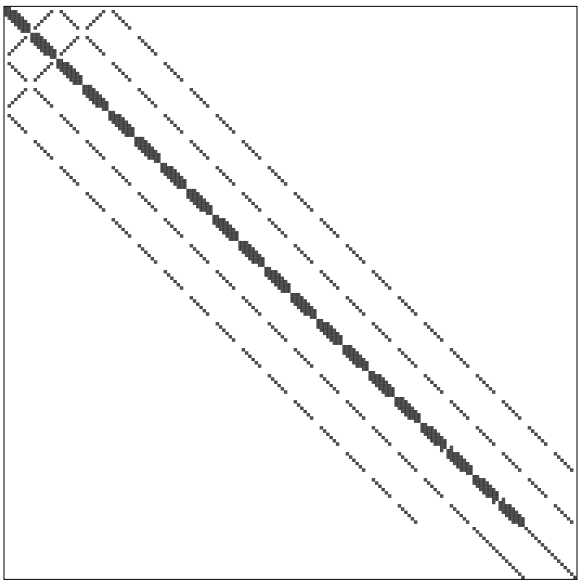


Figure 3.4: Matrix structure of the flow Jacobian matrix using reverse Cuthill-McKee (RCM) reordering.

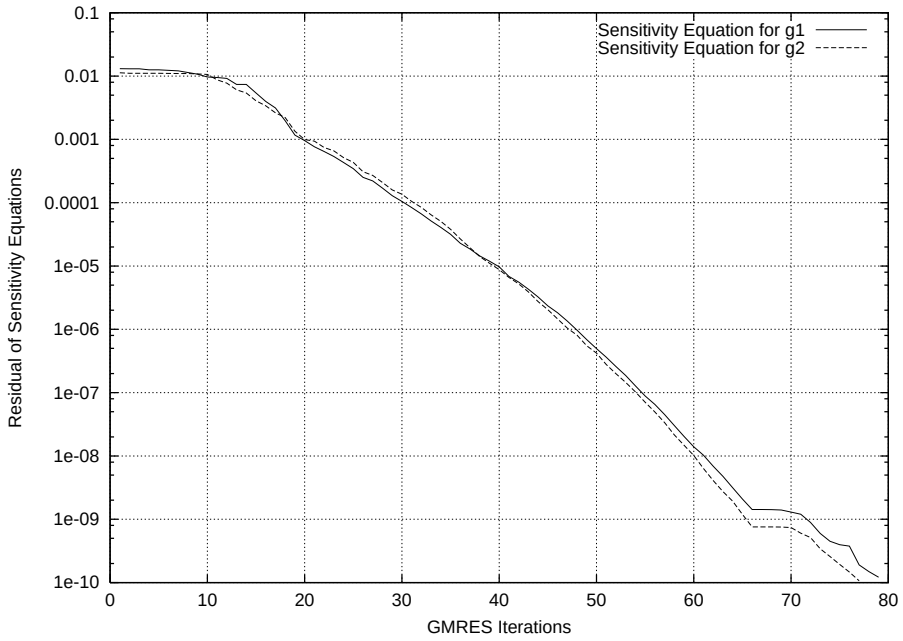


Figure 3.5: Typical GMRES convergence histories for the sensitivity problem.

- Preconditioner decomposed using BFILU(5)
- RCM reordering of unknowns

A parametric study of the gradient solver parameters is shown in Appendix D. Figure 3.5 shows a typical convergence history of the sensitivity problem.

3.3 Gradient Results

3.3.1 Contour Plot of Flow Sensitivities

The discrete flow sensitivities are solved for at every node in the grid. Figure 3.6 shows a contour plot of the flow sensitivities with respect to control point number 6 of the inviscid RAE2822 airfoil. Control point number 6 is located on the bottom surface as shown in figure 3.1(b). The airfoil is at Mach number 0.74 and 2.0 degrees angle of attack. Inviscid results are shown. Flow sensitivites dp/dg are plotted and they represent the corresponding change in pressure with respect to the design variable. They are calculated by differentiation of equation 3.5 analytically. Although control point number 6 is located

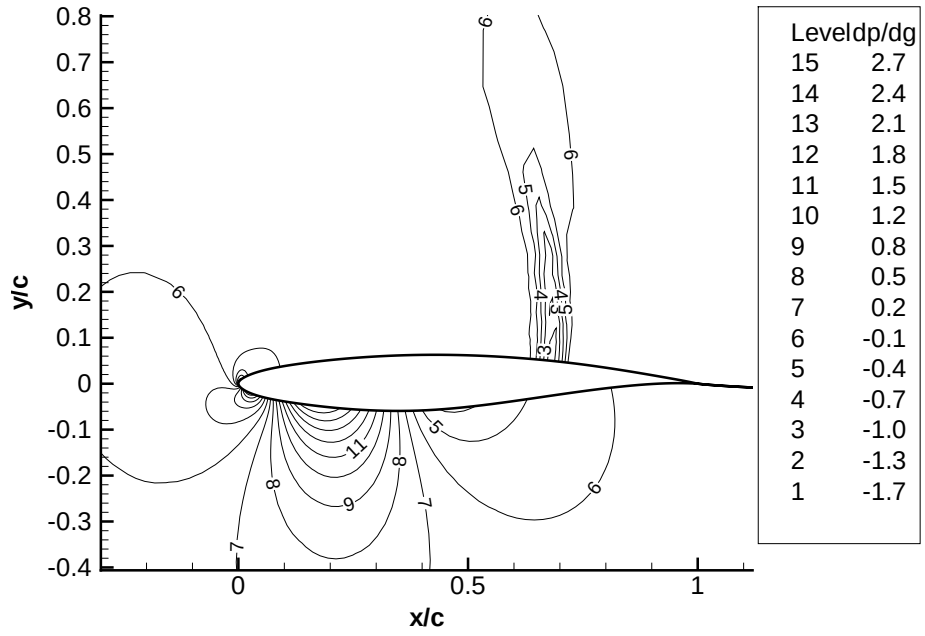


Figure 3.6: Contour plot of the flow sensitivities: change in pressure field with respect to control point number 6.

on the lower surface, it affects the flow near the shock as shown in the figure.

Figure 3.7 shows the flow sensitivities with respect to control point number 12 on the upper surface close to the shock. Flow sensitivities have large values near the shock. This means that the flow at this region is very sensitive to the control point. Figure 3.8 shows the flow sensitivities with respect to the angle of attack.

3.3.2 Finite-Difference Gradient Accuracy

The accuracy of the finite-difference gradients depends on both the flow solver tolerance and the finite-difference stepsize. In this and the next section, the error is defined by comparing to the finite-difference gradients obtained using a flow solver tolerance of 10^{-15} and a finite-difference stepsize of 10^{-6}. Design variables g_1 and g_2 correspond to control points number 10 and 11, respectively, as shown in figure 3.1(b). Inviscid results are shown. Figure 3.9 shows the effect of flow solver tolerance. We use a flow solver tolerance of 10^{-15}. The effect of finite-difference stepsize is shown in figure 3.10. We use a stepsize of 10^{-6}.

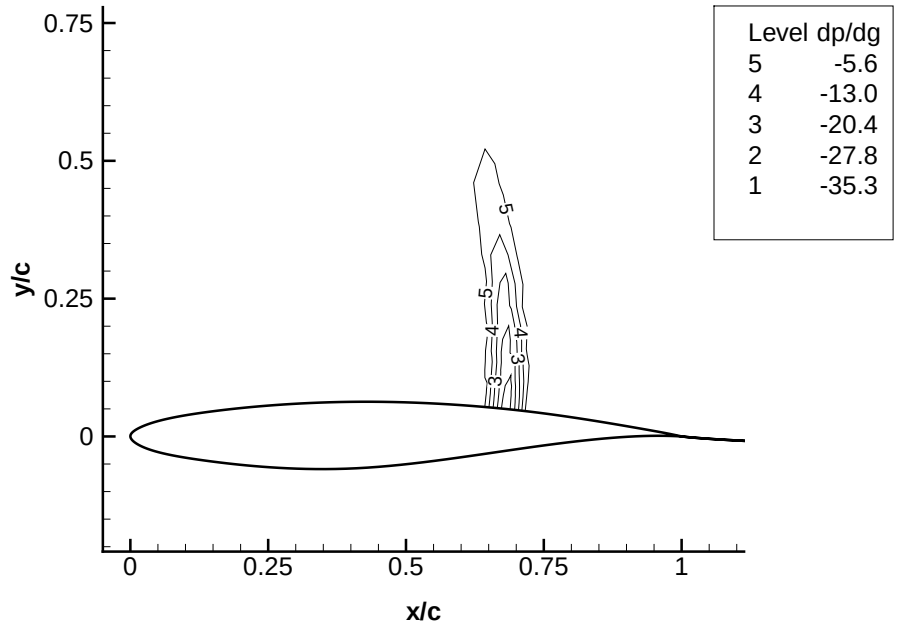


Figure 3.7: Contour plot of the flow sensitivities: change in pressure field with respect to control point number 12.

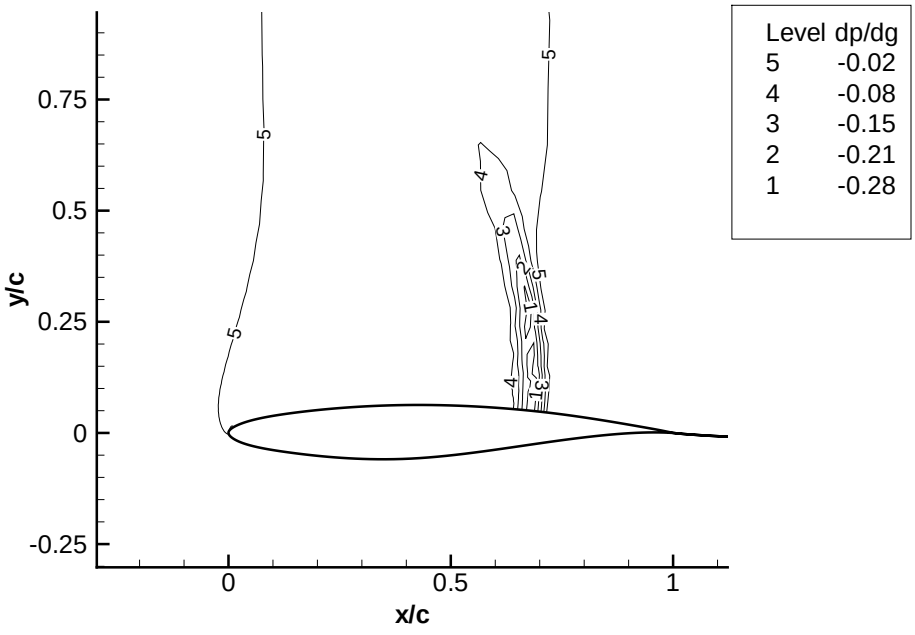


Figure 3.8: Contour plot of the flow sensitivities: change in pressure field with respect to the angle of attack.

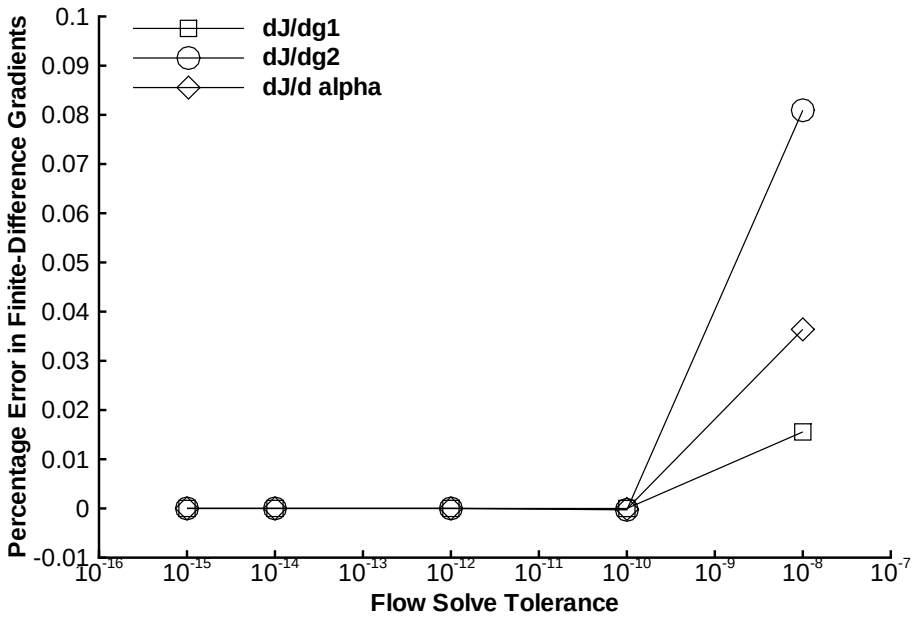


Figure 3.9: Effect of flow solver tolerance on finite-difference gradients.

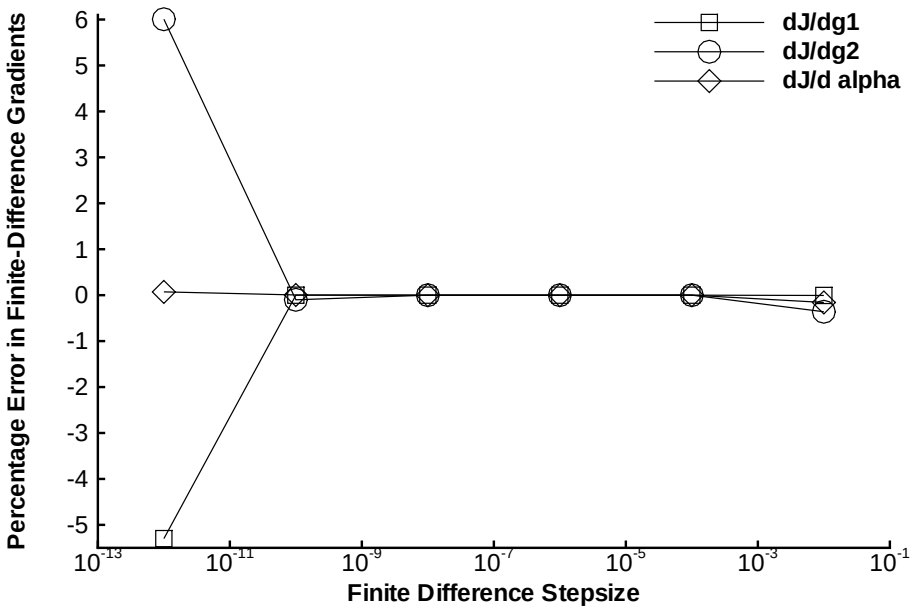


Figure 3.10: Effect of stepsize on finite-difference gradients.

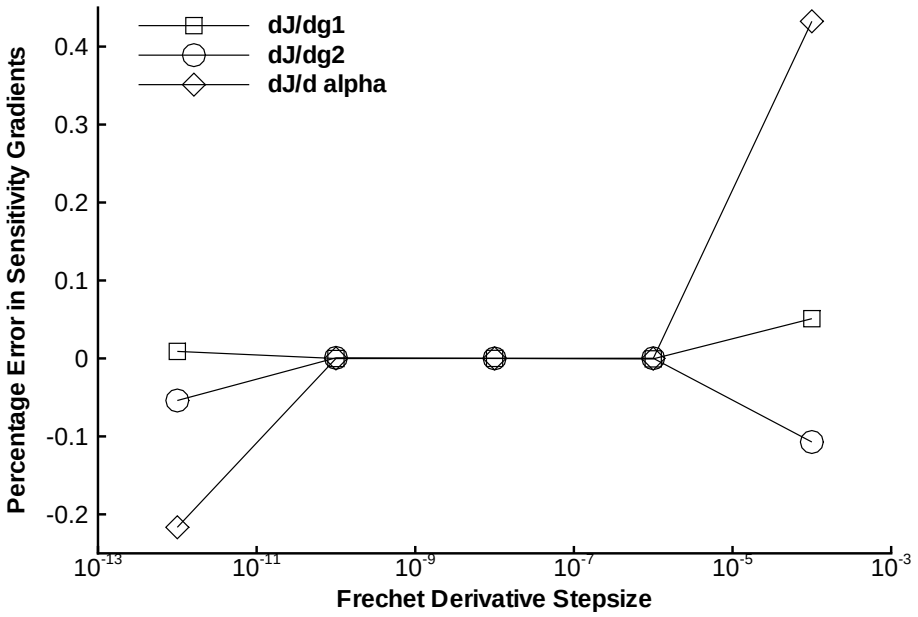


Figure 3.11: Effect of Frechet derivative stepsize on sensitivity gradients.

3.3.3 Sensitivity Gradient Accuracy

The accuracy of the matrix-free sensitivity gradients depends on the Frechet derivative stepsize, GMRES solver tolerance, finite-difference stepsizes (in $\partial J/\partial Q$, $\partial J/\partial g$ and $\partial R/\partial g$), flow solver tolerance and the order of Frechet derivatives used.

Figure 3.11 shows the effect of Frechet derivative stepsize. Figure 3.12 shows the effect of GMRES solver tolerance. We use a GMRES solver tolerance of 10^{-8}. Figure 3.13 shows the effect of finite-difference stepsize in $\partial J/\partial Q$, $\partial J/\partial g$ and $\partial R/\partial g$. We use a stepsize of 10^{-6} for these partial derivatives. Figure 3.14 shows the effect of flow solver tolerance. We use a solver tolerance of 10^{-15}. It is sometimes difficult to obtain low solver tolerance for viscous flows in some situations, as shown in the next section. Figure 3.15 shows the gradient results using first- and second-order Frechet derivatives, compared to finite-difference gradients.

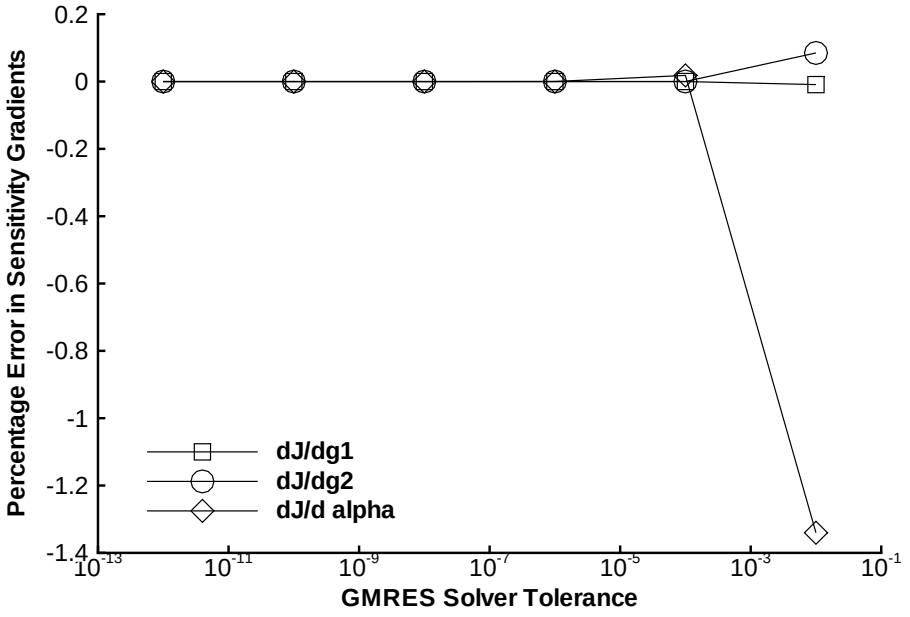


Figure 3.12: Effect of GMRES solver tolerance on sensitivity gradients.

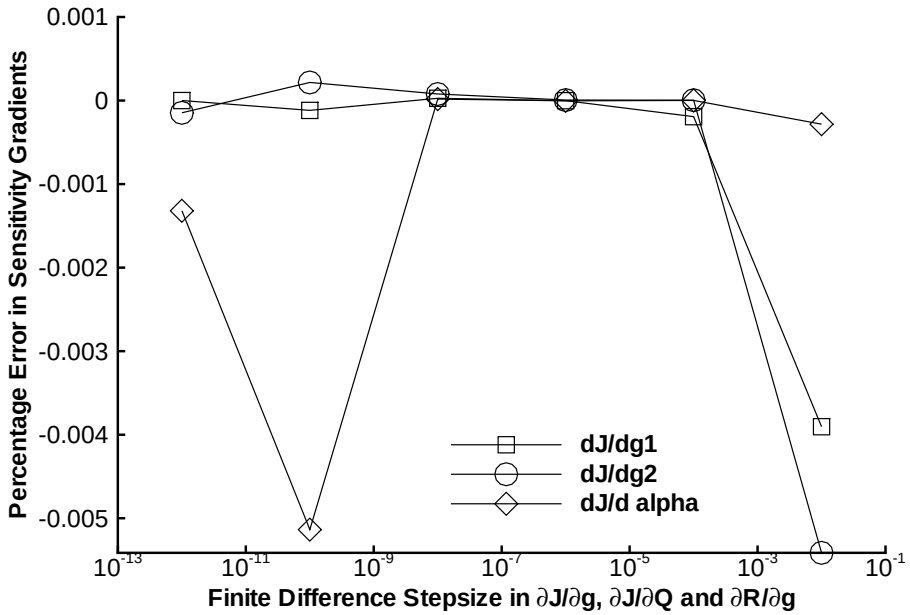


Figure 3.13: Effect of stepsize in $\partial J/\partial Q$, $\partial J/\partial g$ and $\partial R/\partial g$ on sensitivity gradients.

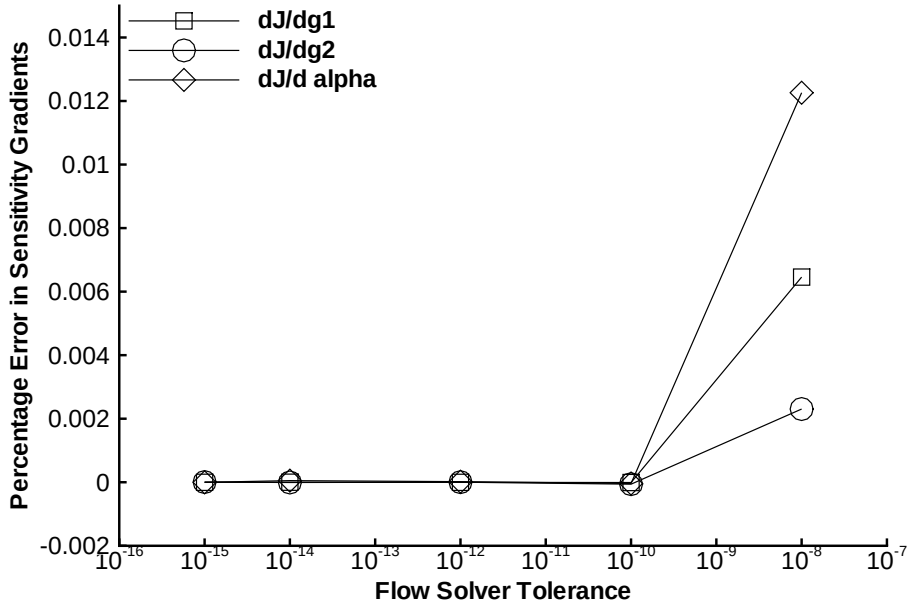


Figure 3.14: Effect of flow solver tolerance on sensitivity gradients.

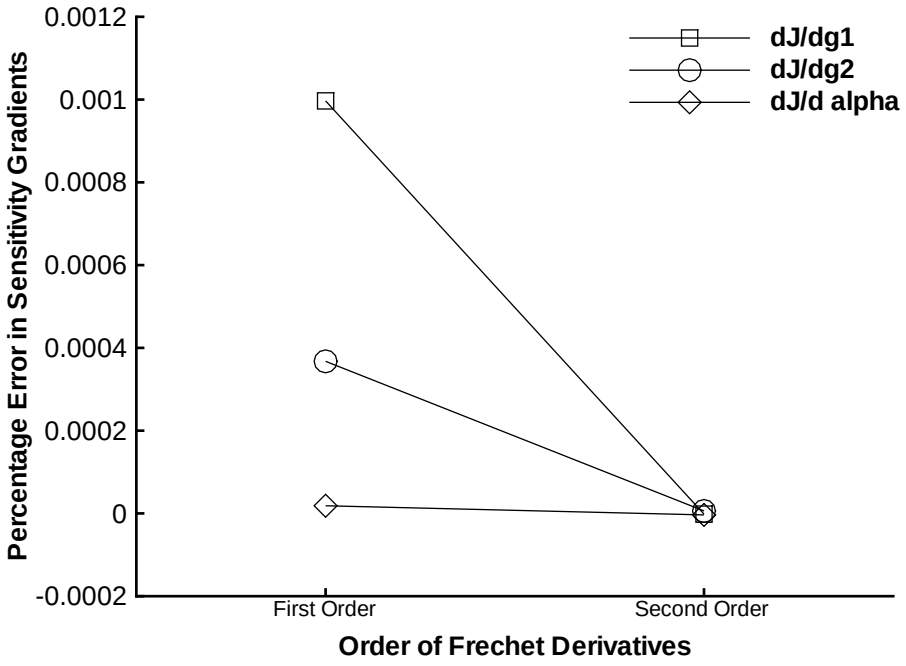


Figure 3.15: Effect of the order of Frechet derivatives on sensitivity gradients.

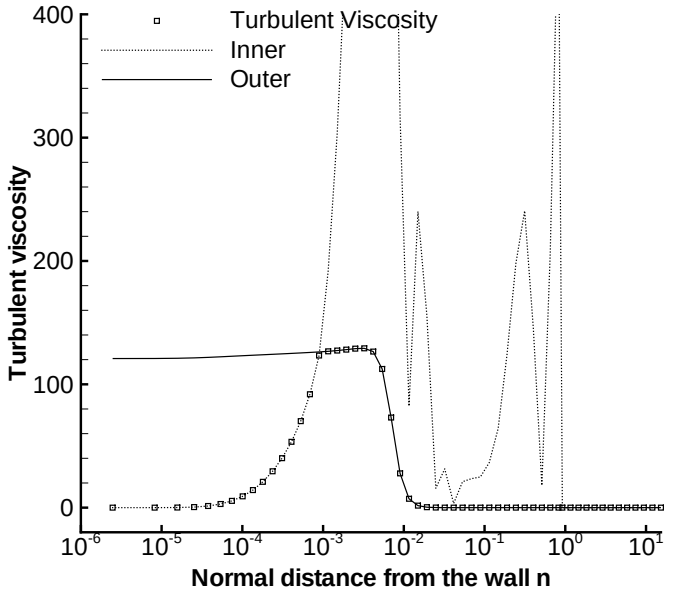


Figure 3.16: Plot of the turbulent viscosity μ_t vs. n.

3.4 Turbulence Model

Turbulent flows are simulated using the Baldwin-Lomax turbulence model. This is an algebraic model. The turbulent viscosity is calculated from the flow variables in each flow solve iteration and updated.

3.4.1 Baldwin-Lomax Turbulence Model

The Baldwin and Lomax model is a two-layer eddy viscosity model. The turbulent viscosity is given by

$$\mu_t = \begin{cases} (\mu_t)_{inner} & n \leq n_{cr} \\ (\mu_t)_{outer} & n > n_{cr} \end{cases} \tag{3.39}$$

where n is the normal distance from the wall. Figure 3.16 shows a plot of a typical turbulent viscosity term, together with its inner and outer components.

The inner turbulent viscosity is calculated using the Prandtl-Van Driest formulation

$$(\mu_t)_{inner} = \rho l^2 |\omega| \tag{3.40}$$

where

$$l = kn[1 - \exp(-n^+/A^+)] \tag{3.41}$$

The magnitude of the vorticity is given by

$$|\omega| = |u_y - v_x| \tag{3.42}$$

and the law-of-the-wall coordinate n^+ is

$$n^+ = \frac{\rho_w u_\tau n}{\mu_w} = \frac{\sqrt{\rho_w \tau_w} n}{\mu_w} \tag{3.43}$$

The subscript w denotes values at the wall, u_τ is the friction velocity, $\sqrt{\rho_w \tau_w}$, and τ_w is the shear stress at the wall.

The outer turbulent viscosity is given by

$$(\mu_t)_{outer} = KC_{cp}\rho F_{wake} F_{kleb}(n) \tag{3.44}$$

where K is the Clauser constant, and

$$F_{wake} = \min \begin{cases} n_{max} F_{max} \\ C_{wk} n_{max} u_{dif}^2 / F_{max} \end{cases} \tag{3.45}$$

The values of F_{max} and n_{max} are determined from the function

$$F(n) = n|\omega|[1 - \exp(-n^+/A^+)] \tag{3.46}$$

A plot of $F(n)$ vs. n is shown in figure 3.17. In wakes, the exponential term in equation 3.46 is set to zero. The value n_{max} is the value of n which $F(n)$ reaches its maximum F_{max}. The function $F_{kleb}(n)$ is the Klebanoff intermittency factor

$$F_{kleb}(n) = \left[1 + 5.5\left(\frac{C_{kleb} n}{n_{max}}\right)^6\right]^{-1} \tag{3.47}$$

The value of u_{dif} is the difference between maximum and minimum total velocity in the profile

$$u_{\text{dif}} = \begin{cases} (\sqrt{u^2 + v^2})_{max} & \text{in boundary layers} \\ (\sqrt{u^2 + v^2})_{max} - (\sqrt{u^2 + v^2})_{min} & \text{in wakes} \end{cases} \tag{3.48}$$

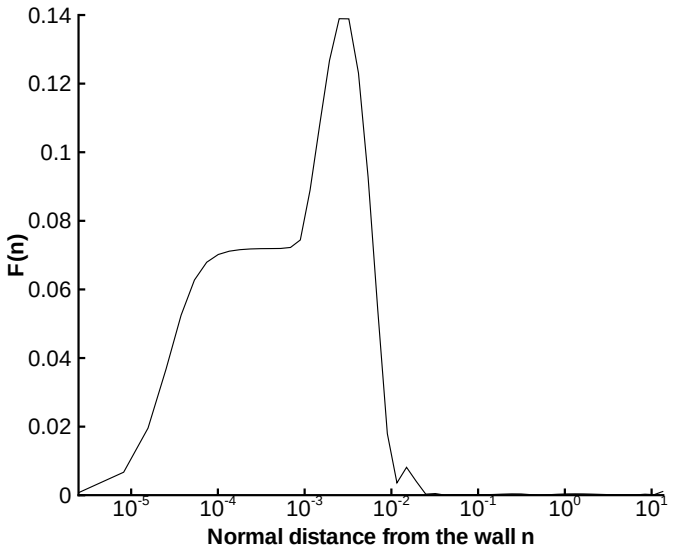


Figure 3.17: Plot of the function $F(n)$ vs. n.

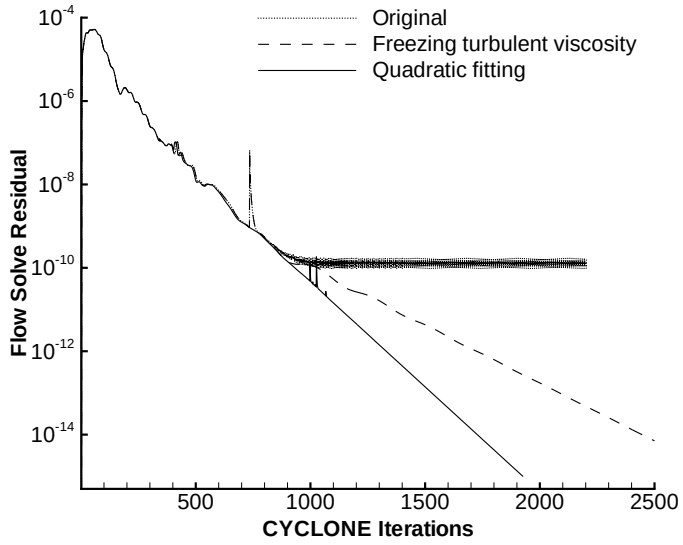


Figure 3.18: Flow solver convergence using the Baldwin-Lomax turbulence model.

3.4.2 Fix to the Baldwin-Lomax Model Convergence

Flow solves using this turbulence model often do not converge below a residual of roughly 10^{-10}. Figure 3.18 shows the hanging of the flow solve convergence history.

In the calculation of the outer viscosity term, we need to determine the maximum of the non-linear function $F(n)$ in equation 3.46. The main problem is that vorticity is only evaluated at discrete grid nodes in the η direction. Since the function F is calculated from the vorticities, its value is known only at discrete nodes.

In older version of the code, the maximum values are found by a search through the grid nodes. The node with the maximum function value is denoted by an integer value $\hat{\eta}_{max}$. There are cases where the maximum will oscillate between two adjacent grid nodes. Figure 3.17 shows an example. Figure 3.19 shows the oscillation in the turbulence viscosity caused by failure in determination of F_{max}. This causes convergence to stall.

We solve the problem using a real number for η_{max}. This real maximum is calculated by a quadratic fit of the function through three grid points:

$$F(\eta) = A\eta^2 + B\eta + C \tag{3.49}$$

The maximum of the parabola is then determined to be the maximum of the function F_{max}, with the corresponding grid location η_{max}. The value of n_{max} is then found using linear interpolation through the grid points:

$$n(\eta) = D\eta + E \tag{3.50}$$

With this modification, the flow solver no longer hangs and converges to machine zero, as shown in figure 3.18. The modified code may experience convergence problems in the first few iterations. We use the original approach for the first 100 iterations and run the rest using the modification.

Comparing the results obtained using both versions, the total lift is different by 0.03% and total drag is different by 0.08%. The new version of the code increased the computational cost of a flow solve by 16%. The current code is not optimized for speed and it is possible to improve this by removing some redundant code.

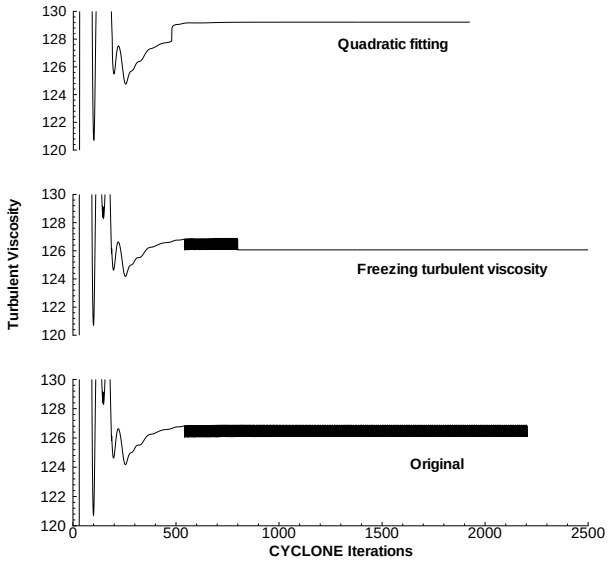


Figure 3.19: Change of the turbulent viscosity during a flow solve.

Chapter 4

Results

4.1 Inviscid Design Results

Three design cases are presented for inviscid airfoil design. Table 4.1 summarizes the number of design variables (n_{dv}) and the free-stream flow conditions for the design cases. Two optimizers are used. Sections 4.1.1 to 4.1.3 show the design results using BFGS. Results using KSOPT are shown in section 4.1.4. The required runtime for the inviscid design cases are recorded in section 4.1.5.

4.1.1 Inverse Design

The first case is the inverse design problem. The NACA0012 is used as the initial airfoil and the RAE2822 is specified as the target. The objective function is given by

$$J(Q) = \frac{1}{2} \sum_j \left(C_{p,j} - C^*_{p,j} \right)^2 \tag{4.1}$$

The objective function is the sum of the differences between the current and target pressure distribution on the airfoil surface. The target pressure C^*_p is generated by running a flow

Case	Title	n_{dv}	M_∞	α_{initial}
1	Inverse design	$8 + \alpha$	0.30	2.0
2	Drag minimization with lift constraint	$8 + \alpha$	0.74	2.0
3	Lift enhancement with pressure constraint	$12 + \alpha$	0.30	2.0

Table 4.1: Inviscid design case parameters.

Control point	Start at target mf(2)	Start at NACA0012	
		Sensitivity mf(2)	Finite differences
3	0.33 E-15	-0.862090 E+02	-0.862091 E+02
4	0.66 E-15	-0.314694 E+02	-0.314694 E+02
5	0.14 E-16	0.143369 E+02	0.143366 E+02
6	0.23 E-15	0.142520 E+02	0.142472 E+02
10	-0.44 E-14	0.140756 E+03	0.140754 E+03
11	-0.69 E-15	0.789935 E+01	0.789863 E+01
12	0.11 E-15	-0.409515 E+02	-0.409515 E+02
13	0.26 E-14	-0.722893 E+02	-0.722893 E+02
α	0.67 E-16	-0.404324 E+01	-0.404324 E+01

Table 4.2: Verification of gradient accuracy for the inviscid inverse design problem. (Inviscid case 1.)

solve on the target RAE2822. In this case, the objective function is a function of the flow variables only.

This is a special case since the optimum design is known in advance. We start with the target airfoil to verify the gradients. Since the airfoil is the optimum design, the gradients should be zero. The results are shown in the second column of table 4.2. The norm of the gradients has an order 10^{-15}. The matrix-free sensitivity approach with second order Frechet derivatives mf(2) are used. The remaining control points not shown in the table are fixed with values that correspond to the target RAE2822 airfoil.

Next we verify the gradient accuracy at the initial airfoil NACA0012. The sensitivity gradients are compared to the finite difference gradients. This is shown in the last two columns in table 4.2. The sensitivity gradients agree well with the finite difference gradients. The norm of difference between the two sets of gradients is 10^{-3}.

We start the optimization process after verifying the gradients. The optimizer BFGS is used. Figure 4.1 shows the design results. The optimizer sucessfully generates the target airfoil geometry by matching the target pressure distribution. The design history is shown in figure 4.2. The objective function is reduced by 16 orders of magnitude. The norm of the difference between the final and target pressure distribution is of the order of machine zero. The norm of the gradients is reduced by 8 orders of magnitude. Usually, gradient convergence can only achieve the square root of the objective function convergence.

The optimizer BFGS requires one flow solve and one gradient calculation in each iteration. In figure 4.2, one design cycle is defined as one BFGS iteration and it consists

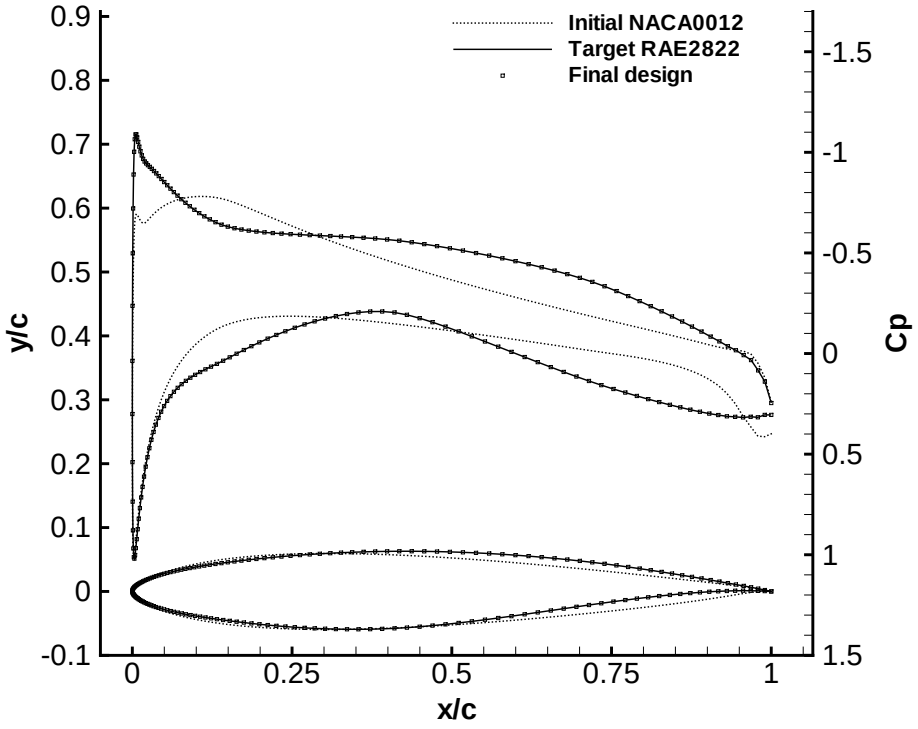


Figure 4.1: Airfoil geometry and pressure distribution of the inviscid inverse design problem. (Inviscid case 1, BFGS.)

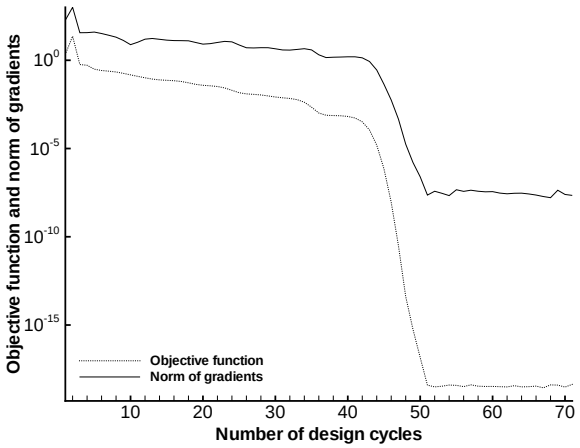


Figure 4.2: Design history of the inviscid inverse design problem. (Inviscid case 1, BFGS.)

of one flow solve and one gradient calculation. The definition is different for the other optimizer KSOPT.

The initial airfoil is modified to achieve convergence to machine zero. The initial airfoil used is not an exact NACA0012 airfoil. The leading and trailing edges of the initial airfoil geometry, as well as the wakecut of the airfoil grid, are modified to be exactly the same as those for the target RAE2822 airfoil. This ensures the pressure can be matched exactly at these positions. If an exact NACA0012 airfoil is used as the initial airfoil with all design variables used, the objective function only reduces 2 orders of magnitude. This is due to the fact that the wakecut is fixed during the design, which cause failure in matching the trailing edge pressure exactly. Regridding the airfoil regularly during the design may solve this problem.

4.1.2 Drag Minimization with Lift Constraint

The second case is the drag minimization problem. The objective is to minimize the drag of the baseline RAE2822 with a lift constraint: the lift must be greater than or equal to the original lift coefficient. No thickness constraints are applied. We fix three control points at the leading edge and four control points at the trailing edge in order to ensure a well-defined leading edge curvature and to prevent cross-over in the trailing edge geometry. The objective function is given by

$$J(Q,g) = \left(\frac{C_d}{C_d^*}\right)^2 + J_{\text{lift}}(Q,g) \tag{4.2}$$

The objective function is composed of two terms. The first term is the drag coefficient to be minimized. This is normalized with the initial drag C_d^*. The second term is the lift constraint penalty term

$$J_{\text{lift}}(Q,g) = \begin{cases} \left(1 - \dfrac{C_l}{C_l^*}\right)^2 & \text{if } C_l < C_l^* \\ 0 & \text{if } C_l \geq C_l^* \end{cases} \tag{4.3}$$

The objective function is penalized if the lift coefficient is less than a lower limit. The lower limit C_l^* is specified as the initial lift in this example.

The sensitivity gradients are verified and compared to the finite-difference gradients before running the optimization. The results are shown in the second and third columns of table 4.3. The two sets of gradients agree well with each other. The norm of difference

Control point	Initial		Final Design	
	Sensitivity mf(2)	Finite differences	Sensitivity mf(2)	Finite differences
3	0.339014 E+00	0.339014 E+00	0.562909 E-02	0.562910 E-02
4	0.165348 E+00	0.165348 E+00	0.279830 E-02	0.279830 E-02
5	0.765403 E-01	0.765403 E-01	0.133344 E-02	0.133344 E-02
6	0.139446 E-01	0.139446 E-01	0.295906 E-03	0.295906 E-03
10	0.877925 E+00	0.877925 E+00	0.557355 E-02	0.557376 E-02
11	0.522648 E+00	0.522648 E+00	0.255430 E-02	0.255431 E-02
12	-0.788470 E-01	-0.788470 E-01	0.294871 E-02	0.294871 E-02
13	0.262841 E+00	0.262841 E+00	0.634060 E-02	0.634060 E-02
α	0.170910 E-01	0.170910 E-01	0.275055 E-03	0.275055 E-03

Table 4.3: Verification of gradient accuracy for the inviscid drag minimization problem. (Inviscid case 2.)

	C_l	C_d	α°
Initial	0.913	0.0149	2.000
Final Design (BFGS)	0.911	0.0005	0.177
Final Design (KSOPT)	0.937	0.0006	0.868

Table 4.4: Optimization results of the inviscid drag minimization problem. (Inviscid case 2.)

between the gradients has a value of 10^{-8}. We also check the gradient accuracy for the final design. The results are shown in the last two columns of table 4.3. The sensitivity gradients of the final design agree with the finite-difference gradients, with a norm of difference of 10^{-7}. The gradients of the final design are not zero in this case. In this work, the optimizer can zero the gradients only for the inverse design problem. The reason is unknown and it depends on the implementation of the optimizer.

The design results using BFGS are shown in figure 4.3. The final design is a thinner airfoil, and the shock is eliminated in the design process. The design history is shown in figure 4.4. Figures 4.5 and 4.6 show the Mach contours of the original airfoil and the final design. The thinner airfoil has reduced the shock to a compression wave. The changes in drag and lift coefficient, as well as the angle of attack, are tabulated in table 4.4. KSOPT results are shown in section 4.1.4.

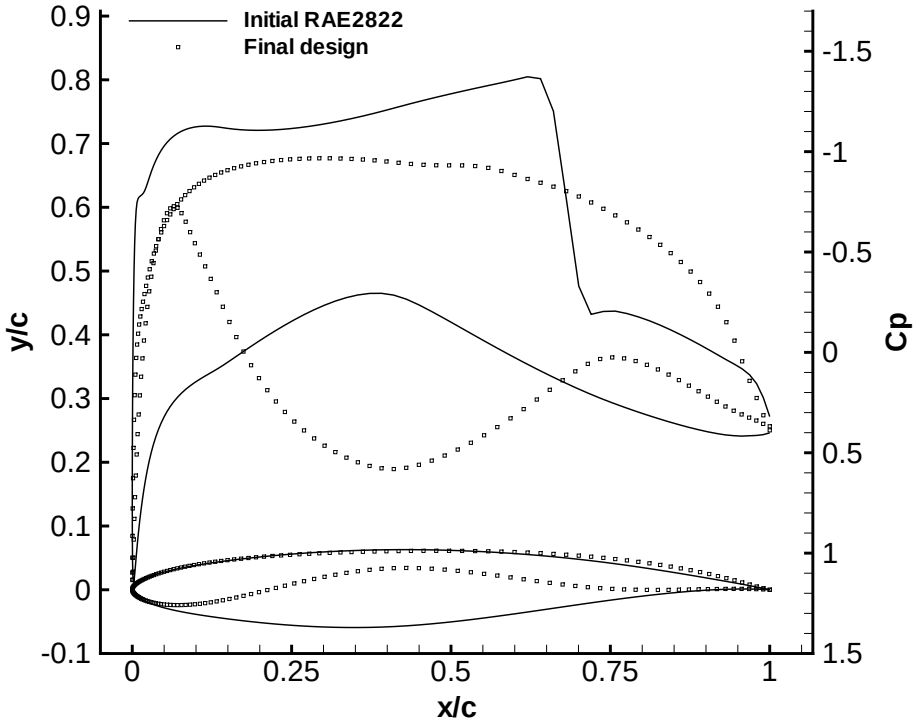


Figure 4.3: Final design of the inviscid drag minimization problem. (Inviscid case 2, BFGS.)

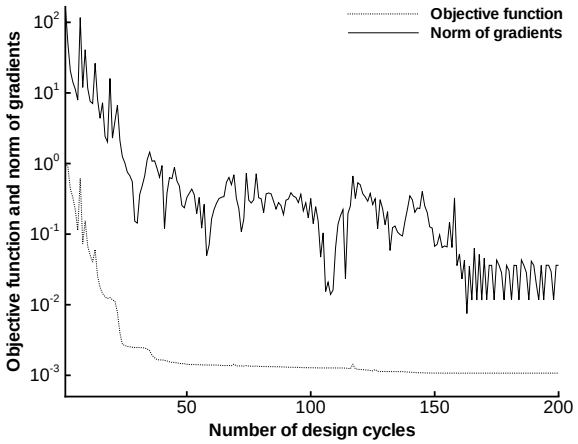


Figure 4.4: Design history of the inviscid drag minimization problem. (Inviscid case 2, BFGS.)

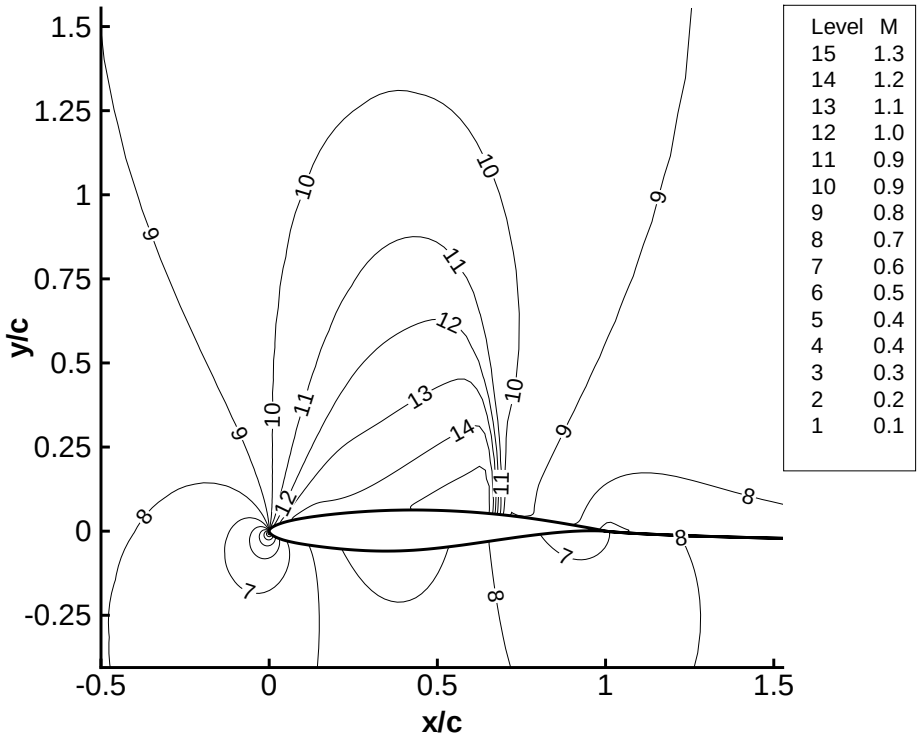


Figure 4.5: Mach contours of the original airfoil of the inviscid drag minimization problem. (Inviscid case 2, BFGS.)

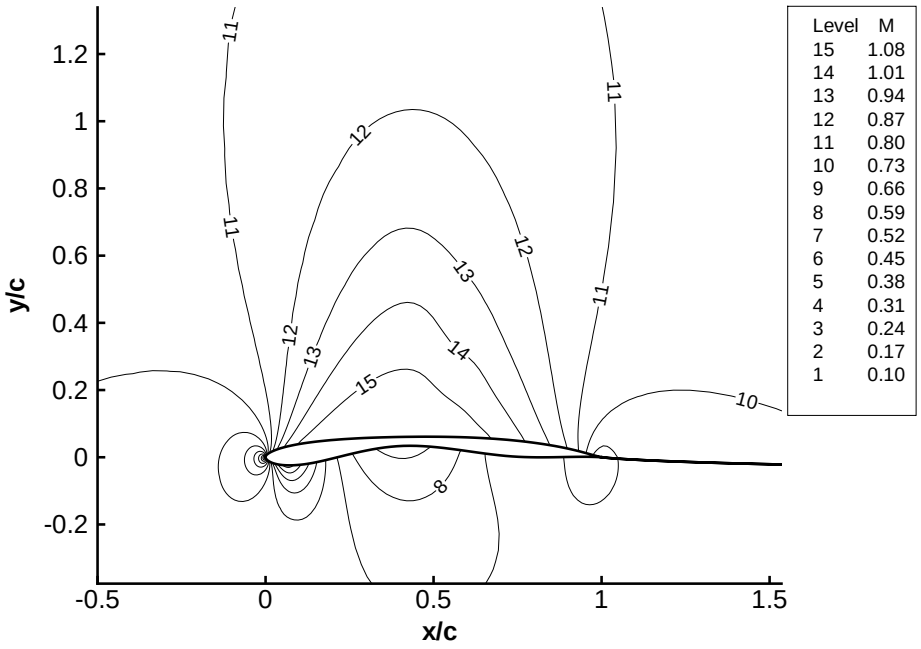


Figure 4.6: Mach contours of the final design of the inviscid drag minimization problem. (Inviscid case 2, BFGS.)

4.1.3 Lift Enhancement with Pressure Constraint

The third case is a constrained lift enhancement problem. The objective is to maximize the lift of the baseline RAE2822, with a lower limit on the surface pressure. All control points are used as design variables except the leading and trailing edges. The objective function is given by

$$J(Q,g) = \left(\frac{C_l^*}{C_l}\right)^2 + J_{\text{pres}}(Q,g) \tag{4.4}$$

The objective function is composed of two terms. The first term is the lift coefficient to be maximized, normalized with the initial lift C_l^*. The second term is the pressure constraint penalty term

$$J_{\text{pres}}(Q,g) = \begin{cases} \left[1 - \dfrac{\min(C_{p,j})}{C_p^*}\right]^2 & \text{if } \min(C_{p,j}) < C_p^* \\ 0 & \text{if } \min(C_{p,j}) \geq C_p^* \end{cases} \tag{4.5}$$

The objective function is penalized if the pressure constraint is violated. A pressure limit C_p^* of -1.2 is used.

Figure 4.7 shows the final design. No thickness constraints are implemented. The

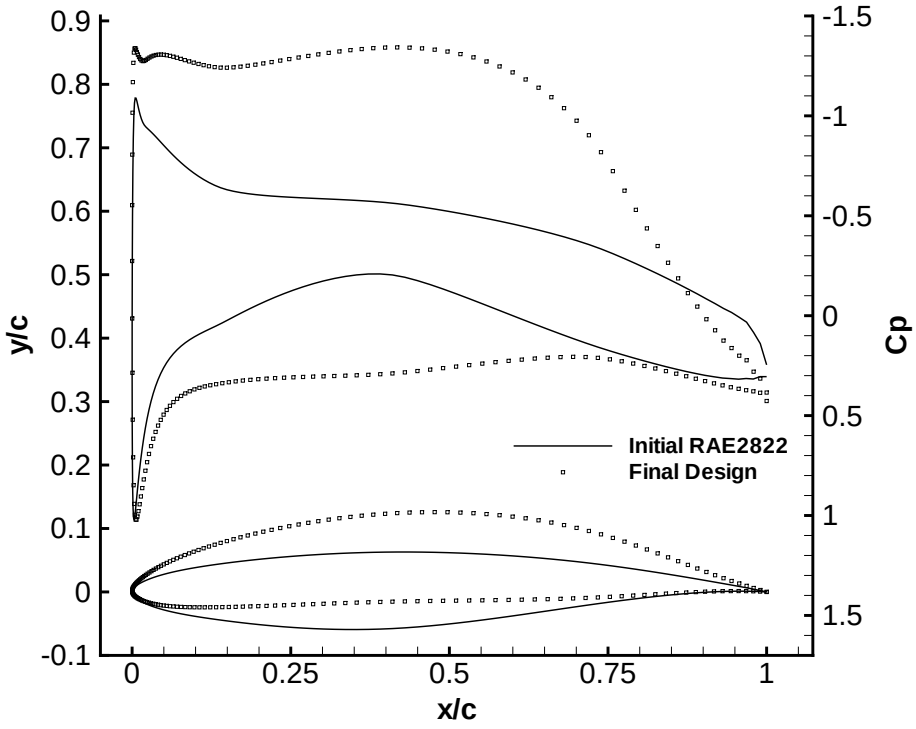


Figure 4.7: Final design of the inviscid lift enhancement problem. (Inviscid case 3, BFGS.)

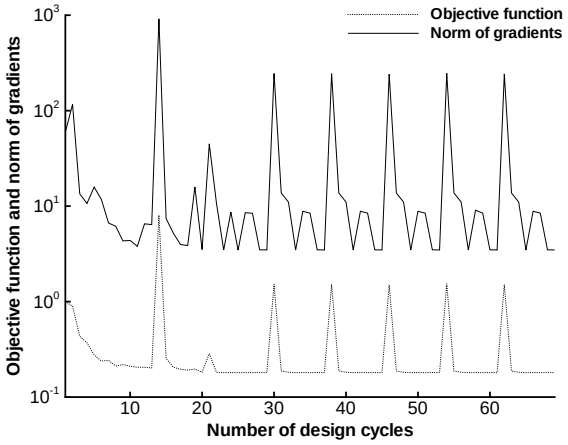


Figure 4.8: Design history of the inviscid lift enhancement problem. (Inviscid case 3, BFGS.)

	C_l	α°
Initial	0.523	2.000
Final Design (BFGS)	1.283	4.214
Final Design (KSOPT)	1.526	1.589

Table 4.5: Optimization results of the inviscid lift enhancement problem. (Inviscid case 3.)

final airfoil is a cambered airfoil at a higher angle of attack. The design history is shown in figure 4.8. It indicates that the optimization does not converge. The pattern shows that the design path is repeating itself. The spikes in the design history are large steps taken by the BFGS linesearch after a new search direction is determined. When a design is close to a constraint boundary, these large steps often violate constraints. The penalty terms cause the design to move backwards. This process repeats and it is the mechanism of the hanging process. A better design can be obtained using the optimizer KSOPT, as shown in figures 4.13 and 4.14 in the next section. This suggests that BFGS with penalty functions may be inadequate for constraint dominated optimization problems.

The changes in lift coefficient and the angle of attack during the design are tabulated in table 4.5. In this particular problem, better results are optained using KSOPT. (See figure 4.13 for the KSOPT results.) The final design with KSOPT has higher lift at a much lower angle of attack.

The optimizer does not produce a more extreme shape in figure 4.7 even though there are no thickness constraints. We fix the upper surface of the airfoil and rerun the

same design using BFGS in order to study the reason.

Figure 4.9 shows the results with the upper surface fixed. A thickness constraint weight factor w_T of 0.1 is used. (w_T is defined below.) Without thickness constraints, the design result has a cross-over in the airfoil geometry. The lower surface of the airfoil is highly curved upwards and crosses over the upper surface. We fix this problem by imposing thickness constraints.

$$\tilde{J}(Q,g) = J(Q,g) + w_T \sum_i T_i(g) \tag{4.6}$$

The thickness penalty terms are

$$T_i(g) = \begin{cases} \left(1 - \dfrac{t_i}{t_i^*}\right)^2 & \text{if } t_i < t_i^* \\ 0 & \text{if } t_i \geq t_i^* \end{cases} \tag{4.7}$$

Thickness constraints are placed at several positions along the x-axis of the airfoil geometry. We specify the thickness constraints t_i^* as the original airfoil thickness. The strength of the thickness constraint is controlled by the weight factor w_T.

Figure 4.10 shows the design history with the upper surface fixed. The norm of gradients shows quadratic convergence as shown in the figure. The reason why the design using all control points does not produce a more extreme shape using BFGS is not clear. However, we are able to design a better lower surface by fixing the upper surface for this particular case.

4.1.4 Constrained Optimization

This section shows the design results obtained using the optimizer KSOPT. Two inviscid design cases are shown: drag minimization and lift enhancement. These cases are defined in previous sections. KSOPT supports inequality constraints and requires constraint gradients. One design cycle of KSOPT consists of one gradient calculation and several flow solves. The constrained formulation of the drag minimization problem is

$$\begin{aligned} J(Q,g) &= \left(\frac{C_d}{C_d^*}\right) \\ C(Q,g) &= \left(1 - \frac{C_l}{C_l^*}\right) \leq 0 \end{aligned} \tag{4.8}$$

Figure 4.11 shows the design results using KSOPT. The final design is similar to that

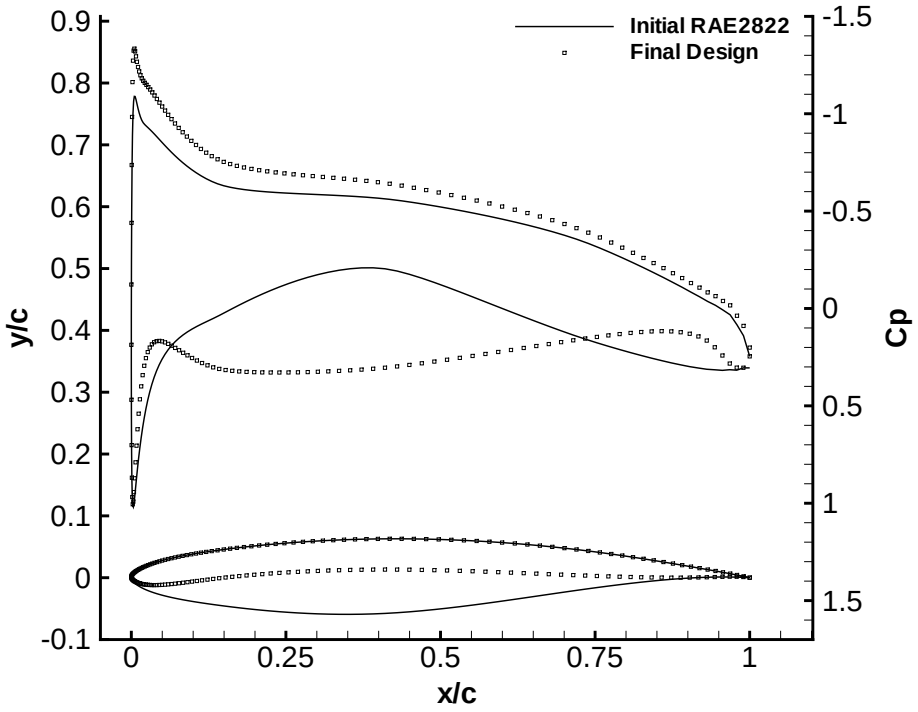


Figure 4.9: Final design of the inviscid lift enhancement problem with upper surface fixed. (Inviscid case 3, BFGS.)

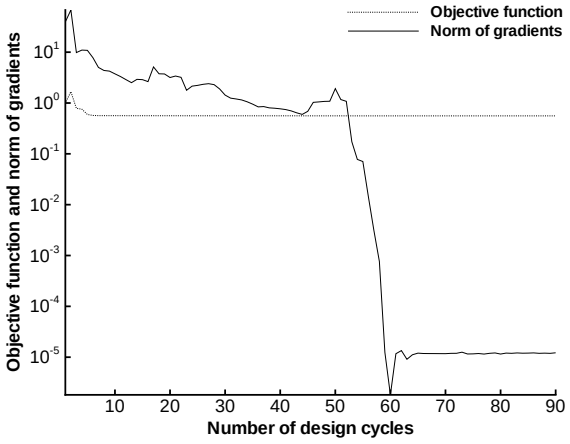


Figure 4.10: Design history of the inviscid lift enhancement problem. with upper surface fixed. (Inviscid case 3, BFGS.)

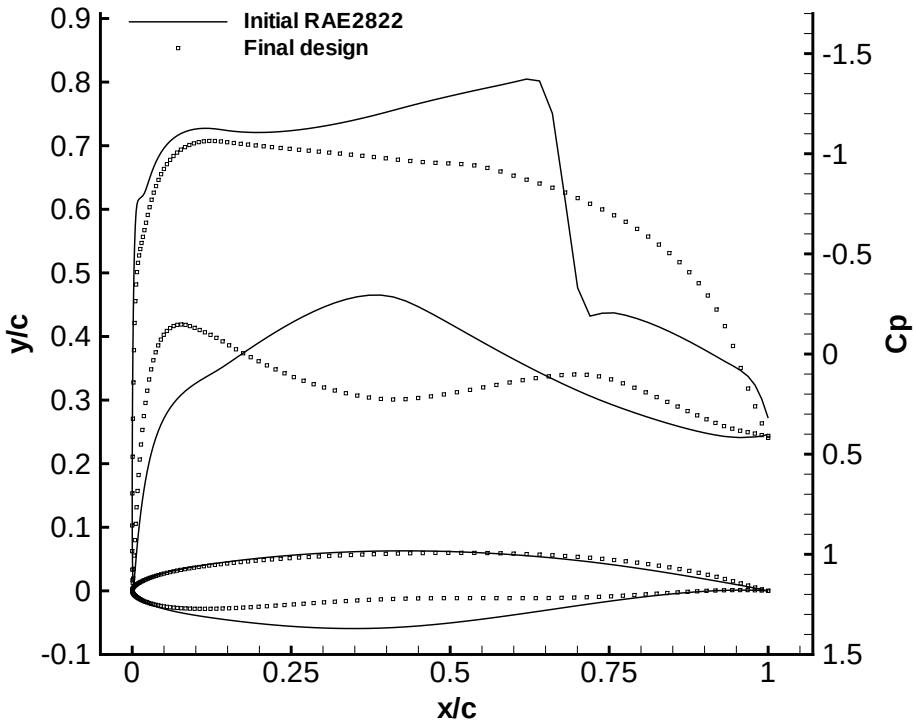


Figure 4.11: Final design of the inviscid drag minimization problem using KSOPT. (Inviscid case 2, KSOPT.)

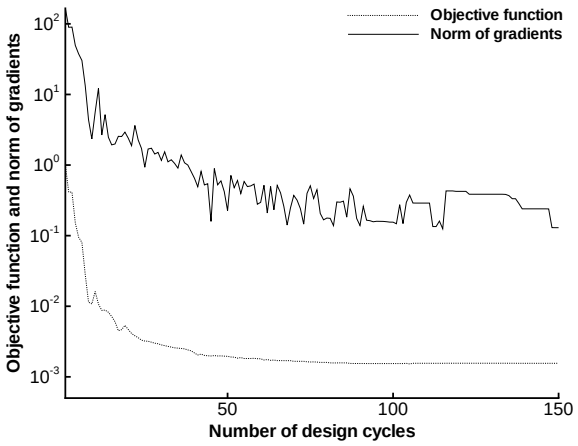


Figure 4.12: Design history of the inviscid drag minimization problem using KSOPT. (Inviscid case 2, KSOPT.)

obtained using BFGS. The changes in lift, drag and the angle of attack are shown in Table 4.4. The design history is shown in figure 4.12. The objective function gradients do not converge to zero for constrained cases.

The constrained formulation of the lift enhancement problem is

$$\begin{aligned} J(Q,g) &= \left(\frac{C_l^*}{C_l}\right) \\ C(Q,g) &= \left[1 - \frac{\min(C_{p,j})}{C_p^*}\right] \leq 0 \end{aligned} \tag{4.9}$$

Thickness constraints are incorporated using equation 4.6 with a constraint factor of 1.0. Figure 4.13 shows the design results using KSOPT. The changes in lift and the angle of attack are shown in table 4.5. In this case, KSOPT produces a better design with higher lift and less constraint violation. There is a pull-up factor in KSOPT that controls the strength of constraints. We increase this factor during the design. Too low a value produces a conservative design for which the constraints are far from violation.

The design history is shown in figure 4.14. Unlike the results obtained using BFGS, the design for this case converges smoothly.

4.1.5 Efficiency of the Inviscid Design Cases

Table 4.6 summarizes the runtime and design cycles for the inviscid design cases. BFGS is more efficient and requires fewer flow solves. In the table, case 3b denotes the lift

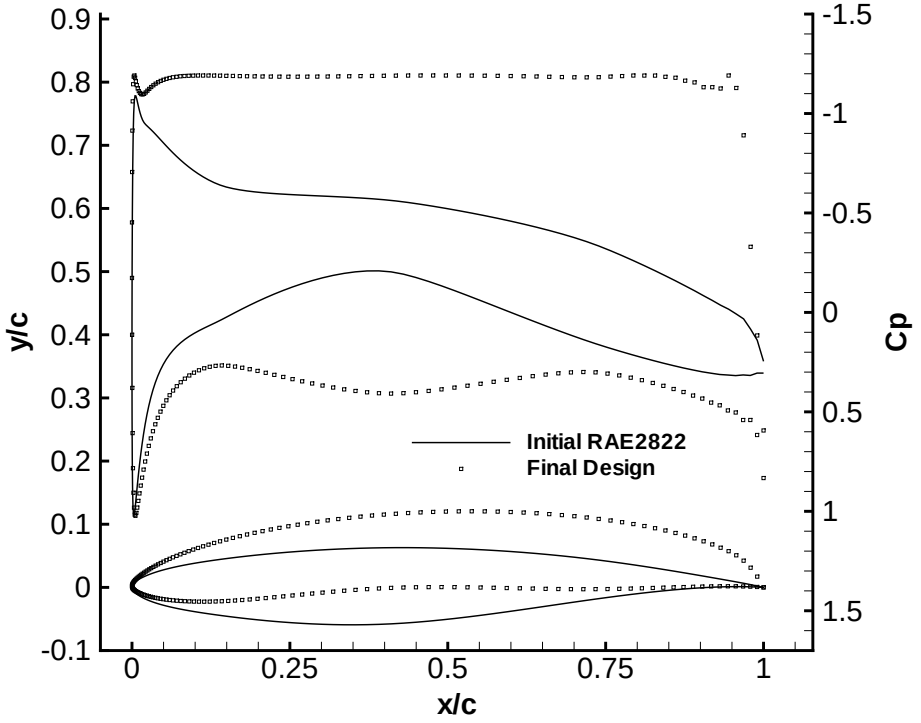


Figure 4.13: Final design of the inviscid lift enhancement problem using KSOPT. (Inviscid case 3, KSOPT.)

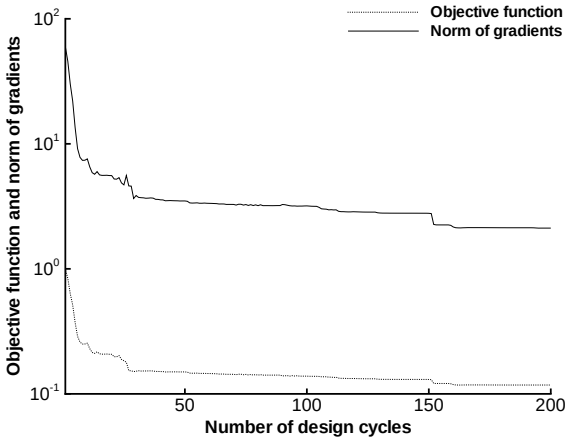


Figure 4.14: Design history of the inviscid lift enhancement problem using KSOPT. (Inviscid case 3, KSOPT.)

Case	Optimizer	Runtime	Number of design cycles	Number of flow solves
1	BFGS	1 hr 47 mins	70	70
2	BFGS	7 hrs 33 mins	199	199
3	BFGS	3 hrs 3 mins	68	68
3b	BFGS	2 hrs 28 mins	89	89
2	KSOPT	1 day 15 hrs 41 mins	149	1013
3	KSOPT	17 hrs 10 mins	199	1247

Table 4.6: Runtime and design cycles of the inviscid design cases.

Case	Runtime	Number of design cycles	Number of flow solves
1	13 mins	8	8
2	16 mins	10	10
3	17 mins	6	6
3b	12 mins	6	6
2	43 mins	8	58
3	2 hrs 19 mins	27	188

Table 4.7: Runtime and design cycles to get 90 % benefit for the inviscid design cases.

enhancement problem with the upper airfoil surface fixed.

For the cases shown, most of the design is done usually in the first few iterations. Table 4.7 shows the required runtime and design cycles to achieve 90 % benefit of the design. This is defined as 90 % reduction of the difference between the initial and final objective functions. As shown in the table, practical designs can be achieved usually in less then ten design cycles. The rest of the computational work does not contribute much to practical engineering design.

4.2 Viscous Design Results

Four design cases are presented for viscous airfoil design. Table 4.8 summarizes the number of design variables and the free-stream flow conditions for the design cases. Here, tr denotes the laminar to turbulent transition points on both upper and lower surfaces (percentage chord). Only BFGS results are presented for viscous flows.

Case	Title	n_{dv}	M_∞	α_{initial}	Re	tr
1	Inverse design	$12 + \alpha$	0.30	2.0	2.88×10^6	1%
2	Drag minimization with lift constraint	$12 + \alpha$	0.74	2.0	9.0×10^6	5%
3	Maximize lift-to-drag ratio	$12 + \alpha$	0.70	2.2	9.0×10^6	5%
4	Lift enhancement with pressure constraint	$16 + \alpha$	0.30	2.0	2.88×10^6	5%

Table 4.8: Viscous design case parameters.

Control point	Initial NACA0012	
	Sensitivity mf(2)	Finite differences
3	-0.187631 E+03	-0.187597 E+03
4	-0.117027 E+03	-0.117013 E+03
5	-0.494222 E+02	-0.493876 E+02
6	0.684423 E+01	0.688854 E+01
7	0.303036 E+02	0.303346 E+02
8	0.353796 E+02	0.353948 E+02
12	0.254680 E+03	0.254828 E+03
13	0.766349 E+02	0.767251 E+02
14	-0.114638 E+02	-0.113366 E+02
15	-0.449036 E+02	-0.447748 E+02
16	-0.771329 E+02	-0.770579 E+02
17	-0.111233 E+03	-0.111208 E+03
α	-0.510418 E+01	-0.510418 E+01

Table 4.9: Verification of gradient accuracy for the viscous inverse design problem. (Viscous case 1.)

4.2.1 Inverse Design

The first case is the inverse design problem from NACA0012 to RAE2822. The objective function is given by equation 4.1. We verify the gradients before running the design. The results are shown in table 4.9. The leading and trailing edges of the airfoil are fixed similar to the inviscid case.

The design results using BFGS are shown in figure 4.15. Figure 4.16 shows the design history. The objective function reduces 16 orders of magnitude. The norm of the gradients is reduced by 8 orders of magnitude.

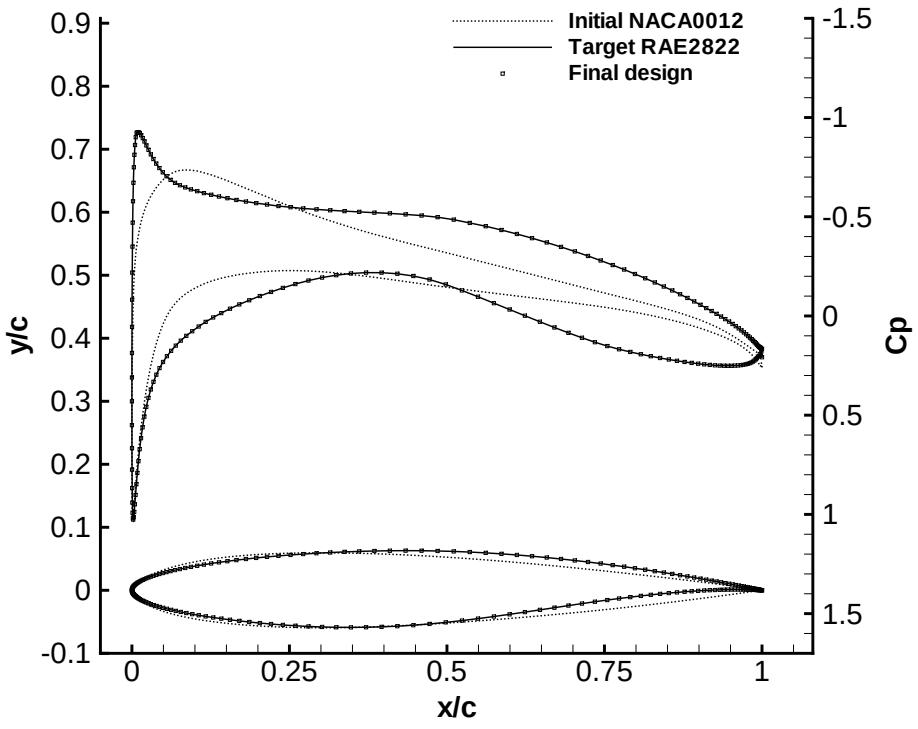


Figure 4.15: Final design of the viscous inverse design problem. (Viscous case 1, BFGS.)

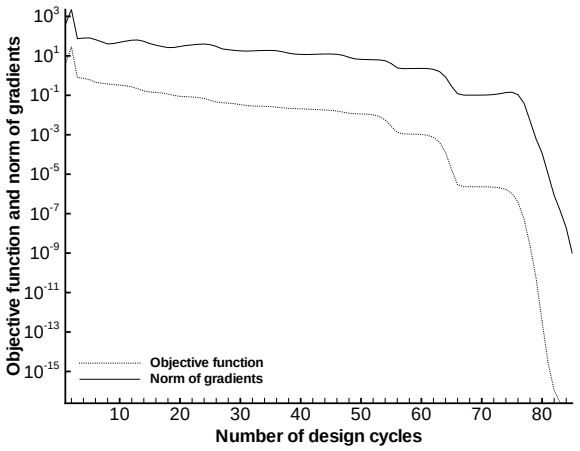


Figure 4.16: Design history of the viscous inverse design problem. (Viscous case 1, BFGS.)

Control point	Initial	
	Sensitivity mf(2)	Finite differences
3	0.520632 E+02	0.520631 E+02
4	0.280185 E+02	0.280184 E+02
5	0.148741 E+02	0.148741 E+02
6	0.643658 E+01	0.643657 E+01
7	0.183497 E+01	0.183497 E+01
8	0.289205 E-02	0.288866 E-02
12	-0.440524 E+02	-0.440536 E+02
13	0.133756 E+03	0.133756 E+03
14	0.294035 E+02	0.294035 E+02
15	-0.202665 E+02	-0.202666 E+02
16	0.101094 E+02	0.101094 E+02
17	0.914149 E+01	0.914151 E+01
α	0.105821 E+01	0.105821 E+01

Table 4.10: Verification of gradient accuracy for the viscous drag minimization problem. (Viscous case 2.)

4.2.2 Drag Minimization with Lift Constraint

The second case is the drag minimization problem. The objective function is given by equation 4.2. We fix three control points at the leading edge and four at the trailing edge, similar to the inviscid case. We verify the gradients before running the design. The results are shown in table 4.10.

The design using BFGS has a cross-over in the airfoil geometry. We impose thickness constraints with a weight factor of 0.03. This is different from the inviscid case, which does not require a thickness constraint. Viscous effects are important for accurate drag prediction. The viscous flow model should be used for drag-related design problems.

The final design is a thin airfoil as shown in figure 4.17. The design result depends on the thickness constraint. Figure 4.18 shows the design history. As in the inviscid case, the gradients do not converge to zero. Figures 4.19 and 4.20 show the Mach contours of the original airfoil and the final design. Viscous effects can be seen in these figures, at the boundary layers and wakes. The shock is eliminated in the final design. The changes in drag, lift, and the angle of attack are tabulated in table 4.11.

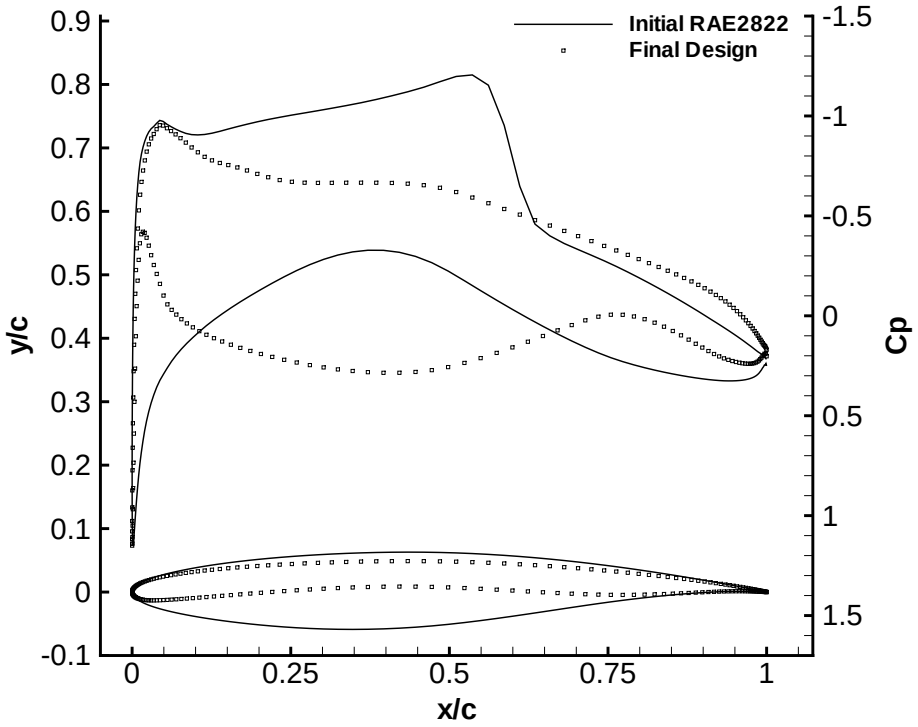


Figure 4.17: Final design of the viscous drag minimization problem. (Viscous case 2, BFGS.)

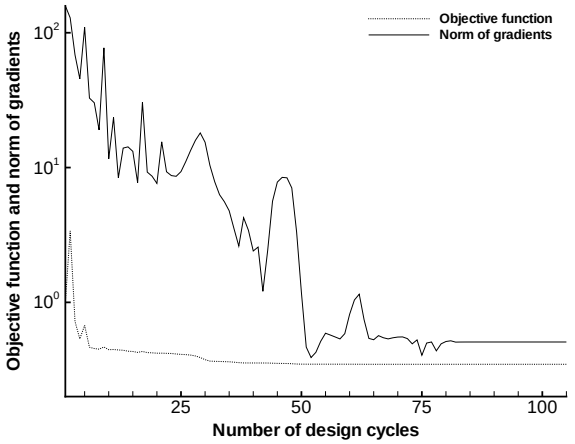


Figure 4.18: Design history of the viscous drag minimization problem. (Viscous case 2, BFGS.)

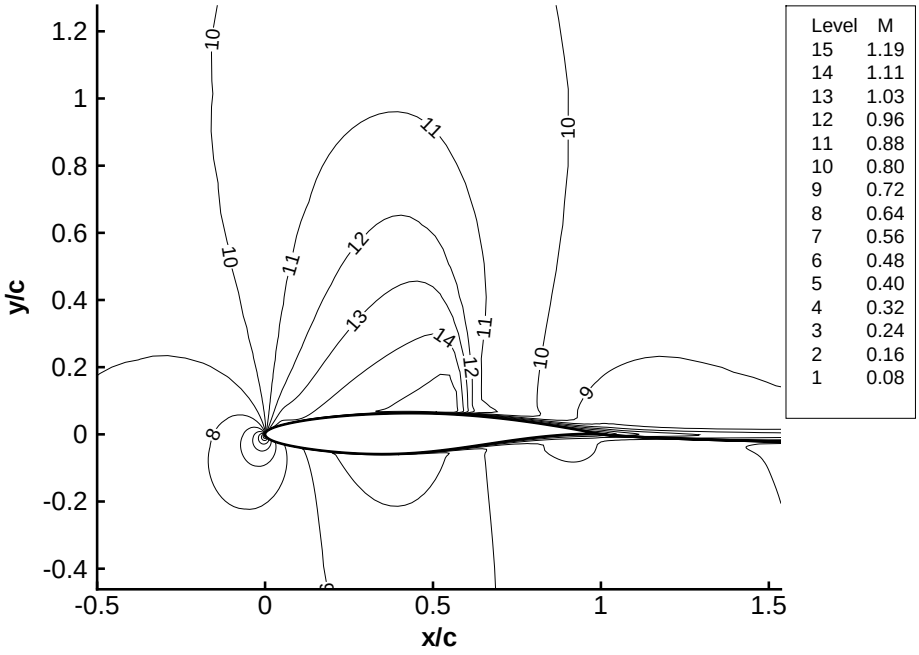


Figure 4.19: Mach contours of the original airfoil of the viscous drag minimization problem. (Viscous case 2, BFGS.)

	C_l	C_d	α°
Initial	0.712	0.015	2.000
Final Design	0.658	0.008	1.036

Table 4.11: Optimization results of the viscous drag minimization problem. (Viscous case 2.)

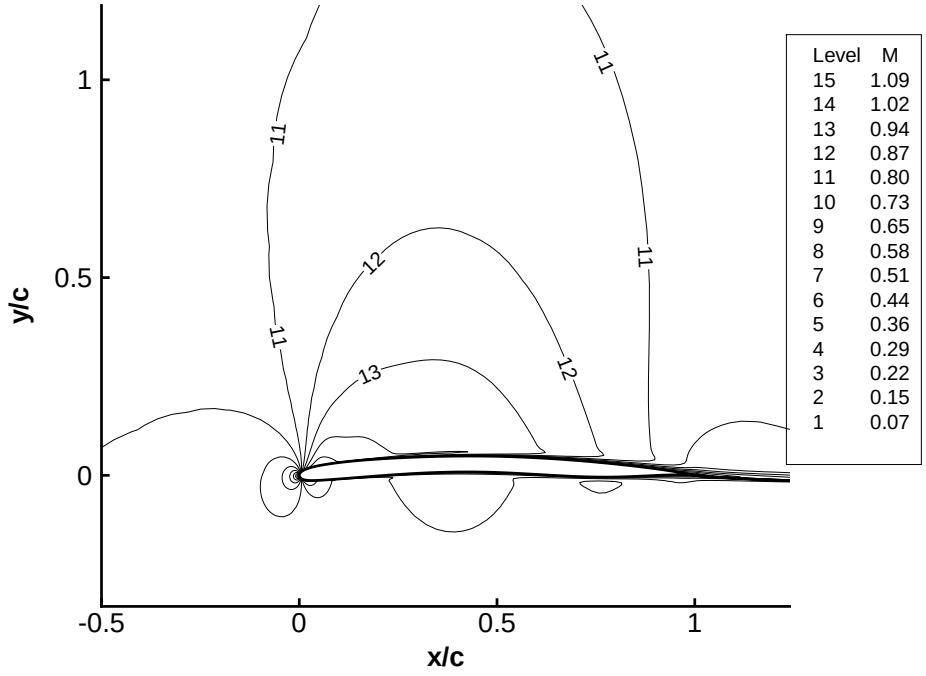


Figure 4.20: Mach contours of the final design of the viscous drag minimization problem. (Viscous case 2, BFGS.)

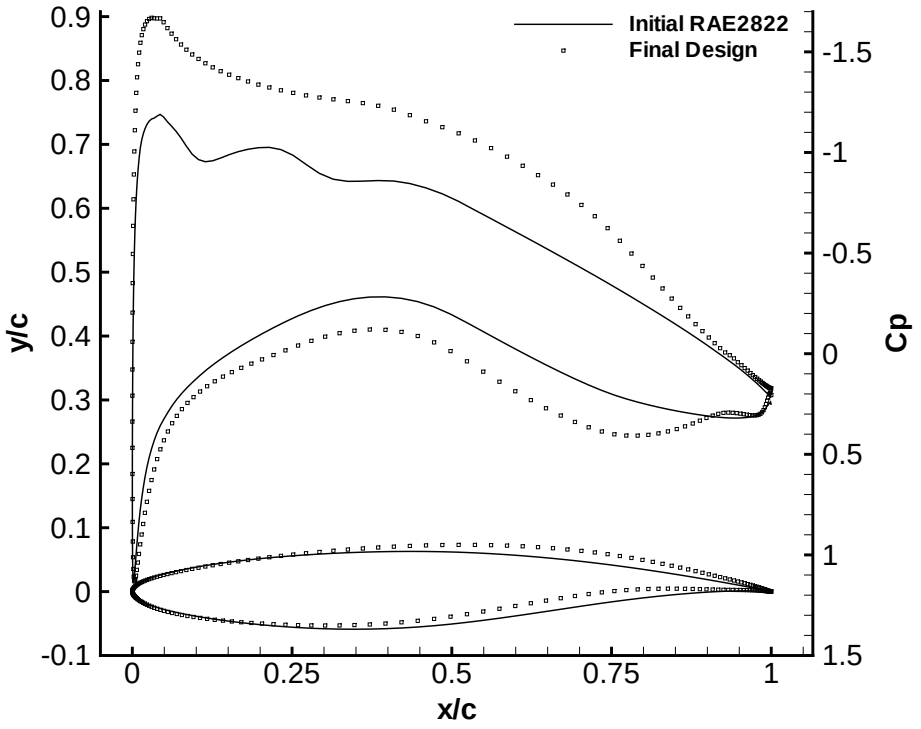


Figure 4.21: Final design for viscous lift-to-drag ratio maximization problem. (Viscous case 3, BFGS.)

4.2.3 Maximize Lift-to-Drag Ratio

The third case is to maximize the lift-to-drag ratio. The objective function is the drag-to-lift ratio to be minimized.

$$J(Q, g) = \frac{C_d}{C_l} \tag{4.10}$$

The design using BFGS has a cross-over in the airfoil geometry. Thickness constraints are imposed with a weight factor of 0.1. The final design is shown in figure 4.21. The design history is shown in figure 4.22. The changes in drag, lift and the angle of attack are tabulated in table 4.12.

4.2.4 Lift Enhancement with Pressure Constraint

The fourth case is the lift enhancement problem. The objective function is given by equation 4.4. All control points are used as design variables except the leading and trailing edges. No thickness constraints are imposed. The design results are shown in figure 4.23.

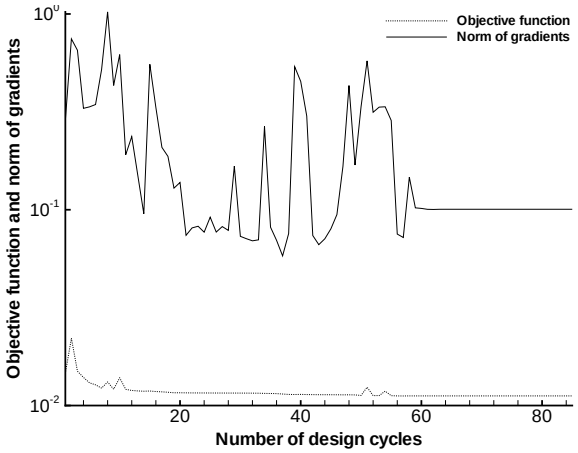


Figure 4.22: Design history for viscous lift-to-drag ratio maximization problem. (Viscous case 3, BFGS.)

	C_l	C_d	α°
Initial	0.678	0.010	2.200
Final Design	1.118	0.012	3.133

Table 4.12: Optimization results for viscous lift-to-drag ratio maximization problem. (Viscous case 3.)

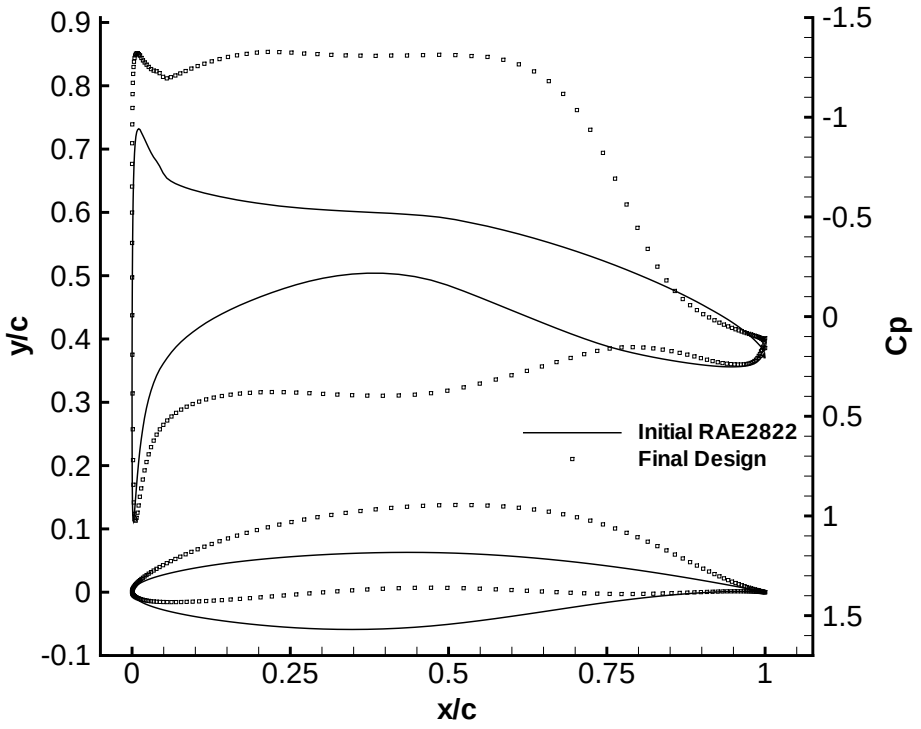


Figure 4.23: Final design of the viscous lift enhancement problem. (Viscous case 4, BFGS.)

	C_l	α°
Initial	0.442	2.000
Final Design	1.322	5.552

Table 4.13: Optimization results of the viscous lift enhancement problem. (Viscous case 4.)

The final design is similar to the inviscid design. Figure 4.24 shows the design history. The design does not converge for the same reason as the inviscid case. The changes in lift coefficient and the angle of attack during the design are tabulated in table 4.13. The final design has a higher angle of attack than the inviscid design.

The design results with the upper airfoil surface fixed are shown in figure 4.25. Similar results to the inviscid case are obtained. The design history is shown in figure 4.26.

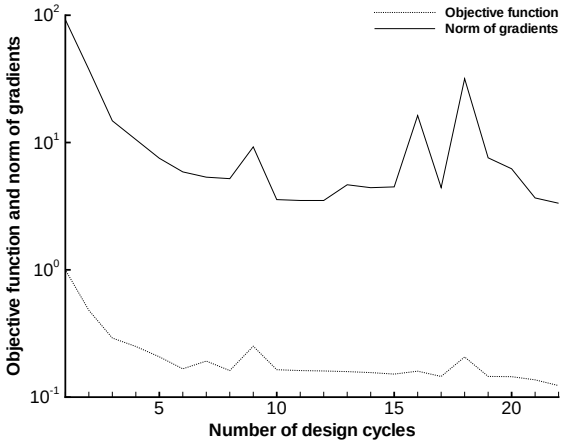


Figure 4.24: Design history of the viscous lift enhancement problem. (Viscous case 4, BFGS.)

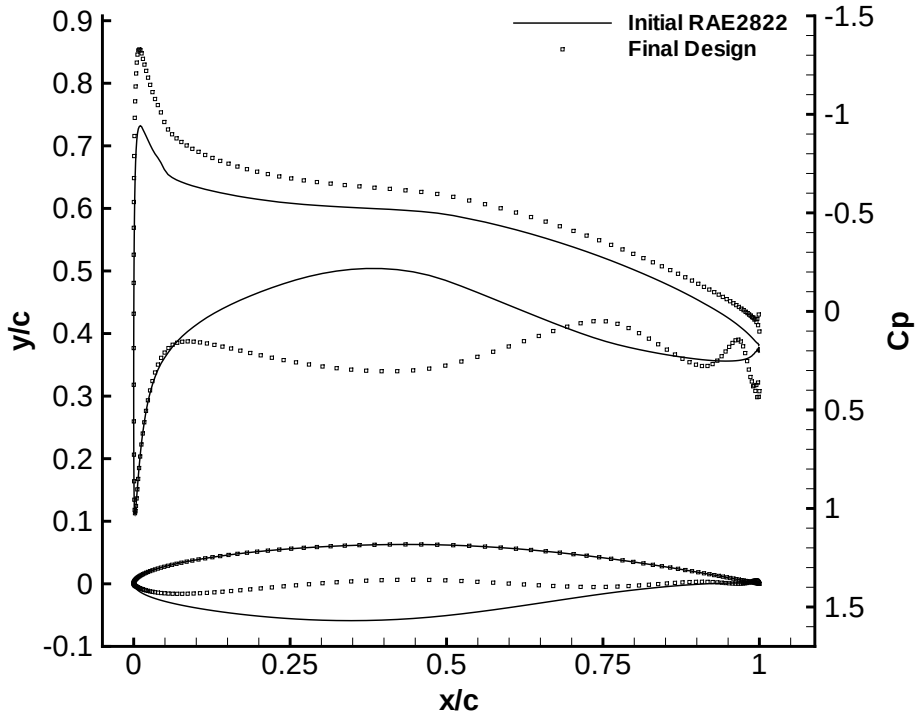


Figure 4.25: Final design of the viscous lift enhancement problem with upper surface fixed. (Viscous case 4, BFGS.)

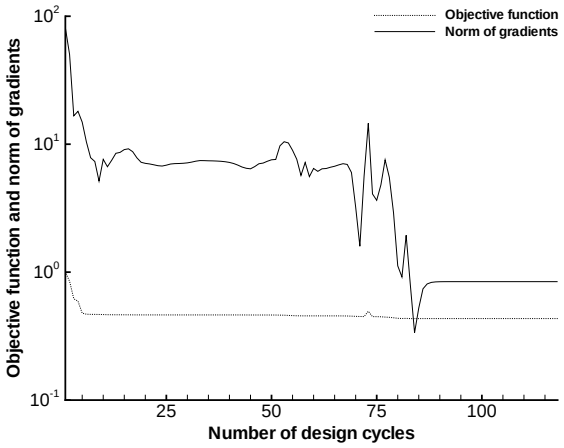


Figure 4.26: Design history of the viscous lift enhancement problem with upper surface fixed. (Viscous case 4, BFGS.)

Case	Runtime	Number of design cycles	Number of flow solves
1	8 hrs 8 mins	84	84
2	8 hrs 25 mins	104	104
3	7 hrs 14 mins	84	84
4	3 hrs 34 mins	21	21
4b	11 hrs 20 mins	117	117

Table 4.14: Runtime and design cycles of the viscous design cases.

4.2.5 Efficiency of the Viscous Design Cases

Table 4.14 summarizes the runtime and design cycles for the viscous design cases. In the table, case 4b denotes the lift enhancement problem with the upper airfoil surface fixed. Viscous designs take longer to run. Both the flow solver and the gradient calculation run longer. This is due to the higher number of grid nodes and the turbulence model. Figure 4.15 shows the required runtime and design cycles to achieve 90 % benefit of the design.

Case	Runtime	Number of design cycles	Number of flow solves
1	50 mins	8	8
2	1 hrs 49 mins	20	20
3	3 hrs 18 mins	38	38
4	1 hrs 22 mins	8	8
4b	45 mins	8	8

Table 4.15: Runtime and design cycles to get 90 % benefit for the viscous design cases.

Chapter 5

Conclusions

The flow sensitivity method using the matrix-free GMRES approach is presented for CFD design problems. The sensitivity approach is capable of calculating accurate gradients comparable to finite-difference gradients at a much lower cost. The flow sensitivity approach is less efficient when compared to the adjoint approach, since it depends on the number of design variables used, but is potentially more accurate due to the matrix-free formulation. If the number of constraint gradients required is high, the flow sensitivity approach can be efficient.

Both unconstrained and constrained design cases are studied. The algorithm described is capable of airfoil design for a wide range of objective functions and constraints. The optimizer KSOPT is found to be more robust than the optimizer BFGS, especially for constrained problems, but BFGS is often faster.

References

[1] Hicks, R. M., Murman, E. M., and Vanderplaats, G. N., "An Assessment of Airfoil Design by Numerical Optimization," Tech. rep., NASA TM X-3092, July 1974.

[2] Jameson, A., "Aerodynamic Design via Control Theory," Tech. Rep. NASA CR-181749, Institute for Computer Applications in Science and Engineering, 1988.

[3] Gunzburger, M. D., "Introduction into Mathematical Aspects of Flow Control and Optimization," *von Karman Institute for Fluid Dynamics Lecture Series*, 1997-05.

[4] Frank, P. D. and Shubin, G. R., "A Comparison of Optimization-Based Approaches for a Model Computational Aerodynamics Design Problem," *Journal of Computational Physics*, Vol. 98, 1992, pp. 74–89.

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

[6] Wrenn, G. A., "An Indirect Method for Numerical Optimization Using the Kreisselmeier-Steinhauser Function," Tech. rep., NASA CR 4220, March 1989.

[7] Nielsen, E. J. and Anderson, W. K., "Aerodynamic Design Optimization on Unstructured Meshes Using the Navier Stokes Equations," *AIAA Journal*, Vol. 37, No. 11, 1999, pp. 1411–1419.

[8] Elliott, J. and Peraire, J., "Constrained, multipoint shape optimisation for complex 3D configurations," *Aeronautical Journal*, Vol. 102, No. 1017, 1998, pp. 365–376.

[9] Turgeon, E., Pelletier, D., and Borggaard, J., "A Continuous Sensitivity Equation Approach to Optimal Design in Mixed Convection," *AIAA Paper 99–3625*, 1999.

[10] Newman, J. C., Anderson, W. K., and Whitfield, D. L., "Multidisciplinary Sensitivity Derivatives Using Complex Variables," *Mississippi State University MSSU-COE-ERC, 98-08*, 1998.

[11] Anderson, W. K., Newman, J. C., Whitfield, D. L., and Nielsen, E. J., "Sensitivity Analysis for the Navier-Stokes Equations on Unstructured Meshes Using Complex Variables," *AIAA Paper 99–3294*, 1999.

[12] Nemec, M., "Nozzle Shape Optimization ARC1D and OPTIMA," Tech. rep., UTIAS report, 2000.

[13] Pulliam, T. H., "Efficient Solution Methods for the Navier-Stokes Equations," Tech. rep., Lecture Notes for the Von Karman Institute for Fluid Dynamics Lecture Series, Brussels, Jan. 1986.

[14] Nadarajah, S. and Jameson, A., "A Comparison of the Continuous and Discrete Adjoint Approach to Automatic Aerodynamic Optimization," *AIAA Paper 2000–0667*, 2000.

[15] Baldwin, B. S. and Lomax, H., "Thin Layer Approximation and Algebraic Model for Separated Turbulent Flows," *AIAA Paper 78–257*, 1978.

[16] Wilkinson, A. R. and Zingg, D. W., "AMBER 2d, User's Guide," Tech. rep., University of Toronto Institute for Aerospace Studies, 1993.

[17] Nemec, M. and Zingg, D. W., "Towards Efficient Aerodynamic Shape Optimization Using the Discrete Adjoint Method," *AIAA Paper 2001–2532*, 2001.

[18] Jameson, A., Schmidt, W., and Turkel, E., "Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time Stepping," *AIAA Paper 81-1259*, 1981.

[19] Pueyo, A., *An Efficient Newton-Krylov Method for the Euler and Navier-Stokes Equations*, Ph.D. thesis, University of Toronto, 1998.

[20] Pueyo, A. and Zingg, D., "Efficient Newton-Krylov Solver for Aerodynamic Computations," *AIAA Journal*, Vol. 36, No. 11, 1998, pp. 1991–1997.

Appendix A

The One-Dimensional Flow Jacobian

The Jacobian of the steady quasi-one-dimensional flow equations is given by

$$\begin{aligned} \frac{\partial \mathcal{F}}{\partial \mathcal{Q}} &= -\frac{d}{dx}\frac{\partial \hat{E}}{\partial \mathcal{Q}} + \frac{\partial \hat{H}}{\partial \mathcal{Q}} \\ &= -\frac{d}{dx}\hat{J}_E + \hat{J}_H \end{aligned} \tag{A.1}$$

The Jacobian of the inviscid fluxes is

$$J_E = \frac{\partial E}{\partial \mathcal{Q}} = \begin{bmatrix} 0 & 1 & 0 \\ -(3-\gamma)\frac{u^2}{2} & (3-\gamma)u & \gamma - 1 \\ (\gamma-1)u^3 - \gamma u\frac{e}{\rho} & \gamma\frac{e}{\rho} - 3\frac{\gamma-1}{2}u^2 & \gamma u \end{bmatrix} \tag{A.2}$$

$$= \begin{bmatrix} 0 & 1 & 0 \\ -\frac{1}{2}\frac{(3-\gamma)q_2^2}{q_1^2} & \frac{(3-\gamma)q_2}{q_1} & \gamma - 1 \\ \frac{(\gamma-1)q_2^3}{q_1^3} - \frac{\gamma q_2 q_3}{q_1^2} & \frac{\gamma q_3}{q_1} - \frac{3}{2}\frac{(\gamma-1)q_2^2}{q_1^2} & \frac{\gamma q_2}{q_1} \end{bmatrix} \tag{A.3}$$

and

$$\hat{J}_E = J_E S \tag{A.4}$$

The Jacobian of the source term is

$$J_H = \frac{\partial H}{\partial \mathcal{Q}} = \begin{bmatrix} 0 & 0 & 0 \\ (\gamma-1)\frac{u^2}{2} & -(\gamma-1)u & \gamma - 1 \\ 0 & 0 & 0 \end{bmatrix} \tag{A.5}$$

$$= \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{2}\frac{(\gamma-1)q_2^2}{q_1^2} & -\frac{(\gamma-1)q_2}{q_1} & \gamma-1 \\ 0 & 0 & 0 \end{bmatrix} \tag{A.6}$$

and

$$\hat{J}_H = J_H \frac{dS}{dx} \tag{A.7}$$

Appendix B

Continuous Adjoint Formulation for the Inverse Problem

The continuous Lagrangian for the inverse problem can be written as

$$\mathcal{L}(\mathcal{Q}, g, \psi) = \mathcal{J}(\mathcal{Q}, g) - < \psi, \mathcal{F}(\mathcal{Q}, g) > \tag{B.1}$$

where $<,>$ denotes an inner product.

B.1 Variation of the Objective Function

The objective function is given by

$$\mathcal{J} = \int_a^b \frac{1}{2}(\frac{q_2}{q_1} - u^\star)^2 dx \tag{B.2}$$

Define

$$M = \frac{1}{2}(\frac{q_2}{q_1} - u^\star)^2 \tag{B.3}$$

The variation of the objective function with respect to the flow variables is given by

$$\delta\mathcal{J} = \int_a^b \frac{\partial M}{\partial \mathcal{Q}} \tilde{\mathcal{Q}} dx \tag{B.4}$$

B.2 Variation of the Inner Product

Recall the flow equations from Appendix A,

$$\mathcal{F} = -\frac{d}{dx}(ES) + H\frac{dS}{dx} \tag{B.5}$$

The inner product between the flow equations and the adjoint variables is given by

$$\begin{aligned} \mathcal{I} &= <\psi, \mathcal{F}> = \int_a^b \psi^T \mathcal{F} dx \\ &= \int_a^b -\psi^T \frac{d}{dx}(ES)dx + \int_a^b \psi^T H \frac{dS}{dx} dx \end{aligned} \tag{B.6}$$

The variation of the inner product is given by

$$\delta\mathcal{I} = \int_a^b -\psi^T \frac{d}{dx}(J_E \tilde{\mathcal{Q}} S)dx + \int_a^b \psi^T J_H \tilde{\mathcal{Q}} \frac{dS}{dx} dx \tag{B.7}$$

Integrate by parts the first term to isolate the variation of the flow variables,

$$\int_a^b -\psi^T \frac{d}{dx}(J_E \tilde{\mathcal{Q}} S)dx = -\psi^T (J_E \tilde{\mathcal{Q}} S)\Big|_a^b + \int_a^b (\frac{d\psi}{dx})^T J_E \tilde{\mathcal{Q}} S dx \tag{B.8}$$

So, the variation of the inner product can be written as

$$\delta\mathcal{I} = -\psi^T (J_E \tilde{\mathcal{Q}} S)\Big|_a^b + \int_a^b (\frac{d\psi}{dx})^T J_E \tilde{\mathcal{Q}} S dx + \int_a^b \psi^T J_H \tilde{\mathcal{Q}} \frac{dS}{dx} dx \tag{B.9}$$

B.3 The Continuous Adjoint Equations

The continuous adjoint equation is obtained by setting the variation of the Lagrangian with respect to the flow variables to zero

$$\delta\mathcal{L} = \delta\mathcal{J} - \delta\mathcal{I} = 0 \tag{B.10}$$

$$\delta\mathcal{L} = \int_a^b \{\frac{\partial M}{\partial \mathcal{Q}} - (\frac{d\psi}{dx})^T J_E S - \psi^T J_H \frac{dS}{dx}\} \tilde{\mathcal{Q}} dx + \psi^T (J_E \tilde{\mathcal{Q}} S)\Big|_a^b = 0 \tag{B.11}$$

Since $\tilde{\mathcal{Q}}$ is arbitrary, the continuous adjoint equation is given by

$$S(J_E)^T \frac{d\psi}{dx} + \frac{dS}{dx}(J_H)^T \psi = (\frac{\partial M}{\partial \mathcal{Q}})^T \tag{B.12}$$

with boundary conditions

$$\begin{aligned} \psi(a) &= 0 \\ \psi(b) &= 0 \end{aligned} \tag{B.13}$$

Appendix C

Equivalence of the Continuous Adjoint and Discrete adjoint

The continuous adjoint formulation is equivalent to the discrete adjoint formulation for the inverse problem with fixed flow boundary conditions and fixed boundary control points.

C.1 Discretize the Continuous Adjoint Equations

The continuous adjoint equations are given by

$$A^T \frac{d\psi}{dx} + B^T \psi = \left(\frac{\partial M}{\partial \mathcal{Q}}\right)^T \tag{C.1}$$

Here, $A = \hat{J_E}$ and $B = \hat{J_H}$. Using second-order centered differences to discretize the derivative, the following discrete form is obtained.

$$-\frac{1}{2\Delta x} A_j^T \psi_{j-1} + B_j^T \psi_j + \frac{1}{2\Delta x} A_j^T \psi_{j+1} = \left(\frac{\partial M}{\partial Q}\right)_j^T \tag{C.2}$$

$$\begin{bmatrix} & & \\ -\frac{1}{2\Delta x} A_j^T & B_j^T & \frac{1}{2\Delta x} A_j^T \\ & & \end{bmatrix} \begin{bmatrix} \psi_{j-1} \\ \psi_j \\ \psi_{j+1} \end{bmatrix} = \begin{bmatrix} \\ (\frac{\partial M}{\partial Q})_j^T \\ \\ \end{bmatrix} \tag{C.3}$$

C.2 Equivalence of the Discrete and Adjoint Equations

The discrete adjoint equations are given by

$$(\frac{\partial R}{\partial Q})^T \psi = (\frac{\partial J}{\partial Q})^T \tag{C.4}$$

Consider the left-hand side.

$$\frac{\partial R}{\partial Q} = -\frac{d}{dx}\hat{J_E} + \hat{J_H} \tag{C.5}$$

$$(\frac{\partial R}{\partial Q})_j = -\frac{A_{j+1} - A_{j-1}}{2\Delta x} + B_j \tag{C.6}$$

$$= \begin{bmatrix} & -\frac{1}{2\Delta x}A_j & \\ \frac{1}{2\Delta x}A_{j-1} & B_j & -\frac{1}{2\Delta x}A_{j+1} \\ & \frac{1}{2\Delta x}A_j & \end{bmatrix} \tag{C.7}$$

The transpose of the derivative is equivalent to the left-hand side of equation C.3.

$$(\frac{\partial R}{\partial Q})_j^T = \begin{bmatrix} & & \\ -\frac{1}{2\Delta x}A_j^T & B_j^T & \frac{1}{2\Delta x}A_j^T \\ & & \end{bmatrix} \tag{C.8}$$

Appendix D

Parametric Study of the Gradient Solver

D.1 Matrix Sensitivity Approach

Table D.1 shows a parametric study of the gradient solver for the matrix sensitivity approach. We study the parameters for a transonic inviscid flow over the RAE2822 airfoil with two design variables (control points 10 and 11). In this case, the second-order Jacobian has 1,377,728 non-zero elements, and the first-order Jacobian (preconditioner) has 780,224 non-zero elements. In the table, GMRES time represents the time required to reduce the residual of the sensitivity equations by eight orders of magnitude, and BFILU time is the time required for the incomplete LU factorization. There are two columns of GMRES time and GMRES iterations, since there are two sensitivity equations for two design variables. The last column shows the number of non-zero elements (n_{nz}) in the preconditioner after incomplete factorization. This number depends on the level of fill.

D.2 Matrix-Free Sensitivity Approach

Table D.2 shows a parametric study of the gradient solver for the matrix-free sensitivity approach. The second-order Jacobian is not required in this case and the first-order Jacobian (preconditioner) has 780,224 non-zero elements.

σ	BFILU(n)	GMRES time (seconds)		BFILU time (seconds)	Iterations		n_{nz} BFILU
2.0	1	DNC					
3.0		39.84	35.64	1.293	136	123	1078864
4.0		39.38	35.81	1.291	138	125	1078864
5.0		41.00	36.33	1.287	143	127	1078864
6.0		46.49	38.09	1.287	159	133	1078864
2.0	2	DNC					
3.0		DNC					
4.0		28.44	27.28	2.002	88	85	1370224
5.0		27.78	27.45	2.008	86	85	1370224
6.0		28.81	28.14	2.001	89	87	1370224
10.0		39.99	38.95	2.005	130	126	1370224
2.0	3	91.36	70.89	2.731	265	208	1664096
3.0		18.53	17.96	2.728	60	58	1664096
4.0		25.13	25.32	2.741	76	76	1664096
5.0		37.82	38.80	2.728	115	117	1664096
6.0		28.10	28.76	2.732	83	84	1664096
10.0		38.06	38.59	2.728	116	117	1664096
2.0	4	DNC					
3.0		15.41	15.69	3.721	49	49	1953936
4.0		DNC					
5.0		92.41	91.75	3.723	256	252	1953936
6.0		25.53	26.17	3.735	73	74	1953936
10.0		38.36	38.38	3.722	109	109	1953936
2.0	5	DNC					
3.0		15.65	14.60	4.759	47	44	2244192
4.0		18.72	18.02	4.789	54	52	2244192
5.0		27.91	27.53	4.753	76	75	2244192
6.0		24.60	24.39	4.776	68	67	2244192
10.0		39.29	38.84	4.753	106	104	2244192
2.0	6	DNC					
3.0		15.85	14.70	6.114	45	42	2529888
4.0		17.60	17.39	6.073	50	49	2529888
5.0		22.78	22.56	6.076	62	61	2529888
6.0		24.48	24.24	6.075	66	65	2529888
10.0		39.80	39.41	6.080	102	100	2529888

Table D.1: Gradient solver parameters for the matrix sensitivity approach.

σ	BFILU(n)	GMRES time (seconds)		BFILU time (seconds)	Iterations		n_{nz} BFILU
2.0	1	DNC					
3.0		50.79	51.79	1.289	173	175	1078864
4.0		61.04	61.03	1.313	178	180	1078864
5.0		59.86	65.55	1.324	177	194	1078864
6.0		67.36	68.52	1.319	200	202	1078864
2.0	2	DNC					
3.0		DNC					
4.0		54.33	54.34	2.045	156	152	1370224
5.0		55.83	52.57	2.054	157	150	1370224
6.0		49.53	42.68	2.015	160	140	1370224
10.0		74.67	62.72	2.055	206	170	1370224
2.0	3	DNC					
3.0		24.32	24.25	2.765	68	67	1664096
4.0		29.35	29.24	2.735	87	86	1664096
5.0		DNC					
6.0		36.27	36.15	2.765	91	91	1664096
10.0		69.07	67.32	2.790	174	171	1664096
2.0	4	DNC					
3.0		18.93	18.38	3.742	58	56	1953936
4.0		DNC					
5.0		DNC					
6.0		29.19	29.41	3.741	82	82	1953936
10.0		66.97	68.16	3.813	161	163	1953936
2.0	5	DNC					
3.0		22.33	20.96	4.868	55	52	2244192
4.0		21.87	21.28	4.776	62	60	2244192
5.0		40.56	39.98	4.872	90	88	2244192
6.0		29.06	28.79	4.780	78	77	2244192
10.0		67.53	71.92	4.858	155	162	2244192
2.0	6	DNC					
3.0		23.38	21.77	6.165	54	51	2529888
4.0		21.93	21.30	6.091	59	57	2529888
5.0		31.81	31.35	6.186	71	69	2529888
6.0		29.88	29.51	6.089	76	75	2529888
10.0		68.24	74.30	6.200	148	159	2529888

Table D.2: Gradient solver parameters for the matrix-free sensitivity approach.