Aerodynamic Optimization Using High-Order Finite-Volume CFD Simulations

by

Mohammad Baher Azab

B.Sc., Aerospace Engineering Cairo University (Egypt), 1999
M.Sc., Aerospace Engineering Cairo University (Egypt), 2002

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

in

The Faculty of Graduate Studies

(Mechanical Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

October 2011

Abstract

The growth of computer power and storage capacity allowed engineers to tackle engineering design as an optimization problem. For transport aircraft, drag minimization is critical to increase range and reduce operating costs. Lift and geometric constraints are added to the optimization problem to meet payload and rigidity constraints of the aircraft. Higher order methods in CFD simulations have proved to be a valuable tool and are expected to replace current second order CFD methods in the near future; therefore, exploring the use of higher order CFD methods in aerodynamic optimization is of great research interest and is one goal of this thesis.

Gradient-based optimization techniques are well known for fast convergence, but they are only local minimizers; therefore their results depend on the starting point in the design space. The gradient-independent optimization techniques can find the global minimum of an objective function but require vast computational effort; therefore, for global optimization with reasonable computational cost, a hybrid optimization strategy is needed.

A new least-squares based geometry parametrization is used to describe airfoil shapes and a semi-torsional spring analogy mesh morphing tool updates the grid everywhere when the airfoil geometry changes during shape optimization.

For the gradient based optimization scheme, both second and fourth order simulations have been used to compute the objective function; the adjoint approach, well known for its low computational cost, has been used for gradient computation and matches well with finite difference gradient. The gradient based optimizer have been tested for subsonic and transonic inverse design problems and for drag minimization without and with lift constraint to validate the developed optimizer. The optimization scheme used is Sequential Quadratic Programming (SQP) with the BFGS approximation of the Hessian matrix. A mesh refinement study is presented for an aerodynamically constrained drag minimization problem to show how second

and fourth order optimal results behave with mesh refinement.

A hybrid particle swarm / BFGS scheme has been developed for use as a global optimizer. It has been tested on a drag minimization problem with a lift constraint; the hybrid scheme obtained a shock free profiles, while gradient-based optimization could not in general.

Preface

The research ideas and methods explored in the three co-authored manuscripts of this thesis are the fruits of a close working relationship between Dr Carl Ollivier-Gooch and Mohammad Azab. The implementation of the methods, the data analysis, and the manuscript preparation were done by Mohammad Azab with invaluable guidance from Carl Ollivier-Gooch throughout the process.

Table of Contents

List of Tables

List of Figures

List of Algorithms

Nomenclature

α	Angle of attack
$\triangle Q$	Change in conservative flow properties
$\triangle t$	Time step
δ_{norm}	Normalized swarm radius
$\frac{\partial R}{\partial Q}$	jacobian matrix
$\frac{\partial U}{\partial M}$	Dependency of premitive flow variables solution on mesh.
$\frac{dC_{L,D,M}}{dD}$	Gradient of lift, drag. and moment coefficient
$\frac{dF}{dD}$	Objective function gradient
$\frac{dR}{dD}$	Residual dependency of design variable
γ	Specific heats ratio
$\hat{n}$	Face unit normal
Ω_i	Control volume i
$\overline{U}_k$	Control volume avaraged primitive flow properties
$\overrightarrow{F}$	Flux vector across control volume boundary
$\overrightarrow{g}$	Swarm best known position in the design space
$\overrightarrow{p_i}$	Particle i best known position in the design space
$\overrightarrow{r_b}$	Position vector of point b

$\overrightarrow{V_i}$ Particle i velocity

$\overrightarrow{x_i}(k)$ Particle position in iteration k

$\overrightarrow{x_i}(k+1)$ Particle position in iteration k+1

$\partial M/\partial D$ Mesh sensitivity

$\partial R/\partial D$ Residual senstivity with respect to design variables

$\partial \overrightarrow{r_b}/\partial D_i$ Dependency of boundary point position $\overrightarrow{r_b}$vector on design variable D

$\Psi_{L,D,M}$ Adjoint vector of of lift, drag. and moment coefficient

ρ Density

$\tilde{\rho}$ Roe avaraged density

$\tilde{A}$ Roe averaged flux jacobian matrix

$\tilde{a}$ Roe avaraged speed of sound

$\tilde{h}$ Roe avaraged enthalpy

$\tilde{u}$ Roe avaraged x-velocity

$\tilde{v}$ Roe avaraged y-velocity

$\vec{F}(Q_R), \vec{F}(Q_L)$ Right and left face fluxes

C_D Drag coefficient

C_L Lift coefficient

C_M Moment coefficient

C_p, C_v Specific heats at constant pressure and volume

D_i Desigen variable i

e_t Specific total energy

K_{ib} Stiffness matrix intries relates interior nodes with boundary points

K_{ii} Stiffness matrix intries relates interior points

P_{Rec} Flow property reconstruction polynomial

P_s Surface pressure

Q Conservative flow variables vector

Q_R, Q_L conserved flow properties at right and left of the cell face

U_{bg} Boundary Gauss point flow premitive properties

U_{f_i} Reconstructed primitive flow properties at face Gaussian integration point i

U_i, U_b Interior and boundary mesh point displacements respectively

U_n Velocity in the direction normal to the control volume boundary

w_s Gaussian integration weight

y_c Geometry control point variable y ordinate (design veriable)

a Speed of sound

E Energy per unit volume

e Specific internal energy

F Aerodynamic objective function

GMRES Generalized minimal residual

h Specific enthalpy

M Mach number

P Pressure

R Residual

RMS Root mean square

T Temperature

Acknowledgments

I would like to express my deepest gratitude to my supervisor Dr Carl Ollivier-Gooch. You are such a great research supervisor and gentleman. I would like also to acknowledge my supervision committee, Dr Michael Friedlander, Dr Philip Loewen, and Dr James Olson for their valuable remarks that helped me to justify my research strategy.

I would like also to thank my fellows in the research and development department of the Egyptian Air Force especially Col. Dr Eng. Mohamed Ibrahim Mustafa and Major General Dr Abdalla El-Ramsisy for their great support during my research in Egypt which has had a great impact on my PhD research progress.

Great thanks goes also to my dear professor Dr Chen Grief for being there when I needed his advice.

Finally, I would like to thank my wife and the light of my life, Deena, for her support and bringing happiness to my life.

Mohammad Baher Azab

Dedication

To my home country, the land of Pharaohs

Egypt

To the land of hockey

Canada

To the souls of my parents

Baher and Laila

Chapter 1

INTRODUCTION

Drag reduction of transport aircraft is of great importance because it reduces aircraft fuel consumption and thereby reduces operating costs and environmental impact in the form of pollution and global warming. Drag, lift and other aerodynamic forces can be predicted using CFD simulations, which have become an essential tool for aerodynamic analysis and design. CFD simulations carried out using unstructured grids give accurate aerodynamic force predictions, and unstructured grids have the advantage of easily representing any complex shape. As most transport aircraft travel at transonic speed, it is of great importance to reduce the wave drag by weakening or entirely eliminating shock waves on the wing. Aerospace engineers use CFD simulations with numerical optimization techniques for aerodynamic design; the optimization problem is to minimize an aerodynamic objective (often drag) by changing geometric design variables, given an initial aerodynamic shape and subject to some geometric and aerodynamic constraints. This process requires accurate assessment of the aerodynamic characteristics of a given geometry. The flow model used in this research is Euler's flow model; although this model neglects viscous flow effects, reducing the drag using this inviscid flow model will ultimately reduce the drag on real configurations as the inviscid drag is about 40% of the total drag [47]. A higher order finite volume CFD solver developed by C. Ollivier-Gooch and co-workers [68, 61, 56] is used in this research.

Optimization schemes can be divided into two main categories: gradient-based and non-gradient-based schemes. Gradient-based schemes require computing the objective function value and its gradient with respect to the design variables, while the non-gradient-based techniques require only computing the objective function value. Gradient-based schemes are fast to converge to a local optimal point in the design space compared to the non-gradient-based methods, but the optimal solution found by gradient-based optimization depends on the starting point [33]. Non-gradient-

based schemes can find global minimum solution regardless of the starting point, but with larger computational cost compared to gradient-based scheme.

1.1 Finite Volume Flow Solver

The two dimensional integral form of Euler's equation can be written for a control volume Ω_i as

$$\frac{\partial}{\partial t}\iint_{\Omega_i} Q dV + \oint_{d\Omega_i} \overrightarrow{F} \cdot \hat{n} dl = 0 \tag{1.1}$$

where $\hat{n} = n_x i + n_y j$ is the outward pointing normal to the control volume faces; Q and $\overrightarrow{F}$ are respectively the conserved variable vector and the flux across the boundary of control volume Ω_i boundaries. These can be expressed as

$$Q = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ E \end{bmatrix}, \quad \overrightarrow{F} \cdot \hat{n} = \begin{bmatrix} \rho U_n \\ \rho U_n u + n_x p \\ \rho U_n v + n_y p \\ (E+p)\, U_n \end{bmatrix}$$

$$p = (\gamma - 1)\left[E - \rho \frac{\left(u^2 + v^2\right)}{2}\right]$$

$$U_n = n_x u + n_y v$$

where is the fluid density, u and v are the Cartesian velocity components, p is the pressure, and E is the energy per unit volume. U_n is the velocity in the direction normal to the control volume boundary. Other thermodynamic relations like the specific heats at constant pressure (C_p) and constant volume (C_v), the ratio of specific heats γ can be expressed in terms of the gas constant for air, R:

$$\gamma = \frac{C_p}{C_\nu}, \qquad C_p = \frac{\gamma R}{\gamma - 1}, \qquad C_\nu = \frac{R}{\gamma - 1}$$

For a thermally and calorically perfect gas, thermodynamic properties can be related by

$$
\begin{aligned}
P &= \rho R T \\
e &= C_v T \\
e_t &= e + \frac{1}{2}\left(u^2 + v^2\right) \\
E &= \rho e_t \\
h &= e + \frac{P}{\rho} \\
a &= \sqrt{\gamma R T}
\end{aligned}
$$

where T is the temperature, e is the specific internal energy, e_t is the specific total energy, h is the specific enthalpy, and a is the speed of sound.

For discrete solution of the Euler equations, flow properties are normalized by some reference values in order to reduce the round-off errors in the discretized linear system; for external aerodynamics, the normalized flow properties are as follows

$$
\begin{aligned}
\bar{\rho} &= \frac{\rho}{\rho_\infty}, \qquad \bar{a} = \frac{a}{a_\infty}, \qquad \bar{P} = \frac{P}{\rho_\infty a_\infty^2} \\
\bar{u} &= \frac{u}{a_\infty}, \qquad \bar{v} = \frac{v}{a_\infty}, \qquad \bar{E} = \frac{\bar{P}}{\gamma - 1} + \frac{1}{2}\bar{\rho}\left(\bar{u}^2 + \bar{v}^2\right) \\
\bar{x} &= \frac{x}{L}, \qquad \bar{y} = \frac{y}{L}, \qquad \bar{t} = \frac{t a_\infty}{L}
\end{aligned}
$$

With this normalization, the normalized Euler equations are identical to their dimensional form with the addition of $\bar{[\cdot]}$ to every variable. From this point on, the normalized flow properties are used and therefore $\bar{[\cdot]}$ will be dropped.

The flux is evaluated using Roe flux difference splitting technique [74] and evaluated at each cell face κ using the following formula

$$
\vec{F} = \frac{1}{2}\left[\vec{F}\left(Q_R\right) + \vec{F}\left(Q_L\right) - \left|\tilde{A}\right|\left(Q_R - Q_L\right)\right]_\kappa \tag{1.2}
$$

where Q_R, Q_L are the conserved flow properties at right and left of the cell face κ and the Roe averaged matrix $\tilde{A}$ is the flux Jacobian $\partial\vec{F}/\partial Q$ at Roe-averaged quantities

as follows

$$\begin{aligned}
\tilde{\rho} &= \sqrt{\rho_L \rho_R} \\
\tilde{u} &= \frac{u_L + u_R\sqrt{\frac{\rho_R}{\rho_L}}}{1+\sqrt{\frac{\rho_R}{\rho_L}}}, \qquad \tilde{v} = \frac{v_L + v_R\sqrt{\frac{\rho_R}{\rho_L}}}{1+\sqrt{\frac{\rho_R}{\rho_L}}} \\
\tilde{h} &= \frac{h_L + h_R\sqrt{\frac{\rho_R}{\rho_L}}}{1+\sqrt{\frac{\rho_R}{\rho_L}}}, \qquad \tilde{a}^2 = (\gamma-1)\left(\tilde{h} - \frac{\left(\tilde{u}^2+\tilde{v}^2\right)}{2}\right)
\end{aligned} \tag{1.3}$$

The Roe-averaged Jacobian matrix $\tilde{A}$ has four eigenvalues. Three of these are distinct, but the eigensystem is complete. The Roe dissipation matrix can be written in terms the eigenvalues and eigenvectors of the Jacobian as

$$\left|\tilde{A}\right| = \tilde{X}\begin{bmatrix} \left|\tilde{U}_n\right| & 0 & 0 & 0 \\ 0 & \left|\tilde{U}_n\right| & 0 & 0 \\ 0 & 0 & \left|\tilde{U}_n+\tilde{a}\right| & 0 \\ 0 & 0 & 0 & \left|\tilde{U}_n-\tilde{a}\right| \end{bmatrix}\tilde{X}^{-1} \tag{1.4}$$

where $\tilde{X}$ is the eigenvector matrix evaluated using Roe-averaged flow quantities; the term $\left|\tilde{A}\right|(Q_R - Q_L)$ can be written according to Frink as follows [27, 26]

$$\begin{aligned}
\left|\tilde{A}\right|(Q_R - Q_L) &= \left|\tilde{A}\right| \triangle Q \\
&= \left|\triangle \tilde{F}_1\right| + \left|\triangle \tilde{F}_4\right| + \left|\triangle \tilde{F}_5\right|
\end{aligned} \tag{1.5}$$

where

$$\left|\triangle\tilde{F}_1\right| = \left|\tilde{U}_n\right| \left\{ \left(\triangle\rho - \frac{\triangle P}{\tilde{a}^2}\right) \begin{bmatrix} 1 \\ \tilde{u} \\ \tilde{v} \\ \frac{\tilde{u}^2+\tilde{v}^2}{2} \end{bmatrix} + \tilde{\rho} \begin{bmatrix} 0 \\ \triangle u - n_x \triangle U \\ \triangle v - n_y \triangle U \\ \tilde{u}\triangle u + \tilde{v}\triangle v - \tilde{U}_n \triangle U \end{bmatrix} \right\}$$

$$\left|\triangle\tilde{F}_{4,5}\right| = \left|\tilde{U}_n \pm \tilde{a}\right| \left(\frac{\triangle P \pm \tilde{\rho}\tilde{a}\triangle U}{2\tilde{a}^2}\right) \begin{bmatrix} 1 \\ \tilde{u} \pm n_x\tilde{a} \\ \tilde{v} \pm n_y\tilde{a} \\ \tilde{h_o} \pm \tilde{U}_n\tilde{a} \end{bmatrix}$$

$$\begin{aligned} \triangle\rho &= \rho_R - \rho_L, \qquad \triangle u = u_R - u_L, \qquad \triangle v = v_R - v_L, \\ \triangle P &= P_R - P_L, \qquad \triangle U = n_x\triangle u + n_y\triangle v \end{aligned}$$

Higher-order accuracy is obtained by least-squares reconstruction of the non-conserved variables $U = \begin{bmatrix} \rho & u & v & p \end{bmatrix}^T$ and Gauss quadrature in flux integration [68, 61]. After integrating fluxes around each control volume in the mesh, an implicit time discretization leads to a sparse system of linear equations, which for the simplest case of a global time step can be written as,

$$\left[\frac{I}{\triangle t} + \frac{\partial R}{\partial Q}\right]^n \triangle Q^n = R^n \tag{1.6}$$

where $\triangle Q = \begin{bmatrix} \triangle\rho & \triangle\rho u & \triangle\rho v & \triangle E \end{bmatrix}^T$, $\frac{\partial R}{\partial Q}$ is the global Jacobian matrix, and R is the residual. The steady state solution is obtained iteratively when $\triangle Q \to 0$. In practice, we use a quasi-Newton generalization of Eq. 1.6 that includes residual-based local time stepping [57] and solve the system using GMRES [75].

1.2 Numerical Aerodynamic Optimization

Aerodynamic design used to rely on CFD simulations in conjunction with experimental testing and engineering intuition of the designer. With the growth of high speed computers, integrating numerical optimization schemes with CFD simulations has

become possible and is now used for aerodynamic design and optimization. Gradient-based optimization techniques are widely used because they reach an optimized shape after a reasonable execution time; however, the final optimal shape is the local minimum located downhill from the optimization starting point. Non-gradient-based methods like genetic algorithm (GA) or particle swarm (PS) are slower to find an optimum but can find the global minimum regardless of the starting point; their drawback is the large number of iterations required to reach this minimum compared to gradient-based schemes.

1.2.1 Gradient-based aerodynamic optimization

Gradient-based optimization depends on evaluating the gradient of the objective function with respect to the design variables and using the gradient in a linear model (steepest descent) or a quadratic model (Newton or quasi-Newton model) to find a search direction; this search direction is the direction in which the design variables should change their values to minimize the objective function [66]. Gradient-based techniques have been widely used in aerodynamic optimization due their fast convergence to an optimal solution. The obtained optimal shape is biased by the optimization starting point (initial aerodynamic shape), and there is no guarantee that gradient-based methods can find the global best optimal shape in the design space [2]. Hicks and Henne where among the first to apply gradient-based optimization techniques in aerodynamic design in the late 1970's [35]. Since then many researchers have investigated the use of gradient-based optimization techniques like steepest descent and quadratic programming in aerodynamic optimization [17, 31, 28]. The most expensive part of gradient-based optimization is the gradient calculation. Hicks and Henne used finite difference rules to calculate the objective function gradient. This means that two CFD simulations were required for each design variable to compute the gradient (using a central finite difference rule), which is computationally expensive. The same strategy was applied by Consentino and Holst to optimize transonic wings [17]. The use of the adjoint method, which was originally applied to aerodynamics problems by Jameson, to compute the gradient reduced the computational cost of gradient calculation to the cost of one flow simulation regardless of the number of design variables [41, 43, 60, 38, 39, 42, 11, 62].

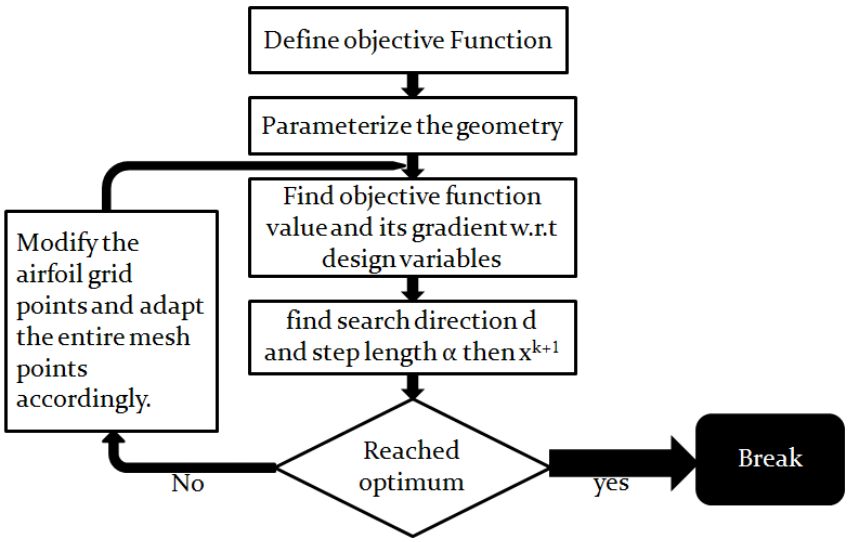


Figure 1.1: Airfoil aerodynamic optimization cycle using gradient-based optimization

Numerous researchers have applied gradient-based optimization using the adjoint approach in aerodynamic optimization since the early 1990's. Reuther and his co-workers applied the adjoint approach for aerodynamic optimization of aircraft configuration using Euler's flow model [73]. Jameson developed an adjoint formulation for the Navier-Stokes equations and applied it to transonic wing optimization [43]; in this work, Jameson suggested that an optimal pressure distribution first be obtained using Euler's flow model then used as a target pressure for an inverse design optimization problem using a Navier-Stokes adjoint optimizer to reduce the overall computational cost. However, Jameson treated the eddy viscosity as constant which was later shown to be a bad assumption. Anderson and Bonhaus examined the effect of the strength of coupling of the turbulence model to the flow equations. They compared the adjoint gradient of the flow equations, with the eddy viscosity frozen, with the finite difference gradient of the combined flow and turbulence equations, and found that freezing the eddy viscosity can lead to significant error in the computed gradient. Therefore, they developed a flow solver that couples the Spalart-Allmaras one-equation turbulence model with the flow equations; their coupled solver used 5×5 blocks for two dimensional flows and 6×6 blocks for three dimensional cases. They found that this coupling of the turbulence equations with the flow equations led

to an accurate adjoint gradient [3]. The same observation was verified by Nielsen and Kleb who extended their adjoint solver to deal with chemically reacting flows [65]. Zymaris et al developed a continuous adjoint optimizer for turbulent flow using the $k-\varepsilon$ turbulence model and applied it to duct optimization; they showed also that the assumption of constant eddy viscosity leads to great inaccuracy in the computed gradient and this leads to a poor search direction [88].

Regarding the optimization technique used, early researchers used the steepest descent scheme, in which the design variables are updated on a search direction exactly opposite to the gradient of the objective function [66, 4]. This scheme has been implemented by Jameson and other researchers [37, 41, 73, 3], but as the steepest descent scheme requires a large number of iterations to converge to minimal point, the sequential quadratic programming (SQP) scheme seems to be an attractive candidate as an optimization technique. The use of SQP requires computing the Hessian of the objective function with respect to the design variables. The exact Hessian is expensive to compute and may not be positive definite; therefore, the Broyden–Fletcher–Goldfarb–Shanno (BFGS) approximate formula is often adopted to approximate the Hessian using objective function gradient history [12, 13, 87]. The BFGS approximation always gives a positive definite approximate Hessian, therefore a real optimization search direction is guaranteed. Dadone et. al used BFGS in aerodynamic optimization for transonic and supersonic wings and compared BFGS and steepest descent; there results showed that the BFGS method is more efficient in finding the optimal solution and is less sensitive to any inaccuracy caused by approximation in gradient computations [19]. BFGS optimization has also been used by Nemec and Zingg in subsonic and transonic turbulent aerodynamic optimization of two dimensional airfoil [63].

1.2.2 Gradient free optimization

Gradient free or gradient independent optimization methods, also known as heuristic optimization methods, are optimization techniques that do not require objective function gradient computation and therefore can be applied to non-differentiable problems. They can be categorized as evolutionary schemes (including genetic algorithms) and random search schemes (like the particle swarm technique).

Genetic algorithm optimization is an evolutionary optimization algorithm that is inspired by Darwin's theory of evolution and natural selection [30]. The design variables are treated as chromosomes and optimization is carried out by crossing and mutating these chromosome to find a better solution that minimizes the objective function. The initial population is randomly generated, the objective function value is computed using CFD for each population member and a fitness value is computed based on that; some members are selected based on their fitness to be the parents of the next generation and are used to generate the chromosomes of the offspring of the next optimization iteration. Transonic wing optimization using a genetic algorithm was explored by Gregg and Misegades [32] and by Gage and Kroo [29] in the late 1980's to minimize drag with lift constraint. Somewhat later, Anderson applied a genetic algorithm in subsonic wing optimization with structural constraints [1]; he added the geometric and aerodynamic constraints to the objective function as penalty terms. Jang and Lee applied a genetic algorithm in subsonic and transonic inviscid airfoil optimization; their objective was to maximize the lift-to-drag ratio of an airfoil, beginning from the NACA 0012[1] [44]. Oyama et al applied a genetic algorithm with a Navier-Stokes solver for transonic wing optimization [70]. They also explored the use of fractal analysis in GA aerodynamic optimization [71].

The particle swarm method (PSOpt) is a stochastic optimization search method developed by Eberhart and Kennedy in 1995, inspired by the social behavior of bird flocks [46, 21]. The general idea of the particle swarm optimization is to randomly generate a swarm of particles in the design space. For each particle a fitness value is calculated based on CFD simulation. Then particles "fly" in the design space according to a simple formula that takes into account that particle's own best fitness position and the swarm's overall best fitness position [22]. PSOpt is known to suffer from premature convergence prior to discovering the true global minimizer; Evers suggests an automatic regrouping PSO (RegPSO) that automatically triggers swarm regrouping when premature convergence is detected. The suggested regrouping strategy aims to liberate particles from sub-optimal solutions and enables finding the global minimum [24]. Although the PSO algorithm has been applied to a wide

[1] The National Advisory Committee for Aeronautics (NACA) airfoil family geometric coordinates can be found at University of Illinois Urbana-Champaign website http://www.ae.illinois.edu/m-selig/ads/coord_database.html. An explanation of the meaning of the digits in the NACA airfoil naming scheme can be found in [36].

range of engineering problems in the literature, very few aerodynamic optimization applications are known. Venter and Sobieszczanski applied the particle swarm optimization technique in the multidisciplinary optimization of a wing; the objective was to maximize the aircraft range by maximizing lift-to-drag ratio and reducing the wing weight subject to geometric constraints [85]. Chandrashekarappa and Duvigneau [15] used particle swarm optimization scheme to aerodynamically optimize wings in supersonic conditions; Duvigneau also applied particle swarm optimization to aerodynamic optimization of wings at transonic speed with free stream Mach number uncertainty [20].

1.3 Contributions of the Thesis

High order CFD methods can compute an accurate value of an aerodynamic objective function at lower computational cost than required when a second order method is used. The first major contribution of this thesis is a study of whether the efficiency of high order methods for CFD analysis translates into improved efficiency for gradient-based aerodynamic optimization.

Gradient based optimization techniques are known to be local minimizers, with results depending on the starting airfoil geometry. The second major contribution of this thesis is the development and study of an optimization scheme that can reach a true global optimum (for inviscid transonic aerodynamics, a shock free airfoil subject to aerodynamic and geometric constraints) after a reasonable number of CFD simulations. This scheme is a hybrid (BFGS + regrouped particle swarm) scheme that takes the advantages of gradient-based and gradient-free optimization schemes.

To achieve these goals, the following components were needed, in addition to the pre-existing high-order accurate flow solver:

An efficient geometry parametrization method. For optimization purposes, the airfoil shape must be represented by a finite number of design variables. Chapter 2 describes the requirements on such a parametrization and presents a new least-squares spline parametrization method developed for this work.

Robust mesh movement. As the airfoil shape changes during optimization, the computational mesh must be updated. Because mesh regeneration will intro-

duce unacceptably large changes in the discretization error, mesh movement is strongly preferred. Chapter 3 describes the semi-torsional spring mesh movement scheme used in this thesis.

Mesh sensitivity calculation. Calculating the gradient of the objective function accurately and efficiently requires information about the movement of the mesh with changes in design variables. Section 3.3 describes this process, which combines aspects of the geometric parametrization and mesh movement schemes.

Objective function gradient calculation. Efficient gradient calculation requires the solution of the adjoint to the governing flow equations. Chapter 4 describes how this is done. A key innovation of the thesis is efficient solution of the discrete adjoint problem for a high-order accurate flow solution scheme. In addition to showing the formulation of the gradient calculation, this chapter compares finite difference and adjoint gradients for subsonic and transonic flows for second and fourth order finite volume schemes. For transonic flow, results are presented both with and without limiting of the computational solution to prevent overshoots.

Optimization drivers. The gradient-based optimizer using in this thesis is the BFGS-based quasi-Newton solver in Matlab's optimization toolbox. Gradient based optimization test cases for inverse design problems and for drag minimization with and without a lift constraint is shown in Chapter 6. Also, the impact of mesh refinement on the second and fourth order optimal results is studied.

The regrouped particle swarm optimization code was written by the author. This scheme and its hybridization with the BFGS scheme is presented in Chapter 7. Examples are given to show the effectiveness of the hybrid scheme in finding global optima when the gradient-based scheme is unable to.

Chapter 2

GEOMETRY PARAMETRIZATION

The geometry of engineered objects is defined mathematically in computer-aided design (CAD) software, then exported as a group of points or polygons which approximates the original geometry. This discrete data provides input to mesh generation software that creates a discrete representation of the computational domain (a *mesh*), which in turn is used as input for CFD analysis. Aerodynamic design and optimization modifies the aerodynamic shape by changing a set of geometric design variables. An obvious choice is to use all the surface grid points of a wing, but this approach causes two problems. First, it makes the design space very large and this may lead to a highly expensive optimization. Second, it may lead to a non-smooth geometry due to the displacement independence of a surface mesh point; this problem can be solved by the use of a smoothing function, as described by Jameson [41]. To avoid these problems, most optimization approaches rely on some form of geometric parametrization. Geometric parametrization techniques can be classified as analytical; piece-wise spline fitting; CAD based; and free form deformation (FFD) approaches. In Section 2.1, a review of various geometry parametrization techniques is presented. Section 2.2 describes in detail the geometry parametrization technique used in this research, a novel piece-wise least-squares fitting technique [5, 6].

2.1 Survey of Geometric Parametrization Techniques

In this review, various techniques applied to aerodynamic optimization are presented; the basics of these techniques and their application limitations are discussed, including reference to the original papers describing them in more detail .

2.1.1 Analytical parametrization

The analytical approach was first applied by Hicks and Henne for airfoil optimization [35]. They suggest that weighted sinusoidal displacement shape functions be added to the base geometry to modify the airfoil shape; the weights are the optimization design variables. The sinusoidal displacement shape function is expressed as

$$h(x) = \left(\sin\left(\pi x^{\frac{\ln(0.5)}{\ln(b)}}\right)\right)^{a} \qquad 0 < x < 1 \tag{2.1}$$

where a and b are constants to control the peak location and the width of the sinusoidal displacement shape function. $a = 4$ is recommended for most cases, while b must be between 0 and 1 [45].

The Class-Shape function-Transformation method (CST) is another analytical geometry parametrization technique presented by Kulfan [49, 48]. CST parametrizes the airfoil geometry using the following formula

$$y = x^{0.5}(1-x)S(x) + x\triangle Z_{TE} \tag{2.2}$$

where $S(x)$ is the shape function and $\triangle Z_{TE}$ is the finite thickness of the trailing edge. Kulfan and Bussoletti recommended using a weighted Bernstein binomial of order n as a shape function

$$S(x) = \sum_{i=0}^{n} b_i \frac{n!}{i!\,(n-i)!} x^i (1-x)^{n-i} \tag{2.3}$$

where the weights b_i are used as the design variables. Although CST gives non-wavy profiles, it is not capable of representing complex geometries. Mousavi and Nadiragah compared the impact of using different geometry parametrization techniques on the optimal wing shape; using CST parametrization gave drag coefficients higher by about 15% than the optimal geometry from a B-spline parametrization [58] for a three-dimensional lift-constrained transonic drag minimization problem.

2.1.2 Piece-wise spline parametrization

Bezier curves can be used for airfoil shape parametrization. Obayashi used Bezier curve for aerodynamic optimization using genetic algorithm [67]; he noticed that the Bezier curve representation fails to represent geometries that gives rooftop pressure distributions[2] because Bezier curves are always convex. B-splines, a generalization of Bezier curves, were found to be more suitable. A cubic B-spline representation is a very good geometry parametrization technique. Better control of the cubic spline representation can be obtained by increasing the number of splines that represent the airfoil; Li et al. optimized the NACA 0012 at single point and multi-point operating conditions with a lift constraint using spline representation as geometry parametrization technique [50]. CAD systems typically use non-uniform rational B-spline (NURBS) representation for geometry modeling, allowing them to represent any complex geometry; a detailed discussion of NURBS can be found in *The NURBS Book* [72]. Mengistu and Ghaly applied successfully a NURBS parametrization scheme to turbomachine blade aerodynamic optimization using a gradient-free method [53]. While piece-wise spline parametrization is well-suited for two-dimensional shapes and simple three-dimensional geometries, complex shapes require a large number of control points, reducing the effectiveness of gradient-based optimization techniques [76].

2.1.3 CAD parametrization

Computer aided design packages have evolved to implement NURBS for geometry representation due the excellent properties of NURBS. Linking the CAD and grid generation software can be done using an API that allows access to the CAD system's internal interface [81, 8]. However, imposing geometric constraints is still an obstacle. Mesh sensitivity calculation is another obstacle for gradient-based optimization techniques based of CAD parametrization: analytical mesh sensitivity with respect to geometry design variables — the NURBS knots — can in principle be computed with the use of automatic differentiation of the CAD software but this is not possible without CAD source code and is unlikely to be practical even then, given the size of

[2] Distributions for which pressure remains almost unchanged over a significant chordwise distance.

the CAD code base. This derivative can also be computed using finite differences, but the risk of poor accuracy still exists, and computational costs are higher [76, 77].

2.1.4 Free form deformation (FFD)

Computer graphics requires large graphical deformations such as stretching, twisting and other surface morphing operations; soft object animation (SOA) algorithms were developed to help with the geometry morphing required in graphics animation [86]. In SOA algorithms, the object surface is treated as a piece of rubber and the desired deformation can be obtained by applying loads on it, so geometry morphing can be obtained without a change in surface topology. The surface itself can be parametrized using Bezier or B-splines or even NURBS splines. A related approach, the free form deformation algorithm (FFD) treats the geometry as a void in a box-shaped piece of rubber. Deformation can be controlled by moving control points placed on the outer surface of the box; the interior of the rubber box with its void is parametrized using a tensor product of three spline representations (one in each coordinate direction). Sederberg and Parry developed an algorithm that uses the FFD concept with Bezier tri-variate volume representation [78]. A disadvantage of the FFD method is that it requires large numbers of control points to obtain local deformation in the deformed geometry. However, Borrel and Rappoport presented a method to allow local shape deformation via FFD by introducing a set of control points with constrained local B-splines that can be used to obtain deformation in a radius of influence determined by the designer [10].

2.1.5 Multidisciplinary aerodynamic/structural shape optimization using deformations (MASSOUD)

The MASSOUD method is an analogy to analytical methods that tries to parametrize the deformations in the geometry rather than the geometry itself. It also utilizes SOA algorithms and allows strong local deformation control. The MASSOUD parametrization requires few design variables because it parametrizes the deformation. Samareh has applied the MASSOUD method to parametrize a simple wing, a wing body blend, and a complex aircraft configuration with success [76]. Nielsen and Anderson successfully adopted the MASSOUD parametrization scheme for aerody-

namic optimization of turbulent flow using unstructured grids [64].

2.2 New Least Squares Parametrization Technique

A new least-squares based surface parametrization method is presented in this section; the airfoil surface is parametrized using piece-wise polynomials whose coefficients are found by solving a least squares problem [6]. This section also describes how to implement a thickness constraint with the new parametrization and presents some validation test cases.

2.2.1 Airfoil geometry parametrization

In the proposed technique, the geometry is parametrized using piece-wise polynomials found by a least-squares fit. The parametrization polynomials are controlled by a set of control points and satisfy $\mathcal{C}^2$ continuity at their meeting points. The airfoil upper and lower surfaces are represented using two least square splines for each surface as shown in Fig. 2.1. The polynomials[3] used are

$$\begin{aligned} P_1(x) &= a_0\sqrt{x} + a_1 x + a_2 x^2 + a_3 x^3 \qquad 0 < x < L_1 \\ P_2(x) &= b_0 + b_1 x + b_2 x^2 + b_3 x^3 \qquad L_1 < x < L \end{aligned} \tag{2.4}$$

where x is the normalized chord-wise position, L_1 is chord-wise position that separates the polynomial regions, and $L = 1$. These polynomials are suitable for an airfoil with a rounded leading edge due to the existence of the $\sqrt{x}$ term in $P_1(x)$, which gives an infinite slope at $x = 0$. The x and y coordinates of the design control points shown in Fig. 2.1 are used to find the values of the polynomial coefficients. These polynomials must satisfy continuity of value, slope, and curvature at their meeting point $x = L_1$. These conditions can be written as

- Value continuity:

$$P_1(L_1) - P_2(L_1) = 0 \tag{2.5}$$

[3] Technically, P_1 is not a polynomial because of the presence of the $\sqrt{x}$ term used to give infinite slope and finite radius of curvature at $x = 0$. However, the label is convenient and not overly confusing.

- Slope continuity:

$$P_1^{'}(L_1) - P_2^{'}(L_1) = 0 \tag{2.6}$$

- Curvature continuity:

$$P_1^{''}(L_1) - P_2^{''}(L_1) = 0 \tag{2.7}$$

An additional constraint should be added on $P_2(L)$ to assure zero thickness at the trailing edge. The above "hard" constraints should be strictly satisfied by the geometry parametrization polynomials; they can be written in matrix form as

$$BP = 0$$

where

$$B = \begin{bmatrix} \sqrt{L_1} & L_1 & L_1^2 & L_1^3 & -1 & -L_1 & -L_1^2 & -L_1^3 \\ \frac{1}{2\sqrt{L_1}} & 1 & 2L_1 & 3L_1^2 & 0 & -1 & -2L_1 & -3L_1^2 \\ \frac{-1}{4\sqrt{L_1^3}} & 0 & 2 & 6L_1 & 0 & 0 & -2 & -6L_1 \\ 0 & 0 & 0 & 0 & 1 & L & L^2 & L^3 \end{bmatrix}, \qquad P = \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ b_0 \\ b_1 \\ b_2 \\ b_3 \end{Bmatrix}$$

The free parameters are chosen to best approximate the y coordinates of the airfoil shape control points; the x coordinates of the control points are fixed. The resulting least square system with constraints applied can be expressed as

$$\min \|AP - c\|^2 \quad \text{subject to } BP = 0 \tag{2.8}$$

where A contains powers of the x coordinates at the design points so that AP gives the y coordinates of the parametrized shape at the design points and c contain the actual y coordinates of the design points. I have used a set of control points that controls the airfoil shape functions instead of using the coefficients of the shape functions to ease constraints and boundary definitions. To give an example of how this least-squares system is constructed, consider the parametrized airfoil surface

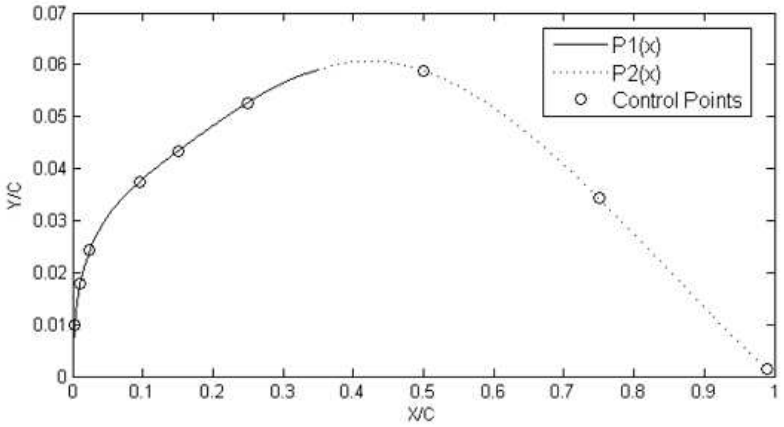


Figure 2.1: Least square surface presentation of RAE 2822 airfoil using two polynomials $P_1(x)$ & $P_2(x)$ fitted using nine control points.

shown in Fig. 2.1. Six control points lies in the region of the polynomial $P_1(x)$, while three points lies in the $P_2(x)$ region. The corresponding least-squares system is

$$\begin{bmatrix} \sqrt{x_1} & x_1 & x_1^2 & x_1^3 & 0 & 0 & 0 & 0 \\ \sqrt{x_2} & x_2 & x_2^2 & x_2^3 & 0 & 0 & 0 & 0 \\ \sqrt{x_3} & x_3 & x_3^2 & x_3^3 & 0 & 0 & 0 & 0 \\ \sqrt{x_4} & x_4 & x_4^2 & x_4^3 & 0 & 0 & 0 & 0 \\ \sqrt{x_5} & x_5 & x_5^2 & x_5^3 & 0 & 0 & 0 & 0 \\ \sqrt{x_6} & x_6 & x_6^2 & x_6^3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & x_7 & x_7^2 & x_7^3 \\ 0 & 0 & 0 & 0 & 1 & x_8 & x_8^2 & x_8^3 \\ 0 & 0 & 0 & 0 & 1 & x_9 & x_9^2 & x_9^3 \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ b_0 \\ b_1 \\ b_2 \\ b_3 \end{Bmatrix} = \begin{Bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \\ y_9 \end{Bmatrix} \tag{2.9}$$

Because the constraint equation, Eq. 2.8, has a zero right hand side, the solution vector P must lie in the null space of the constraint equations, i.e. it should be a linear combination of the null space basis of the constraint equations. The matrix B is full row rank and to find the null space basis of it, QR factorization can be used:

$$B^T = QR$$

$$Q = \left[\begin{array}{cccc|cccc} \overrightarrow{q_1} & \overrightarrow{q_2} & \overrightarrow{q_3} & \overrightarrow{q_4} & \overrightarrow{q_5} & \overrightarrow{q_6} & \overrightarrow{q_7} & \overrightarrow{q_8} \end{array} \right] = \left[\begin{array}{c|c} Q_1 & Q_2 \end{array} \right]$$

The vectors of the $[Q_2]$ matrix are unit vectors forming a basis of the null space of matrix B. The solution vector P must be a linear combination of the vectors of the matrix Q_2

$$P = z_1 \cdot \overrightarrow{q_5} + z_2 \cdot \overrightarrow{q_6} + z_3 \cdot \overrightarrow{q_7} + z_4 \cdot \overrightarrow{q_8} = Q_2 z \tag{2.10}$$

Substituting Eq. 2.10 into Eq. 2.8

$$AP = AQ_2 z = c \tag{2.11}$$

Solving the least-squares system by singular value decomposition for numerical stability,

$$z = [AQ_2]^{\dagger} c \tag{2.12}$$

where the rectangular matrix $[AQ_2]^{\dagger}$ is the pseudo-inverse of $[AQ_2]$. Finally,

$$P = Q_2 [AQ_2]^{\dagger} c \tag{2.13}$$

Equation 2.13 gives the relationship between the polynomial coefficients P and the y locations of the control points c.

The sensitivity of the polynomial coefficients with respect to the y location of the i^{th} control point, which is needed to calculate the mesh sensitivity $\partial M / \partial D$, is the i^{th} column of the matrix $Q_2 [AQ_2]^{\dagger}$.The dependency of an airfoil surface point $\overrightarrow{r_b} = \left[\begin{array}{c} x_a \\ y_a \end{array} \right]$ on a design variable D_i can be found from

$$\frac{\partial \overrightarrow{r_b}}{\partial D_i} = \left[\begin{array}{c} 0 \\ \frac{\partial a_0}{\partial D_i}\sqrt{x_a} + \frac{\partial a_1}{\partial D_i} x_a + \frac{\partial a_2}{\partial D_i} x_a^2 + \frac{\partial a_3}{\partial D_i} x_a^3 \end{array} \right] \qquad 0 < x_a < L_1 \tag{2.14}$$

$$\frac{\partial \overrightarrow{r_b}}{\partial D_i} = \left[\begin{array}{c} 0 \\ \frac{\partial b_0}{\partial D_i} + \frac{\partial b_1}{\partial D_i} x_a + \frac{\partial b_2}{\partial D_i} x_a^2 + \frac{\partial b_3}{\partial D_i} x_a^3 \end{array} \right] \qquad L_1 < x_a < L$$

where $\left[\begin{array}{cccccccc} \frac{\partial a_0}{\partial D_i} & \frac{\partial a_1}{\partial D_i} & \frac{\partial a_2}{\partial D_i} & \frac{\partial a_3}{\partial D_i} & \frac{\partial b_0}{\partial D_i} & \frac{\partial b_1}{\partial D_i} & \frac{\partial b_2}{\partial D_i} & \frac{\partial b_3}{\partial D_i} \end{array} \right]^T$ is the i^{th} vector of the constrained pseudo inverse matrix $Q_2 \left[AQ_2\right]^\dagger$. This procedure can be extended to parametrize airfoil surface using any number of piece-wise polynomials, with an accompanying increase in system size. Leading edge radius and trailing edge thickness constraints can be added to the $\mathcal{C}^2$ continuity requirements forming the constraint system $BP = d$, and the control point coordinates are used to construct the system $AP = \{y_c\}$.

In practice, all airfoil surface points are used in the least squares system to find the polynomials coefficients. After finding the polynomials coefficients, the designer can select a set of control points which lies on the parametrized surface; the least number of control points is two per polynomial. The less control points used, the less control of airfoil geometry. I have selected nine control points at specific chord-wise x-stations based on my engineering sense; it turns out that my selected set of control points were able to produce shock free optimal profiles as will be shown in the next chapters, however, selection of the control points x-stations can be made by formulating a simple minimization problem. In this problem, a set of airfoil geometric data gathered and the objective is to minimize the RMS error between the original airfoil surfaces and the parametrized airfoil surfaces, where the design variables are the control points x-stations.

2.2.2 Thickness constraint

Some aircraft fuel tanks are placed inside the wing, and the main landing gear of some aircraft are stored in the wings after being retracted. In addition, the wing must have sufficient bending rigidity from a structural point of view; therefore, the wing thickness is a constraint at some chord-wise stations. This subsection demonstrates how to add a thickness constraint to the parametrization.

Suppose the airfoil is parametrized using four polynomials, as shown in Fig. 2.2: two for the upper surface

$$\begin{aligned} P_1(x) &= a_0\sqrt{x} + a_1 x + a_2 x^2 + a_3 x^3 \qquad 0 < x < L_1 \\ P_2(x) &= b_0 + b_1 x + b_2 x^2 + b_3 x^3 \qquad L_1 < x < L \end{aligned}$$

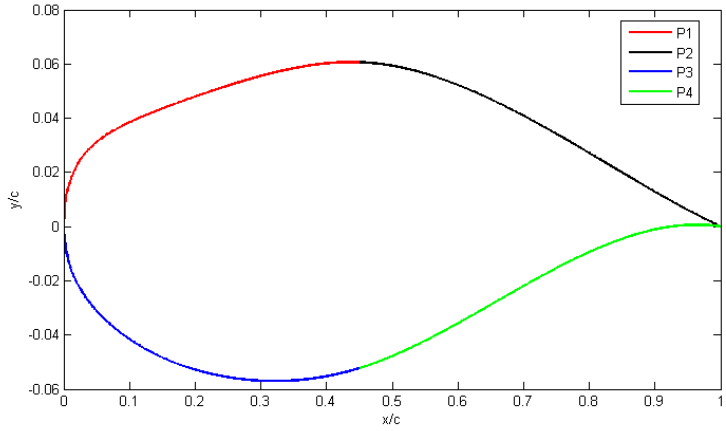


Figure 2.2: Parametrized airfoil using four polynomials

and two for the lower surface

$$\begin{aligned} P_3(x) &= c_0\sqrt{x} + c_1 x + c_2 x^2 + c_3 x^3 \qquad 0 < x < L_1 \\ P_4(x) &= d_0 + d_1 x + d_2 x^2 + d_3 x^3 \qquad L_1 < x < L \end{aligned}$$

If the thickness t_c need to be constrained at a station x_c where $0 < x_c \leq L_1$, this constraint will be

$$C_t : \; P_1(x_c) - P_3(x_c) = t_c \tag{2.15}$$

$$C_t : \; \left(a_0\sqrt{x_c} + a_1 x_c + a_2 x_c^2 + a_3 x_c^3\right) - \left(c_0\sqrt{x_c} + c_1 x_c + c_2 x_c^2 + c_3 x_c^3\right) = t_c$$

Equation 2.15 provides a link between the upper and lower surface polynomials; therefore, a coupled least-squares system needs to be constructed and solved; the

"hard" constraint equations are expressed as

$$\begin{bmatrix} B & 0 \\ 0 & B \\ P_1(x_c)\ \{0\}_{1\times 4} & -P_3(x_c)\ \{0\}_{1\times 4} \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ \vdots \\ d_2 \\ d_3 \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ t_c \end{Bmatrix} \tag{2.16}$$

$$[B_a]\,\{P_a\} = \{d_a\} \tag{2.17}$$

The global least-squares system is

$$\begin{bmatrix} A & 0 \\ 0 & A \end{bmatrix} \{P_a\} = \begin{Bmatrix} c_u \\ c_l \end{Bmatrix} \tag{2.18}$$

$$[A_a]\,\{P_a\} = \{c_a\}$$

The above hard constraint equations do not have a null right hand side due to the thickness constraint; therefore, the solution (polynomials coefficients) does not belong to the null space of B_a. The solution procedure for the constrained least-squares problem expressed by Eq. (2.16,2.18) is described by Masuda et al [52]:

- Apply QR factorization to get $B_a^T = QR$.
- Let $Q = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix}$, $R = \begin{bmatrix} R_1 \\ 0 \end{bmatrix}$ where Q_1contains the first 7 columns of Q and Q_2 contains the rest of the columns, and $R_1$is the first 7×7 sub matrix of R.
- The hard constraints can now be written as $BP_a \equiv R^T Q^T P_a = d_a$
- Let $Q^T P_a = \begin{bmatrix} y \\ z \end{bmatrix}$. Then $P_a = Q \begin{bmatrix} y \\ z \end{bmatrix} = Q_1 y + Q_2 z$

$$\therefore R^T Q^T P_a = \begin{bmatrix} R_1^T \\ 0 \end{bmatrix} \cdot \begin{bmatrix} Q_1 & Q_2 \end{bmatrix}^T P_a = d_a \longmapsto R_1^T y + 0 = d_a$$

$$\therefore y = R_1^{-T} d_a \tag{2.19}$$

- Because R is upper triangular, Equation 2.19 can be solved using forward substitution.

- The soft constraints in equation 2.18 can be rewritten as

$$\begin{aligned} A_a P_a &= A_a \begin{bmatrix} Q_1 y & Q_2 z \end{bmatrix} = c_a && (2.20) \\ \therefore A_a Q_2 z &= c_a - A_a Q_1 y && (2.21) \\ \therefore z &= [A_a Q_2]^{\dagger} [c_a - A_a Q_1 y] && (2.22) \end{aligned}$$

- Finally, the polynomial coefficients can be found as $P_a = Q_1 y + Q_2 z$:

$$P_a = \left[Q_1 - Q_2 [A_a Q_2]^{\dagger} A_a Q_1 \right] R_1^{-T} d_a + Q_2 [A_a Q_2]^{\dagger} c_a \qquad (2.23)$$

  Using the last equation we notice that the sensitivity of the polynomial coefficients with respect to the i^{th} control point y location, y_{ci}, is the i^{th} vector of the matrix $[Q_2] [AQ_2]^{\dagger}$.

2.2.3 Validation cases

In this subsection, testing of the proposed least-squares surface parametrization is carried out; parametrization of various types of airfoils is done to show that the proposed geometry parametrization scheme has the required flexibility to represent various types of airfoils. These airfoils include NACA 4-, 5-, and 6-digit series, laminar flow, supersonic, and super-critical airfoil sections.[4] Figures 2.3–2.10 show the least-squares fitting polynomials for various airfoils, while Table 2.1 shows the RMS error in the parametrized geometry for upper and lower surfaces.

The last parametrized airfoil, the laminar LV2[5] airfoil of German Aerospace Center, was parametrized using 10 polynomials, 5 for each of its surfaces. Although the RMS error is small (order of 10^{-5}chord for all parametrization method), as shown in Table 2.1, Fig. 2.11 shows that the difference of the surface pressure distribution, especially at the peak velocity at the leading edge upper surface, is still significant. There are many x-stations at which LV2 airfoil changes its curvature, and also a

[4]Readers interested in particulars for the NACA airfoils are referred to Abbott and von Doenhoff's textbook [36]. The seminal reference for the RAE airfoil is Cook et al [18].

[5]LV2 geometry obtained by personal contact with German Aerospace Center (DLR) researchers.

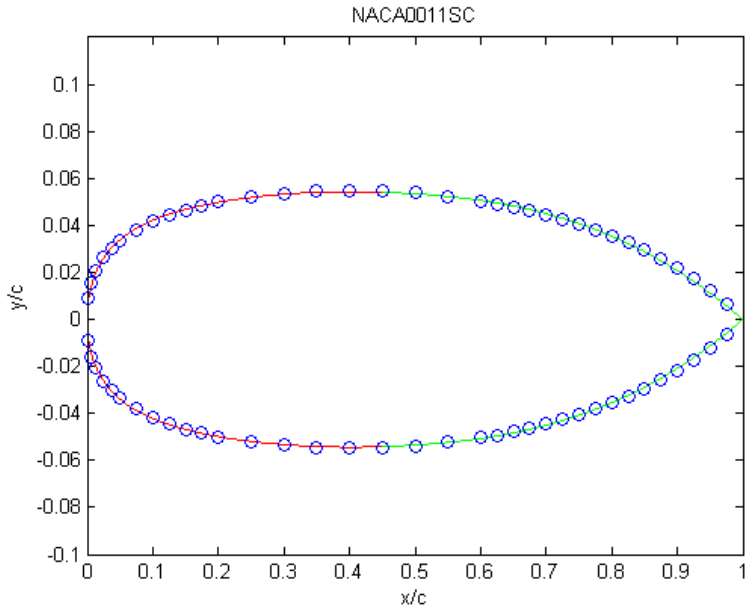


Figure 2.3: Parametrized NACA 0011SC, 'O' are the original airfoil ordinates

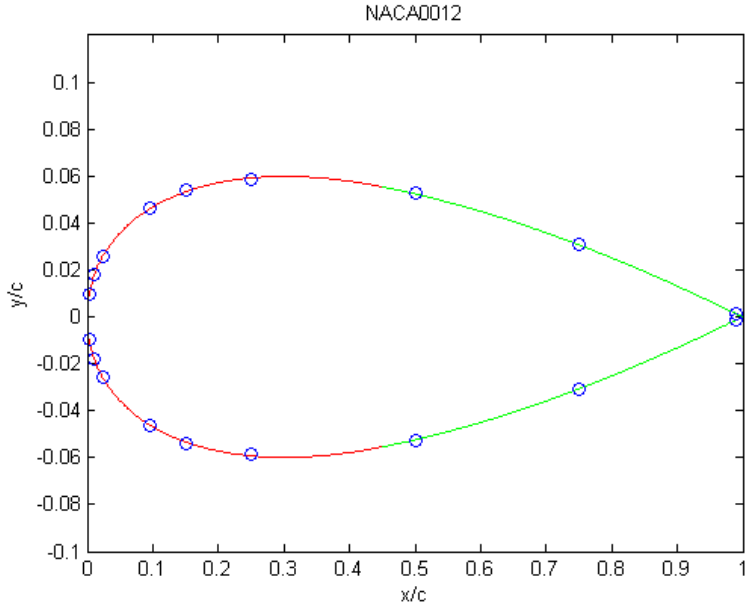


Figure 2.4: Parametrized NACA 0012, 'O' are the original airfoil ordinates

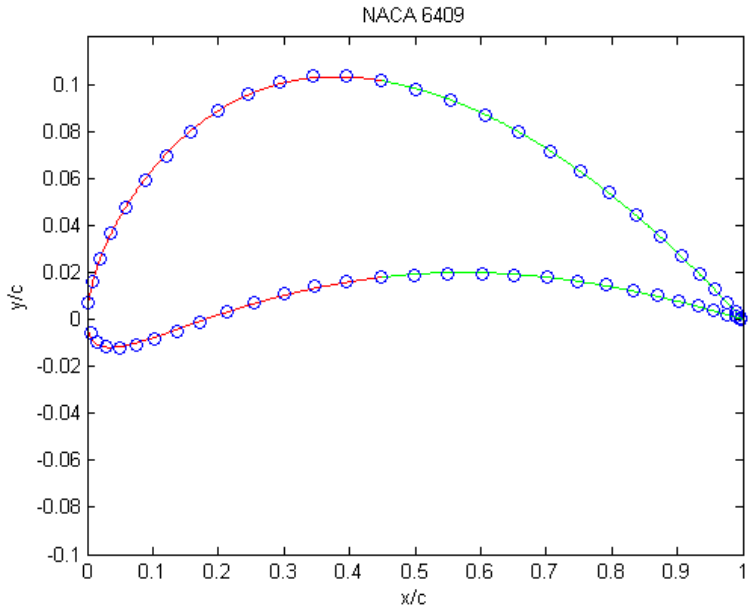


Figure 2.5: Parametrized NACA 6509, 'O' are the original airfoil ordinates

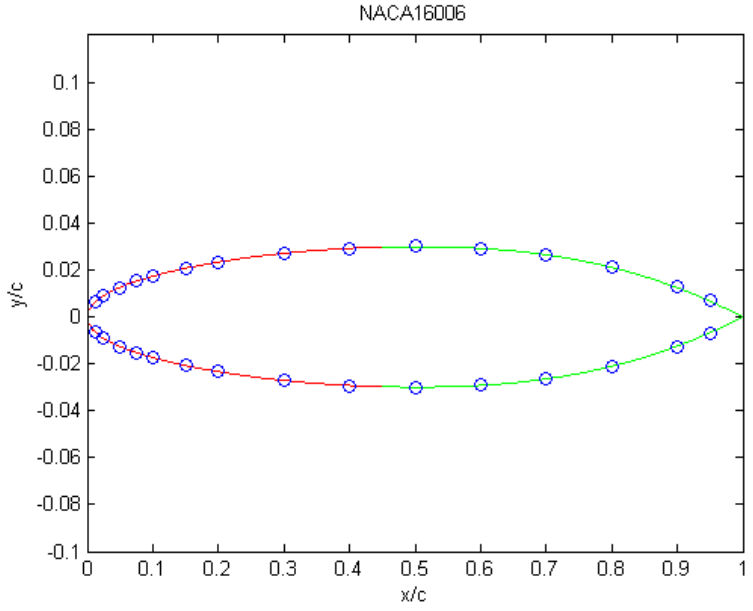


Figure 2.6: Parametrized NACA 16006, 'O' are the original airfoil ordinates

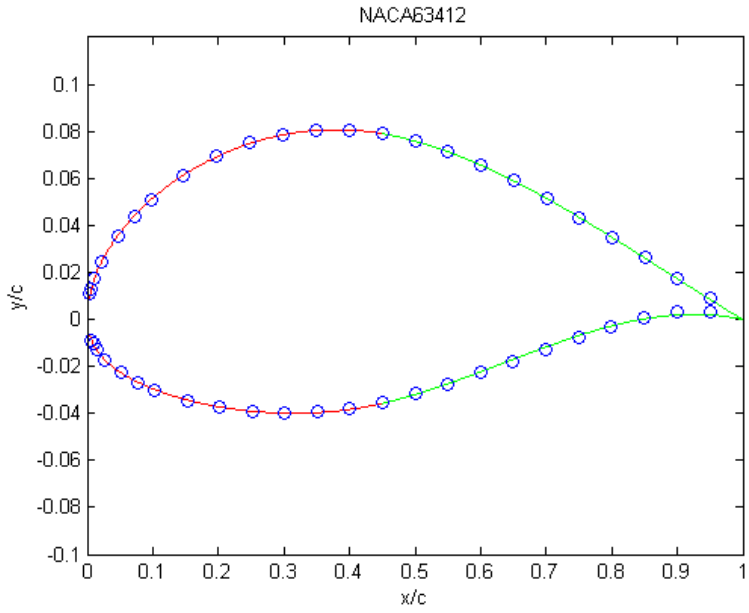


Figure 2.7: Parametrized NACA 63412, 'O' are the original airfoil ordinates

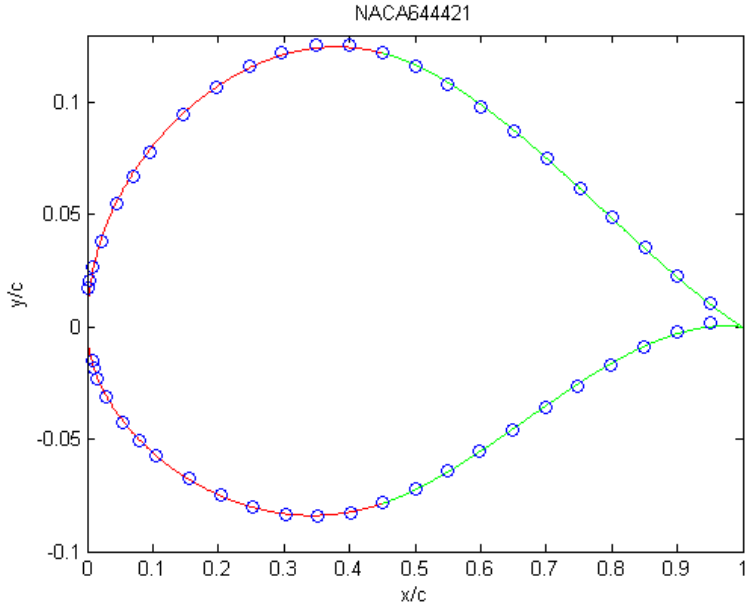


Figure 2.8: Parametrized NACA 644421, 'O' are the original airfoil ordinates

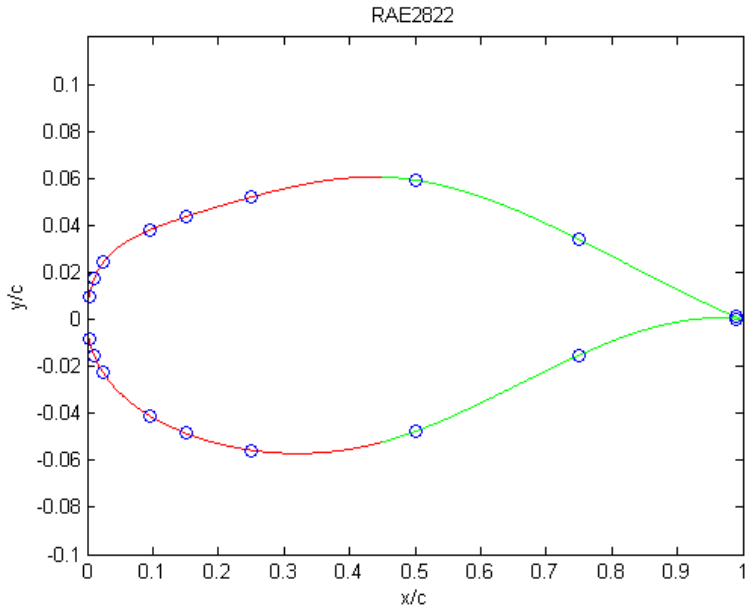


Figure 2.9: Parametrized RAE2822, 'O' are the original airfoil ordinates

long interval of almost infinite curvature value which makes it hard for the geometry parametrization scheme to accurately present it and this causes the fluctuation in pressure resulted in the parametrized airfoil because of lack of accurate presentation of curvature fluctuations. Better matching can be obtained but this will increase the number of geometry design variables significantly. This case illustrates clearly the trade off between accurately representing the geometry (which will change during optimization iteration) and choosing a reasonable number of design variables; this choice must be left for the designer. If the change in pressure distribution becomes unacceptable, or if the pressure distribution requires large number of airfoil control points (design variables) to be accurately represented, parametrization the geometry perturbation is the more attractive option.

The values of RMS error in Table 2.1 are small compared to the maximum airfoil thickness value: only the NACA 0011SC has an RMS error of more than 0.2% of maximum thickness. However, if the RMS error was large, increasing the number of surface parametrization polynomials would reduce the error.

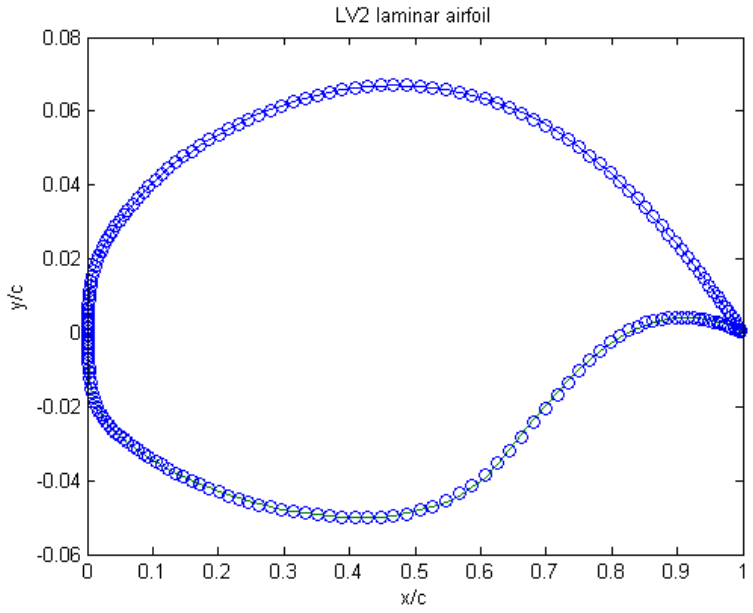


Figure 2.10: Parametrized LV2 laminar airfoil, parametrized using 5 polynomials per surface, 'O' are the original airfoil ordinates

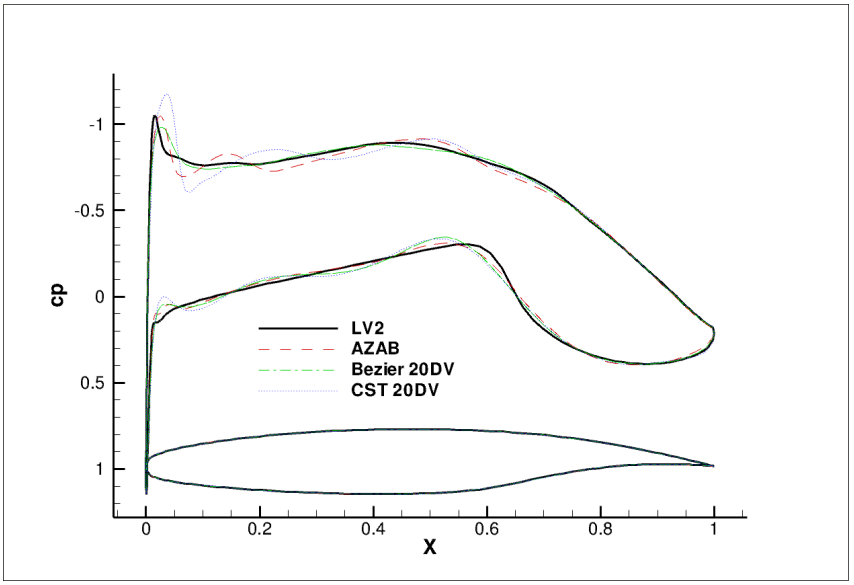


Figure 2.11: LV2 airfoil parametrized using different methods with 20 design variables, presented with permission of Brezillon.

Airfoil	Upper surface error	Lower surface error	Maximum Thickness
NACA 0011SC	$6.37 \cdot 10^{-4}$	$6.37 \cdot 10^{-4}$	$11 \cdot 10^{-2}$
NACA 0012	$2.35 \cdot 10^{-4}$	$2.35 \cdot 10^{-4}$	$12 \cdot 10^{-2}$
NACA 6409	$9.28 \cdot 10^{-5}$	$1.24 \cdot 10^{-4}$	$9 \cdot 10^{-2}$
NACA 16-006	$1.31 \cdot 10^{-6}$	$2.03 \cdot 10^{-5}$	$6 \cdot 10^{-2}$
NACA 63-412	$1.17 \cdot 10^{-5}$	$5.63 \cdot 10^{-5}$	$12 \cdot 10^{-2}$
NACA 64-421	$1.70 \cdot 10^{-4}$	$7.50 \cdot 10^{-5}$	$21 \cdot 10^{-2}$
RAE 2822	$1.20 \cdot 10^{-5}$	$3.20 \cdot 10^{-5}$	slightly $> 12 \cdot 10^{-2}$
LV2	$1.45 \cdot 10^{-5}$	$3.22 \cdot 10^{-5}$	slightly $> 12 \cdot 10^{-2}$

Table 2.1: RMS error in different parametrized airfoil geometries

Chapter 3

MESH MORPHING AND MESH SENSITIVITY

3.1 Mesh Morphing

The modification of the aerodynamic shape during optimization requires a change of the mesh that presents the shape. This can be done by grid regeneration around the new geometry but this is time consuming and will change the discretization error [51]. Another strategy is to adapt the old mesh to fit the new shape of the airfoil using mesh movement.

The tension spring analogy is one of the most widely used mesh deformation strategies for aerodynamic optimization. The main idea is to replace the grid edges by springs with stiffness inversely proportional to their length. The boundary points that lie on the airfoil surface are moved with displacement specified by the optimizer, far field points are kept fixed, and the interior point displacements are determined by equilibrium of the spring network [7]. For large grid displacements, the linear spring analogy is not robust, and negative area cells can present after mesh morphing.

Farhat et. al. improved the tension spring analogy by adding torsional springs at the grid nodes to prevent element flipping [25]. Each edge faces two angles as shown in Fig. 3.1; the edge stiffness is modified to include terms with the reciprocal of the sine of these angles. This allows the edge stiffness to grow to infinity if the angle tends to be zero and therefore prevent element flipping.

Another strategy is to modify the mesh by solving a linear elasticity problem in which the boundary displacements are known [14]; the element modulus of elasticity can be the reciprocal of the distance from the wall, or it can be the reciprocal of the element size [82]. The later strategy was applied by Stein et al. and the results showed that this method is robust, especially for viscous calculations. In this case,

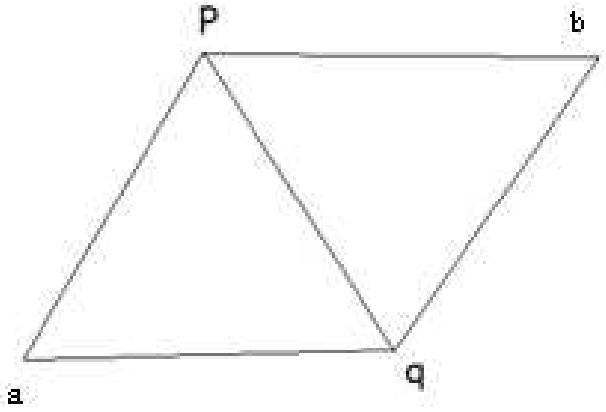


Figure 3.1: Schematic drawing of an edge $\overline{pq}$ and its facing angles a and b.

the elements of the boundary layer experienced small geometrical change while the elements away from the airfoil experienced larger changes [80]. However, the linear elasticity mesh movement scheme is computationally expensive compared to the spring analogy method. The semi-torsional spring analogy method is adopted in this research due to its simplicity and robustness.

Consider the edge $\overline{pq}$ shown in Fig 3.1. The relationship between the forces and the node displacements when treating this edge as a spring follows Hooke's Law as

$$\begin{bmatrix} F_{px} \\ F_{py} \\ F_{qx} \\ F_{qy} \end{bmatrix} = \frac{1}{l_{pq}} \left(1 + \frac{1}{\sin(\theta_a)} + \frac{1}{\sin(\theta_b)} \right) \begin{bmatrix} -1 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix} \begin{Bmatrix} u_{px} \\ u_{py} \\ u_{qx} \\ u_{qy} \end{Bmatrix} \tag{3.1}$$

where l_{pq} is the length of edge $\overline{pq}$, and θ_a and θ_b are the angles facing the edge. After the assembly of the global stiffness matrix, the system of equations that relates grid point displacement with nodal forces can be written as

$$\begin{bmatrix} K_{ii} & K_{ib} \\ K_{bi} & K_{bb} \end{bmatrix} \begin{Bmatrix} U_i \\ U_b \end{Bmatrix} = \begin{Bmatrix} 0 \\ F_b \end{Bmatrix} \tag{3.2}$$

where U_i and U_b are the interior and boundary mesh point displacements respectively. We do not need to know the values of boundary nodal forces F_b because the boundary points displacement vector U_b is known explicitly: it is the deformation required in the airfoil profile to minimize the objective function. Therefore, Eq 3.2 can be written as

$$\begin{bmatrix} K_{ii} & K_{ib} \\ 0 & I \end{bmatrix} \begin{Bmatrix} U_i \\ U_b \end{Bmatrix} = \begin{Bmatrix} 0 \\ U_b \end{Bmatrix} \tag{3.3}$$

Substituting $U_i^k = \overrightarrow{r_i^{k+1}} - \overrightarrow{r_i^k}$, $U_b = \overrightarrow{r_b} - \overrightarrow{r_{bo}}$in Eq 3.3 we get

$$\begin{bmatrix} K_{ii} & K_{ib} \\ 0 & I \end{bmatrix}^k \begin{Bmatrix} \overrightarrow{r_i^{k+1}} \\ \overrightarrow{r_b} \end{Bmatrix} = \begin{bmatrix} K_{ii} & K_{ib} \\ 0 & I \end{bmatrix}^k \begin{Bmatrix} \overrightarrow{r_i^k} \\ \overrightarrow{r_{bo}} \end{Bmatrix} + \begin{Bmatrix} 0 \\ U_b \end{Bmatrix} \tag{3.4}$$

where $\overrightarrow{r_{bo}}$ is the initial position vector of the boundary points before mesh morphing. The stiffness matrices $[K_{ii}]$ and $[K_{ib}]$ depend on the mesh face lengths, which change during the mesh morphing stage; therefore, Eq. 3.4 is a non-linear equation and needs to be solved iteratively.

3.2 Testing Mesh Morphing

Farhat et al [25] demonstrated the robustness of the semi-torsional mesh movement scheme. In this section, testing results are presented to demonstrate that the current implementation of this scheme shows the same good behavior for several high-deformation cases. The first test case is an unstructured triangular mesh around a NACA 0012. The thickness of the airfoil is doubled, which means that the airfoil surface points will translate by several computational cells in the y-direction. Figure 3.2 shows that even with this large displacement (multiple cells) of the boundary, no cells are inverted and no edges intersected with another. Displacement decreases with the distance from the airfoil surface, and there is almost no movement on the symmetry line.

The second case tests mesh movement when the outer boundary is changing. The original mesh is an unstructured triangular mesh (shown in Fig 3.3a). The outer boundaries are changed, reducing the total area by almost 50% and turning the original right-angled corners nearly into cusps. Figure 3.3b shows that the semi-

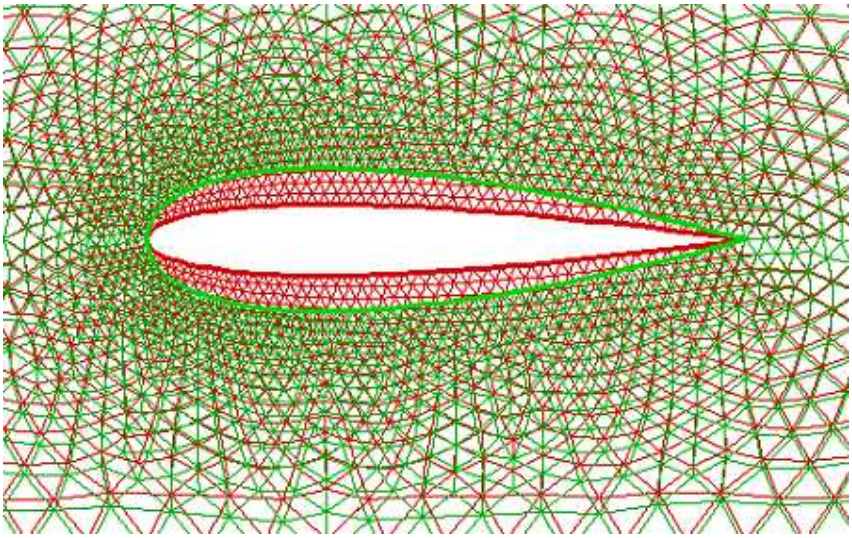


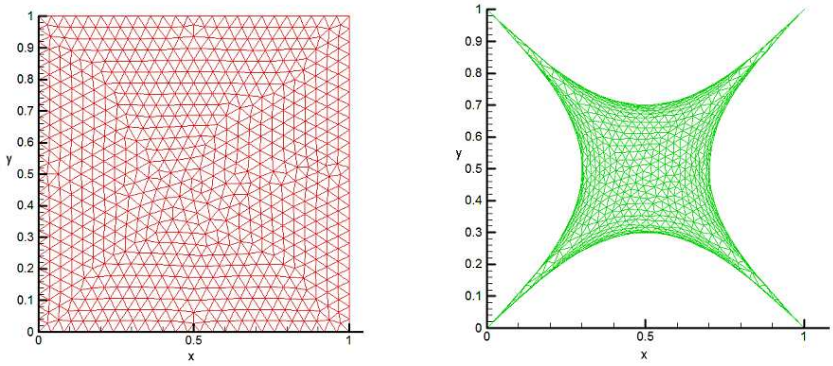
Figure 3.2: Mesh movement scheme results of doubling the thickness of NACA 0012

torsional mesh movement was capable of adapting the mesh in the entire field without element flipping.

3.3 Mesh Sensitivity

For gradient-based optimization, as shown in the next chapter, the gradient computation requires calculation of the dependency of the residual on the design variables $\partial R/\partial D$, which in turn depends on the evaluation of mesh sensitivity. The mesh sensitivity tells how mesh points translate in the (x, y) plane with the perturbation of the geometric parametrization control points (design variables). This translation, obviously, depends on the mesh movement scheme. Truong et. al. compared algebraic mesh movement to linear elasticity mesh movement schemes to study the impact of the adopted mesh movement scheme on the final optimized shape. There was a noticeable difference between the final optimal airfoil shapes for a subsonic case, although the difference in the optimal objective function value was of order of discretization error. For a transonic test case, the difference in the optimal shapes was almost negligible [83].

The mesh sensitivity with respect to one of the design variables $\partial M/\partial D_i$ (that is,



(a) Initial mesh (b) Final mesh

Figure 3.3: Mesh movement results, large outer boundary deformation of a rectangular domain

the change in mesh point locations with a change of a design variable) is calculated by differentiating Eq. 3.3:

$$\begin{bmatrix} K_{ii} & K_{ib} \\ 0 & I \end{bmatrix} \begin{Bmatrix} \frac{\partial U_i}{\partial D_i} \\ \frac{\partial U_b}{\partial D_i} \end{Bmatrix} = \begin{Bmatrix} 0 \\ \frac{\partial \overrightarrow{r_b}}{\partial D_i} \end{Bmatrix}$$

$$\therefore \left\{ \frac{\partial M}{\partial D_i} \right\} = \begin{Bmatrix} \frac{\partial U_i}{\partial D_i} \\ \frac{\partial U_b}{\partial D_i} \end{Bmatrix} = \begin{bmatrix} K_{ii} & K_{ib} \\ 0 & I \end{bmatrix}^{-1} \begin{Bmatrix} 0 \\ \frac{\partial \overrightarrow{r_b}}{\partial D_i} \end{Bmatrix} \tag{3.5}$$

where $\partial \overrightarrow{r_b} / \partial D_i$ is obtained using Eq. 2.14 and is related to the design variables via the pseudo inverse of the constrained least-squares system solved in parametrizing the geometry.

Chapter 4

GRADIENT CALCULATION USING ADJOINT APPROACH

Gradient calculation plays a key role in gradient-based optimization. The traditional finite difference strategy is computationally expensive as it requires at least as many CFD simulations as the number of design variables to compute the gradient; each of these is the solution to a large non-linear system of equations. The *forward strategy* to compute the gradient is less expensive as it requires the calculation of the flow sensitivity with respect to the design variables, and uses the computed flow sensitivity to compute the gradient; the forward strategy requires solving a number of *linear* systems equal to the number of design variables to find the flow property sensitivity with respect to all the design variables. The adjoint strategy is computationally cheaper; it requires the solution of one linear system whose right hand side is the dependency of the aerodynamic objective function on the flow field properties. Due to its numerical efficiency and the corresponding reduction in computational effort, the adjoint strategy is adopted in this research.

4.1 Forward and Adjoint Formulations

The objective function, F, for aerodynamic optimization is a function of the design variables, D, and the flow field solution at the surface points of the boundary control volumes U_s

$$F = F\left(U_s, D\right) \tag{4.1}$$

U_s is expressed most conveniently in primitive variables: $U = \begin{pmatrix} \rho & u & v & P \end{pmatrix}^T$. Consider, for instance, the lift and drag forces of a two dimensional airfoil, which are perpendicular and parallel, respectively, to the incoming flow, which is inclined

at an angle α to the airfoil chord. These can be evaluated as follows:

$$F_L = -\left\{\oint P_s n_x \, ds\right\} \sin\alpha + \left\{\oint P_s n_y \, ds\right\} \cos\alpha$$
$$F_D = \left\{\oint P_s n_x \, ds\right\} \cos\alpha + \left\{\oint P_s n_y \, ds\right\} \sin\alpha$$

or in discrete form,

$$F_L = -\sum w_s P_s n_x \sin\alpha + \sum w_s P_s n_y \cos\alpha \tag{4.2}$$
$$F_D = \sum w_s P_s n_x \cos\alpha + \sum w_s P_s n_y \sin\alpha \tag{4.3}$$

where P_s is the pressure at surface integration point pressure, n_x and n_y are the unit normal components at the surface integration point, α is the angle of attack, and w_s is the arc length associated with the surface integration point. Note that this form uses the dimensional pressure and coordinates gives the dimensional lift and drag forces. The lift and drag coefficients, which are the non-dimensional equivalents, are identical in form but use non-dimensional pressure and coordinates.

These discrete integrals expressed as a function of geometric and flow properties of the control volume such as the length of each face and the unit normal at each Gauss point. The geometric properties depend on the design variables through the mesh sensitivity, while the flow properties at the Gauss points depend on the flow properties of the control volume itself and its neighbors, which in turns depend on the mesh and the boundary shape which ultimately depends on the geometric design variables. The gradient of the objective function can be obtained by using the chain rule

$$\frac{dF}{dD} = \frac{\partial F}{\partial U_{bg}} \frac{\partial U_{bg}}{\partial U} \frac{\partial U}{\partial M} \frac{\partial M}{\partial D} + \frac{\partial F}{\partial M} \frac{\partial M}{\partial D} \tag{4.4}$$

where U_{bg} is the boundary Gauss point flow properties, U is the control volume averaged solution written in primitive variables, and M is the mesh point locations. $\partial M/\partial D$ is the mesh dependency on the design variables which computations are presented in details in Chapter 3. The residual of the flow governing equations can be written as a function of the flow field solution U and mesh geometric design variables D. If we apply the constraint that the flow solution is converged regardless

of variations in the design variables, we can write

$$\frac{dR}{dD} = \frac{\partial R}{\partial M}\frac{\partial M}{\partial D} + \frac{\partial R}{\partial U}\frac{\partial U}{\partial M}\frac{\partial M}{\partial D} = 0 \tag{4.5}$$

Solving this for the solution sensitivity $\frac{\partial U}{\partial D} \equiv \frac{\partial U}{\partial M}\frac{\partial M}{\partial D}$, we get

$$\frac{\partial U}{\partial M}\frac{\partial M}{\partial D} = -\left[\frac{\partial R}{\partial U}\right]^{-1} \cdot \frac{\partial R}{\partial M}\frac{\partial M}{\partial D} = -\left[\frac{\partial R}{\partial Q}\frac{\partial Q}{\partial U}\right]^{-1} \cdot \left\{\frac{\partial R}{\partial M}\frac{\partial M}{\partial D}\right\} \tag{4.6}$$

where the last equality expands the residual Jacobian with respect to the primitive flow variables into the product of the residual Jacobian with respect to the conserved flow variables (which is used in implicit flow solvers) and a change of variables for conserved to non-conserved. Note that $\frac{\partial Q}{\partial U}$ is a block diagonal matrix.

Substituting Eq.4.6 in Eq. 4.4, we get the forward formulation for gradient computation

$$\frac{dF}{dD} = -\frac{\partial F}{\partial U_{bg}}\frac{\partial U_{bg}}{\partial U}\left[\frac{\partial R}{\partial Q}\frac{\partial Q}{\partial U}\right]^{-1} \cdot \left\{\frac{\partial R}{\partial M}\frac{\partial M}{\partial D}\right\} + \frac{\partial F}{\partial M}\frac{\partial M}{\partial D} \tag{4.7}$$

This form of the gradient requires solving as many linear systems as there are design variables in the optimization problem.

Taking the transpose of Eq. 4.7,

$$\frac{dF}{dD}^T = -\left\{\frac{\partial R}{\partial M}\frac{\partial M}{\partial D}\right\}^T \cdot \left[\frac{\partial R}{\partial Q}\frac{\partial Q}{\partial U}\right]^{-T} \left\{\frac{\partial F}{\partial U_{bg}}\frac{\partial U_{bg}}{\partial U}\right\}^T + \left\{\frac{\partial F}{\partial M}\frac{\partial M}{\partial D}\right\}^T \tag{4.8}$$

where the residual sensitivity to mesh movement can be written using the chain rule as:

$$\begin{aligned}\frac{\partial R}{\partial M} \quad = \quad & \frac{\partial R}{\partial A_{\Omega_i}}\frac{\partial A_{\Omega_i}}{\partial M} + \sum_{\text{faces}\,\Omega_i}\left\{\frac{\partial R}{\partial n_x}\frac{dn_x}{dM} + \frac{\partial R}{\partial n_y}\frac{dn_y}{dM} + \right. \\ & \left.\frac{\partial R}{\partial w_i}\frac{dw_i}{dM} + \frac{\partial R}{\partial U_{\text{face}}}\frac{\partial U_{\text{face}}}{\partial M}\right\}\end{aligned} \tag{4.9}$$

We get the adjoint method, presented by A. Jameson [37, 41, 43, 38, 40]. Now only one linear system solve is required. However, this linear system solve requires

explicitly forming the global Jacobian matrix $\frac{\partial R}{\partial Q}$ because its transpose is required. Because the CFD solver used in this thesis can form the global Jacobian matrix explicitly **(author?)** [57], the transpose of the Jacobian can easily be formed as well. The Jacobian from the last GMRES iteration is re-used, so the computational effort of solving the adjoint problem is reduced to solving this linear system, a cost on the order of 1% of the CFD simulation computational effort.

To ease the programming effort when changing the objective function, we form three adjoint problems, one each for the lift coefficient C_L, the drag coefficient C_D, and the moment coefficient C_M to find $\partial C_L/\partial D$, $\partial C_D/\partial D$ and $\partial C_M/\partial D$, respectively. These aerodynamic coefficient gradients can be used to evaluate any objective function gradient for a function that depends on aerodynamic forces:

$$\frac{dF}{dD} = f\left(\frac{dC_L}{dD}, \frac{dC_D}{dD}, \frac{dC_M}{dD}\right)$$

The solution procedure of the three adjoint problems can be summarized as follows,

- Using the steady state flow solution, we construct the CFD simulation Jacobian matrix $\partial R/\partial Q$ expressed in Eq. 1.6.
- We construct $\partial R/\partial U$ as:
$$\frac{\partial R}{\partial U} = \frac{\partial R}{\partial Q}\frac{\partial Q}{\partial U}$$
where $\partial Q/\partial U$ is the transformation matrix from conservative to primitive flow variables and is block diagonal.
- We construct $\frac{\partial F}{\partial U_{bg}}\frac{\partial U_{bg}}{\partial U}$, where $\partial F/\partial U_{bg}$ is the analytic dependency of the objective function on the primitive flow properties at the airfoil surface points, and $\partial U_{bg}/\partial U$ is the dependency of surface point primitive flow properties on the control volume average values of the primitive flow properties. The latter is known as a side effect of solution reconstruction.
- We solve the three linear systems $\left[\frac{\partial R}{\partial U}\right]^T \Psi_{L,D,M} = \left[\frac{\partial C_{L,D,M}}{\partial U_{bg}}\frac{\partial U_{bg}}{\partial U}\right]^T$ to get the adjoint vectors $\Psi_{L,D,M}$.
- We construct the objective function gradient vector $\frac{dC_{L,D,M}}{dD}^T = \left\{\frac{\partial F}{\partial M}\frac{\partial M}{\partial D}\right\}^T - \left\{\frac{\partial R}{\partial M}\frac{\partial M}{\partial D}\right\}^T \Psi_{L,D,M}$, which requires only a vector dot product for each design

variable.

As an example of how to use $\frac{dC_{L,D,M}}{dD}$ to construct the gradient of an aerodynamic function, consider the following objective function which represents a typical drag minimization function with a lift constraint applied using a penalty term

$$F = C_D + k_1 \left(C_L - C_{Lc}\right)^2 \tag{4.10}$$

The gradient of the above function can be written as

$$\frac{dF}{dD} = \frac{dC_D}{dD} + 2k_1 \left(C_L - C_{Lc}\right) \frac{dC_L}{dD} \tag{4.11}$$

also

$$\frac{\partial F}{\partial D} = \frac{\partial C_D}{\partial D} + 2k_1 \left(C_L - C_{Lc}\right) \frac{\partial C_L}{\partial D} \tag{4.12}$$

The discrete (inviscid) forms of C_L and C_D follow from the non-dimensionalization of Eqs. 4.2 and 4.3, where now the pressure and coordinates are non-dimensionalized:

$$\begin{aligned} C_L &= -\left(\sum w_s P_s n_x\right) \sin\alpha + \left(\sum w_s P_s n_y\right) \cos\alpha \\ C_D &= \left(\sum w_s P_s n_x\right) \cos\alpha + \left(\sum w_s P_s n_y\right) \sin\alpha \end{aligned} \tag{4.13}$$

and their partial derivatives with respect to the geometry design variables are

$$\begin{aligned} \frac{\partial C_L}{\partial D} &= -\left(\sum \frac{\partial w_s}{\partial D} P_s n_x\right) \sin\alpha + \left(\sum \frac{\partial w_s}{\partial D} P_s n_y\right) \cos\alpha \\ &\quad -\left(\sum w_s P_s \frac{\partial n_x}{\partial D}\right) \sin\alpha + \left(\sum w_s P_s \frac{\partial n_y}{\partial D}\right) \cos\alpha \\ \frac{\partial C_D}{\partial D} &= \left(\sum \frac{\partial w_s}{\partial D} P_s n_x\right) \cos\alpha + \left(\sum \frac{\partial w_s}{\partial D} P_s n_y\right) \sin\alpha + \\ &\quad \left(\sum w_s P_s \frac{\partial n_x}{\partial D}\right) \cos\alpha + \left(\sum w_s P_s \frac{\partial n_y}{\partial D}\right) \sin\alpha \end{aligned} \tag{4.14}$$

Using Eq. 4.9 to calculate $\partial R/\partial M$, we need to compute the terms $\partial A_{\Omega_i}/\partial M$, $\partial n_x/\partial M$, $\partial n_y/\partial M$, and $\partial w_i/\partial M$, which only depend on mesh points' spatial locations, while the terms $\partial R/\partial n_x$ and $\partial R/\partial n_y$ are obtained by direct differentiation of the Roe flux, and finally $(\partial R/\partial U_{face}) \cdot (\partial U_{face}/\partial M)$ is obtained by differentiating

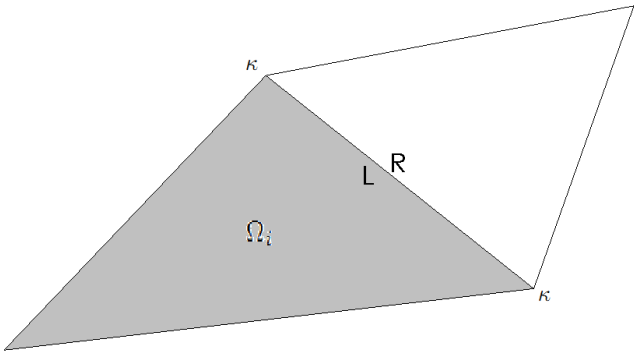


Figure 4.1: Schematic drawing of an element face κ, and illustration of its left and right sides

the face property reconstruction scheme which depends on CFD solver technology. The remainder of this section will focus on the derivatives of the residual R, with the derivatives on the geometric terms in the following section.

Let us consider the residual contribution of face κ of the control volume Ω_i shown in Fig. 4.1.

The discrete form of this edge's residual contribution to the control volume Ω_i is

$$R_{\kappa,\Omega_i} = \frac{1}{A_{\Omega_i}} \sum_{j=1}^{J} \left\{ \frac{1}{2} \left(\vec{F}_{R_j} + \vec{F}_{L_j} - |\triangle F_1|_j - |\triangle F_4|_j - |\triangle F_5|_j \right) w_j \right\} \tag{4.15}$$

where $J = 1$ when using one Gauss integration point at the middle of faceκ for a second order accurate scheme. For the fourth order scheme, we have two Gauss integration points, i.e. $J = 2$, on the face κ. w_j is the integration weight associated with the Gauss point j.

To find $\partial R_{\kappa,\Omega_i}/\partial A_{\Omega_i}$, Eq. 4.15 can be directly differentiated to get

$$\frac{\partial R_{\kappa,\Omega_i}}{\partial A_{\Omega_i}} = -\frac{1}{(A_{\Omega_i})^2}\sum_j \left\{ \frac{1}{2}\left(\vec{F}_{R_j} + \vec{F}_{L_j} - |\triangle F_1|_j - |\triangle F_4|_j - |\triangle F_5|_j \right) w_j \right\} = -\frac{1}{A_{\Omega_i}} R_{\kappa,\Omega_i} \tag{4.16}$$

The term $\partial R_{\kappa,\Omega_i}/\partial n_{x,y}$ can be written as

$$\frac{\partial R_{\kappa,\Omega_i}}{\partial n_{x,y}} = \frac{1}{A_{\Omega_i}}\sum_j \left\{ \frac{1}{2}\left(\frac{\partial \vec{F}_{R_j}}{\partial n_{x,y}} + \frac{\partial \vec{F}_{L_j}}{\partial n_{x,y}} - \frac{\partial |\triangle F_1|_j}{\partial n_{x,y}} - \frac{\partial |\triangle F_4|_j}{\partial n_{x,y}} - \frac{\partial |\triangle F_5|_j}{\partial n_{x,y}} \right) w_j \right\} \tag{4.17}$$

where component terms can be expanded using the definition of the Roe flux to give:

$$\frac{\partial \vec{F}_{R,L_j}}{\partial n_x} = \begin{bmatrix} (\rho u)_{R,L} \\ \left(\rho u^2 + p\right)_{R,L} \\ (\rho u v)_{R,L} \\ (\{E+p\}\, u)_{R,L} \end{bmatrix}, \qquad \frac{\partial \vec{F}_{R,L_j}}{\partial n_y} = \begin{bmatrix} (\rho v)_{R,L} \\ (\rho u v)_{R,L} \\ \left(\rho v^2 + p\right)_{R,L} \\ (\{E+p\}\, v)_{R,L} \end{bmatrix}$$

and

$$\begin{aligned} \frac{\partial |\triangle F_1|_j}{\partial n_x} \;=\; & \frac{u\tilde{U}_n}{\sqrt{\tilde{U}_n^2}} \left\{ \left(\triangle\rho - \frac{\triangle P}{\tilde{a}^2} \right) \begin{bmatrix} 1 \\ \tilde{u} \\ \tilde{v} \\ \frac{\tilde{u}^2+\tilde{v}^2}{2} \end{bmatrix} + \tilde{\rho} \begin{bmatrix} 0 \\ \triangle u - n_x \triangle U \\ \triangle v - n_y \triangle U \\ \tilde{u}\triangle u + \tilde{v}\triangle v - \tilde{U_n}\triangle U \end{bmatrix} \right\} + \\ & \left|\tilde{U}_n\right| \left\{ \tilde{\rho} \begin{bmatrix} 0 \\ -\triangle U - n_x \triangle u \\ -n_y \triangle v \\ -\tilde{u}\triangle U - \tilde{U_n}\triangle u \end{bmatrix} \right\} \end{aligned}$$

$$\frac{\partial \left|\triangle F_1\right|_j}{\partial n_y} = \frac{v\tilde{U}_n}{\sqrt{\tilde{U}_n^2}} \left\{ \left(\triangle\rho - \frac{\triangle P}{\tilde{a}^2} \right) \begin{bmatrix} 1 \\ \tilde{u} \\ \tilde{v} \\ \frac{\tilde{u}^2+\tilde{v}^2}{2} \end{bmatrix} + \tilde{\rho} \begin{bmatrix} 0 \\ \triangle u - n_x \triangle U \\ \triangle v - n_y \triangle U \\ \tilde{u}\triangle u + \tilde{v}\triangle v - \tilde{U_n}\triangle U \end{bmatrix} \right\} + \left|\tilde{U}_n\right| \left\{ \tilde{\rho} \begin{bmatrix} 0 \\ -n_x \triangle u \\ -\triangle U - n_y \triangle v \\ -\tilde{v}\triangle U - \tilde{U_n}\triangle v \end{bmatrix} \right\}$$

$$\frac{\partial \left|\triangle \tilde{F}_{4,5}\right|}{\partial n_x} = \frac{\left(\tilde{U}_n \pm \tilde{a}\right)\tilde{u}}{\sqrt{\left(\tilde{U}_n \pm \tilde{a}\right)^2}} \left(\frac{\triangle P \pm \tilde{\rho}\tilde{a}\triangle U}{2\tilde{a}^2} \right) \begin{bmatrix} 1 \\ \tilde{u} \pm n_x\tilde{a} \\ \tilde{v} \pm n_y\tilde{a} \\ \tilde{h_o} \pm \tilde{U}_n\tilde{a} \end{bmatrix} + \left|\tilde{U}_n \pm \tilde{a}\right| \left(\frac{\pm\tilde{\rho}\tilde{a}\triangle u}{2\tilde{a}^2} \right) \begin{bmatrix} 1 \\ \tilde{u} \pm n_x\tilde{a} \\ \tilde{v} \pm n_y\tilde{a} \\ \tilde{h_o} \pm \tilde{U}_n\tilde{a} \end{bmatrix} + \left|\tilde{U}_n \pm \tilde{a}\right| \left(\frac{\triangle P \pm \tilde{\rho}\tilde{a}\triangle U}{2\tilde{a}^2} \right) \begin{bmatrix} 0 \\ \pm\tilde{a} \\ 0 \\ \pm\tilde{u}\tilde{a} \end{bmatrix}$$

$$
\begin{aligned}
\frac{\partial \left|\triangle \tilde{F}_{4,5}\right|}{\partial n_y} &= \frac{\left(\tilde{U}_n \pm \tilde{a}\right)\tilde{v}}{\sqrt{\left(\tilde{U}_n \pm \tilde{a}\right)^2}} \left(\frac{\triangle P \pm \tilde{\rho}\tilde{a}\triangle U}{2\tilde{a}^2}\right) \begin{bmatrix} 1 \\ \tilde{u} \pm n_x\tilde{a} \\ \tilde{v} \pm n_y\tilde{a} \\ \tilde{h_o} \pm \tilde{U}_n\tilde{a} \end{bmatrix} + \\
& \left|\tilde{U}_n \pm \tilde{a}\right| \left(\frac{\pm\tilde{\rho}\tilde{a}\triangle v}{2\tilde{a}^2}\right) \begin{bmatrix} 1 \\ \tilde{u} \pm n_x\tilde{a} \\ \tilde{v} \pm n_y\tilde{a} \\ \tilde{h_o} \pm \tilde{U}_n\tilde{a} \end{bmatrix} + \\
& \left|\tilde{U}_n \pm \tilde{a}\right| \left(\frac{\triangle P \pm \tilde{\rho}\tilde{a}\triangle U}{2\tilde{a}^2}\right) \begin{bmatrix} 0 \\ 0 \\ \pm\tilde{a} \\ \pm\tilde{v}\tilde{a} \end{bmatrix}
\end{aligned}
$$

To find $\partial R_{\kappa,\Omega_i}/\partial w_j$, Eq. 4.15 is differentiated; we get

$$
\frac{\partial R_{\kappa,\Omega_i}}{\partial w_j} = \frac{1}{(A_{\Omega_i})} \left\{ \frac{1}{2} \left(\vec{F}_{R_j} + \vec{F}_{L_j} - |\triangle F_1|_j - |\triangle F_4|_j - |\triangle F_5|_j \right) \right\} \tag{4.18}
$$

The dependency of the face residual contribution R_{κ,Ω_i} on the reconstructed flow properties at the face U_{face} can found as the sum of the dependency on the right and left face flow properties as

$$
\begin{aligned}
\frac{\partial R_{\kappa,\Omega_i}}{\partial U_{\text{face}}} &= \frac{\partial R_{\kappa,\Omega_i}}{\partial U_R} + \frac{\partial R_{\kappa,\Omega_i}}{\partial U_L} \qquad (4.19) \\
&= \frac{1}{(A_{\Omega_i})} \sum \frac{1}{2} \left\{ \left(\frac{\partial \vec{F}_{R_j}}{\partial U_R} + \frac{\partial \vec{F}_{L_j}}{\partial U_L} - \frac{\partial \left\{ |\triangle F_1|_j + |\triangle F_4|_j + |\triangle F_5|_j \right\}}{\partial U_R} \right. \right. \\
&\left. \left. - \frac{\partial \left\{ |\triangle F_1|_j + |\triangle F_4|_j + |\triangle F_5|_j \right\}}{\partial U_L} \right) w_j \right\}
\end{aligned}
$$

The terms in Eq. 4.19 can be obtained using symbolic manipulator like Maple® or Matlab®. Another possible approach is to use an automatic differentiation package to differentiate the C, C++, or FORTRAN function that computes the face flux.

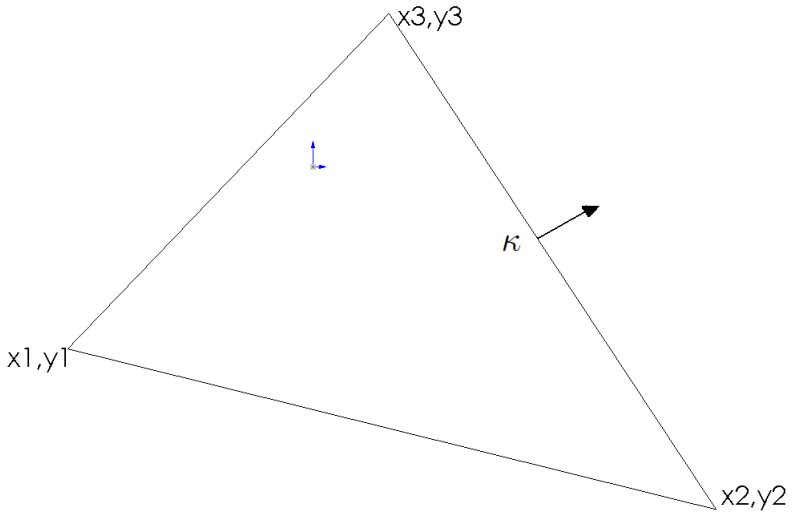


Figure 4.2: General triangular element with unit normal $\hat{n}$ on one of its faces κ.

Finite differences can also be easily implemented to evaluate $\partial R_{\kappa\,\Omega_i}/\partial U_{\text{face}}$

$$\frac{\partial R_{\kappa,\Omega_i}}{\partial U_{\text{face}}} = \frac{R_{\kappa,\Omega_i}\left(U_R+\epsilon\right) - R_{\kappa,\Omega_i}\left(U_R-\epsilon\right)}{2\epsilon} + \frac{R_{\kappa,\Omega_i}\left(U_L+\epsilon\right) - R_{\kappa,\Omega_i}\left(U_L-\epsilon\right)}{2\epsilon} + o\left(\epsilon\right)^2 \tag{4.20}$$

where ϵ is chosen to be 10^{-8} in the above central finite difference formula so that the error will be on the order of machine zero. The ANSLib CFD code can use both approaches; they lead to nearly identical answers.

4.2 Element and Face Geometric Properties Dependency on Mesh Coordinates

Element and face geometric properties depend directly on the spatial coordinates of the vertices. Fig. 4.2 shows a schematic drawing of a triangular element used for cell centered finite volume simulations, with its three vertices and one of its three faces labeled for later reference. In the next subsections, the procedure for evaluating geometric properties like element area, face length, and face normals will be presented in addition to the mesh coordinate dependency of these properties.

4.2.1 Element area mesh dependency

The element area of the triangular element shown in Fig 4.2can be obtained as half of the cross product of the two vectors $\overrightarrow{r_{12}}, \overrightarrow{r_{13}}$

$$A = \sigma \frac{(x_2 - x_1)(y_3 - y_1) - (y_2 - y_1)(x_3 - x_1)}{2} \tag{4.21}$$

The vertices' spatial locations are related to the airfoil surface points via the mesh movement scheme, while the airfoil surface points are related to the design variables through the pseudo inverse matrix in the least-squares fit during geometry parametrization. It worth mentioning that any general polygon can be split into several triangles and the area of each triangle can be calculated using Eq. 4.21.

The element area mesh dependency can be calculated by direct differentiation of Eq. 4.21

$$\begin{aligned} \frac{dA}{dM} &= \frac{\partial A}{\partial x_1}\frac{\partial x_1}{\partial M} + \frac{\partial A}{\partial x_2}\frac{\partial x_2}{\partial M} + \frac{\partial A}{\partial x_3}\frac{\partial x_3}{\partial M} \\ &+ \frac{\partial A}{\partial y_1}\frac{\partial y_1}{\partial M} + \frac{\partial A}{\partial y_2}\frac{\partial y_2}{\partial M} + \frac{\partial A}{\partial y_3}\frac{\partial y_3}{\partial M} \end{aligned} \tag{4.22}$$

where

$$\begin{aligned} \frac{\partial A}{\partial x_1} &= \frac{\sigma (y_2 - y_3)}{2} \\ \frac{\partial A}{\partial x_2} &= \frac{\sigma (y_3 - y_1)}{2} \\ \frac{\partial A}{\partial x_3} &= \frac{\sigma (y_1 - y_2)}{2} \end{aligned}$$

$$\begin{aligned} \frac{\partial A}{\partial y_1} &= \frac{\sigma (x_3 - x_2)}{2} \\ \frac{\partial A}{\partial y_2} &= \frac{\sigma (x_1 - x_3)}{2} \\ \frac{\partial A}{\partial y_3} &= \frac{\sigma (x_2 - x_1)}{2} \end{aligned}$$

$$\sigma = \mathrm{sgn}\left(\left(x_2 - x_1\right)\left(y_3 - y_1\right) - \left(y_2 - y_1\right)\left(x_3 - x_1\right)\right)$$

4.2.2 Face length mesh dependency

The length of face κ in Fig. 4.2, which is a general straight face in the mesh, can be calculated as

$$L = \sqrt{\left(x_3(M) - x_2(M)\right)^2 + \left(y_3(M) - y_2(M)\right)^2} \tag{4.23}$$

The face length mesh dependency can be obtained as

$$\frac{dL}{dM} = \frac{\partial L}{\partial x_2}\frac{\partial x_2}{\partial M} + \frac{\partial L}{\partial x_3}\frac{\partial x_3}{\partial M} + \frac{\partial L}{\partial y_2}\frac{\partial y_2}{\partial M} + \frac{\partial\, L}{\partial y_3}\frac{\partial y_3}{\partial M}$$

where

$$\begin{aligned}
\frac{\partial L}{\partial x_2} &= \frac{-x_3 + x_2}{\sqrt{\left(x_3 - x_2\right)^2 + \left(y_3 - y_2\right)^2}} = \frac{-\left(x_3 - x_2\right)}{L} \\
\frac{\partial L}{\partial x_3} &= \frac{x_3 - x_2}{\sqrt{\left(x_3 - x_2\right)^2 + \left(y_3 - y_2\right)^2}} = \frac{\left(x_3 - x_2\right)}{L} \\
\frac{\partial L}{\partial y_2} &= \frac{-y_3 + y_2}{\sqrt{\left(x_3 - x_2\right)^2 + \left(y_3 - y_2\right)^2}} = \frac{-\left(y_3 - y_2\right)}{L} \\
\frac{\partial L}{\partial y_3} &= \frac{y_3 - y_2}{\sqrt{\left(x_3 - x_2\right)^2 + \left(y_3 - y_2\right)^2}} = \frac{\left(y_3 - y_2\right)}{L}
\end{aligned}$$

The previous relations are for straight faces which exist all over the domain interior, but for higher order schemes, the boundary faces are curved to enable higher order flux integration. A boundary element with one curved face is shown in Fig. 4.3

The curved face length is obtained using numerical integration, therefore to avoid differentiating the numerical integration scheme, $\partial L/\partial x_{2,3}$ and $\partial L/\partial y_{2,3}$ are evalu-

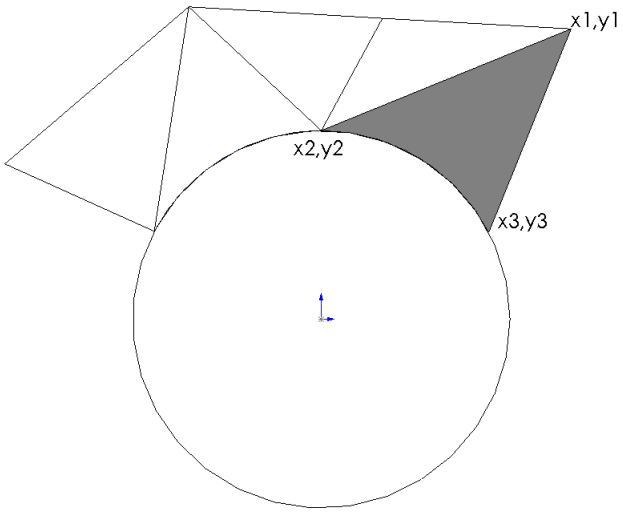


Figure 4.3: Boundary element with curved face for high order integration scheme

ated using finite differences as

$$\begin{aligned} \frac{\partial L}{\partial x_{2,3}} &= \frac{L(x_{2,3}+\epsilon) - L(x_{2,3}-\epsilon)}{2\epsilon} \\ \frac{\partial L}{\partial y_{2,3}} &= \frac{L(y_{2,3}+\epsilon) - L(y_{2,3}-\epsilon)}{2\epsilon} \end{aligned} \tag{4.24}$$

For a second order scheme, only one Gauss integration point exists at the middle of the face with $w_i = L$, therefore

$$\frac{dw_i}{dM} = \frac{dL}{dM}$$

For the fourth order scheme, two Gauss integration points are required for flux integration. Figure 4.2.2 shows a schematic drawing of a curved edge with two Gauss points on it. Table 4.1 shows their integration weights and their parametric locations k_i on the face starting from the point x_2, y_2.

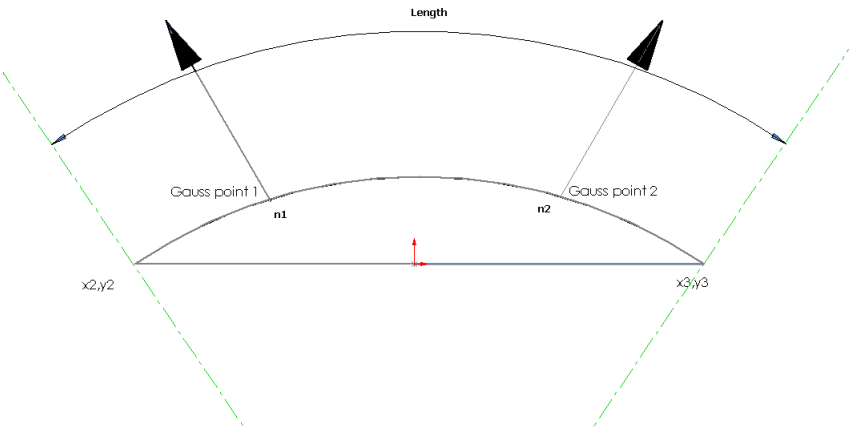


Figure 4.4: Schematic drawing of a curved face with two Gauss points used

	Point 1	Point 2
w_i	$L/2$	$L/2$
k_i	$(\frac{1}{2} - \frac{1}{\sqrt{12}})L$	$(\frac{1}{2} + \frac{1}{\sqrt{12}})L$

Table 4.1: Two point Gauss quadrature rule

4.2.3 Face normal mesh dependency

The unit normal vector of face κ that points outward from element Ω_i (shown in Fig. 4.2) can be found using the three vertices of the element as

$$\hat{n} = \begin{bmatrix} n_x \\ n_y \end{bmatrix} = \frac{(\vec{c}_3 - \beta \hat{r}_{23})}{\|\vec{c}_3 - \beta \hat{r}_{23}\|} \tag{4.25}$$

where

$$\begin{aligned}
\vec{c}_3 &= \begin{bmatrix} 2/3\, x_3 - (1/3\, x_1 + 1/3\, x_2) \\ 2/3\, y_3 - (1/3\, y_1 + 1/3\, y_2) \end{bmatrix} \\
\hat{r}_{23} &= \begin{bmatrix} \frac{x_3 - x_2}{\sqrt{(|-x_3+x_2|)^2 + (|-y_3+y_2|)^2}} \\ \frac{y_3 - y_2}{\sqrt{(|-x_3+x_2|)^2 + (|-y_3+y_2|)^2}} \end{bmatrix} \\
\beta &= \frac{(x_3 - x_2)\,(2/3\, x_3 - (1/3\, x_1 + 1/3\, x_2))}{\sqrt{(|-x_3 + x_2|)^2 + (|-y_3 + y_2|)^2}} \\
&\quad + \frac{(y_3 - y_2)\,(2/3\, y_3 - (1/3\, y_1 + 1/3\, y_2))}{\sqrt{(|-x_3 + x_2|)^2 + (|-y_3 + y_2|)^2}}
\end{aligned}$$

The mesh dependency of the unit normal can be written as

$$\begin{aligned}
\frac{d\hat{n}}{dM} &= \frac{\partial \hat{n}}{\partial x_1}\frac{\partial x_1}{\partial M} + \frac{\partial \hat{n}}{\partial x_2}\frac{\partial x_2}{\partial M} + \frac{\partial \hat{n}}{\partial x_3}\frac{\partial x_3}{\partial M} \\
&\quad + \frac{\partial \hat{n}}{\partial y_1}\frac{\partial y_1}{\partial M} + \frac{\partial \hat{n}}{\partial y_2}\frac{\partial y_2}{\partial M} + \frac{\partial \hat{n}}{\partial y_3}\frac{\partial y_3}{\partial M}
\end{aligned} \tag{4.26}$$

where

$$\begin{aligned}
\frac{\partial \hat{n}}{\partial x_i} &= \frac{\alpha_1 - \alpha_2}{\|\vec{c}_3 - \beta \hat{r}_{23}\|^2} \\
\alpha_1 &= \|\vec{c}_3 - \beta \hat{r}_{23}\| \cdot \frac{\partial}{\partial x_i}(\vec{c}_3 - \beta \hat{r}_{23}) \\
\alpha_2 &= (\vec{c}_3 - \beta \hat{r}_{23}) \cdot \frac{\partial}{\partial x_i}\|\vec{c}_3 - \beta \hat{r}_{23}\|
\end{aligned}$$

$$
\begin{aligned}
\frac{\partial \hat{n}}{\partial y_i} &= \frac{\alpha_3 - \alpha_4}{\left\| \vec{c}_3 - \beta \hat{r}_{23} \right\|^2} \\
\alpha_3 &= \left\| \vec{c}_3 - \beta \hat{r}_{23} \right\| \cdot \frac{\partial}{\partial y_i} \left(\vec{c}_3 - \beta \hat{r}_{23} \right) \\
\alpha_4 &= \left(\vec{c}_3 - \beta \hat{r}_{23} \right) \cdot \frac{\partial}{\partial y_i} \left\| \vec{c}_3 - \beta \hat{r}_{23} \right\| \\
& \quad i = 1, 2, 3
\end{aligned}
$$

The terms in the last equations can be found using automatic differentiation of the unit normal expression.

4.3 Face Flow Properties Dependency on The Mesh

The evaluation of the $\partial U_{\text{face}} / \partial M$ term in the mesh sensitivity of the residual, using Eq. 4.9, depends on the details of the CFD solver. The CFD solver used in this research is the Advanced Numerical Simulation Library (ANSLib) which is a multi-physics finite volume solver capable of conducting CFD simulations up to fourth order accuracy; this solver has been built by Ollivier-Gooch and co-workers [61, 56, 54]. In ANSLib, the flow properties are assumed to change within each control volume according to a linear polynomial for second order simulations, or a cubic polynomial for fourth order simulations. These polynomials are found using a least-squares reconstruction; the least-squares system depends on the control volume properties like moments of area which eventually depend on mesh point locations. In the next two subsections, the reconstruction and mesh dependency of flow properties at faces will be presented.

4.3.1 Face flow properties reconstruction

Solution reconstruction is the key to determine solver accuracy. The solution is assumed to vary linearly within the control volume for second order accuracy; for higher order methods, flow properties are assumed to vary according to higher order polynomial; variation according to cubic polynomial is fourth order accurate. The flow solution reconstruction method presented in this subsection follows the method described in Ollivier-Gooch and Van Altena [68] and Ollivier-Gooch et al [69].

For second order solution reconstruction, the primitive variables $U = \begin{bmatrix} \rho & u & v & p \end{bmatrix}^T$ are reconstructed at the face Gauss points as

$$U_2^R(x,y) = U_{ref} + (x - x_{ref}) \left.\frac{\partial U}{\partial x}\right|_{ref} + (y - y_{ref}) \left.\frac{\partial U}{\partial y}\right|_{ref} \tag{4.27}$$

For fourth order solution reconstruction, the reconstruction polynomial takes the form of a third degree Taylor series expansion in two variables:

$$\begin{aligned} U_4^R(x,y) = U_2^R(x,y) &+ \frac{(x - x_{ref})^2}{2} \left.\frac{\partial^2 U}{\partial x^2}\right|_{ref} \\ &+ (x - x_{ref}) \,.\, (y - y_{ref}) \left.\frac{\partial^2 U}{\partial x \partial y}\right|_{ref} + \frac{(y - y_{ref})^2}{2} \left.\frac{\partial^2 U}{\partial y^2}\right|_{ref} \\ &+ \frac{(x - x_{ref})^3}{6} \left.\frac{\partial^3 U}{\partial x^3}\right|_{ref} + \frac{(x - x_{ref})^2 (y - y_{ref})}{2} \left.\frac{\partial^3 U}{\partial x^2 \partial y}\right|_{ref} \\ &+ \frac{(x - x_{ref}) (y - y_{ref})^2}{2} \left.\frac{\partial^3 U}{\partial x \partial y^2}\right|_{ref} + \frac{(y - y_{ref})^3}{6} \left.\frac{\partial^3 U}{\partial y^3}\right|_{ref} \end{aligned} \tag{4.28}$$

The reference point $\vec{x}_{ref}$ is chosen to be the triangle center for cell-centered finite volume scheme and the vertex location for vertex-centered scheme.

A constrained least-squares system is constructed to find the reconstruction polynomial coefficient. The hard constraint is that the reconstruction polynomial should preserve the computed control volume average solution. The rest of the equations — which are satisfied only in a least-squares sense — are obtained by requiring that the average of the reconstruction polynomial approximates the control volume average solution for neighboring control volumes. Neighbors are chosen based on their topological distance from the element, and entire layers of neighbors is added until the number of neighbor elements in the stencil equals or exceeds 4 for the second order scheme and 16 for the fourth order scheme. Figure 4.5 shows a schematic drawing of an element k and its neighbor layers.

The mean constraint equation is obtained by requiring conservation of the control

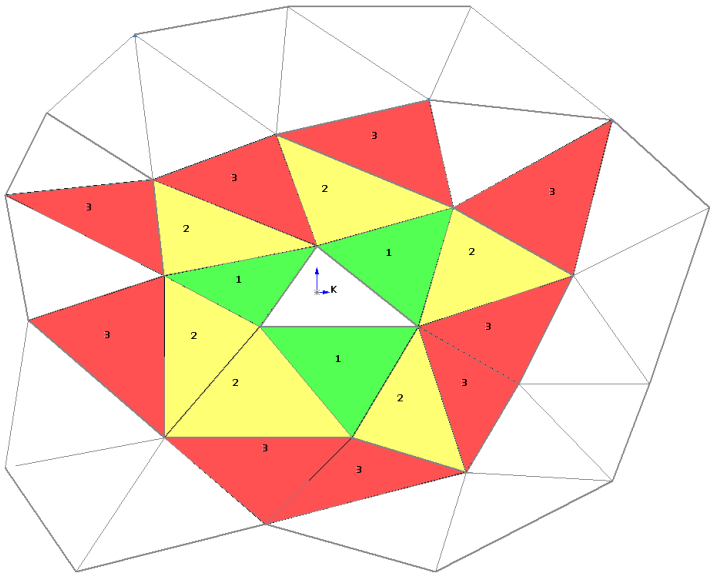


Figure 4.5: Schematic drawing of first, second and third neighbor layers of triangular element k.

volume average by the reconstruction polynomial and can be written as

$$\begin{aligned}\overline{U}_k &= \frac{1}{A_k}\int_{A_k} U_k^R dA \equiv U_{k\,ref} + \overline{x}_k \left.\frac{\partial U}{\partial x}\right|_{k\,ref} + \overline{y}_k \left.\frac{\partial U}{\partial y}\right|_{k\,ref} + \\ &\quad \frac{\overline{x^2}_k}{2}\left.\frac{\partial^2 U}{\partial x^2}\right|_{k\,ref} + \overline{xy}_k \left.\frac{\partial^2 U}{\partial x \partial y}\right|_{k\,ref} + \frac{\overline{y^2}_k}{2}\left.\frac{\partial^2 U}{\partial y^2}\right|_{k\,ref} + \ldots\end{aligned} \tag{4.29}$$

where

$$\overline{x^n y^m}_k = \frac{1}{A_k}\iint_{A_k} (x - x_{k\,ref})^n (y - y_{k\,ref})^m \, dA$$

Matching the control volume average in a neighbor j would require that

$$\begin{aligned}\overline{U}_j &= \frac{1}{A_j}\int_{A_j} U_k^R dA \equiv U_{k\,ref} + \frac{1}{A_j}\int_{A_j}(x - x_{k\,ref})\left.\frac{\partial U}{\partial x}\right|_{ref} dA + \\ &\frac{1}{A_j}\int_{A_j}(y - y_{k\,ref})\left.\frac{\partial U}{\partial y}\right|_{ref} dA + \frac{1}{A_j}\int_{A_j}\frac{(x - x_{k\,ref})^2}{2}\left.\frac{\partial^2 U}{\partial x^2}\right|_{ref} dA + \\ &\frac{1}{A_j}\int_{A_j}(x - x_{k\,ref})(y - y_{k\,ref})\left.\frac{\partial^2 U}{\partial x \partial y}\right|_{ref} dA + \\ &\frac{1}{A_j}\int_{A_j}\frac{(y - y_{k\,ref})^2}{2}\left.\frac{\partial^2 U}{\partial y^2}\right|_{ref} dA + ...\end{aligned} \tag{4.30}$$

Using

$$(x - x_{k\,ref}) = (x - x_{j\,ref}) - (x_{k\,ref} - x_{j\,ref})$$

and

$$(y - y_{k\,ref}) = (y - y_{j\,ref}) - (y_{k\,ref} - y_{j\,ref}),$$

we get

$$\begin{aligned}\overline{U}_j &= \frac{1}{A_j}\int_{A_j} U_k^R dA \equiv U_{k\,ref} + \widehat{x}_{k,j}\left.\frac{\partial U}{\partial x}\right|_{k\,ref} + \widehat{y}_{k,j}\left.\frac{\partial U}{\partial y}\right|_{k\,ref} + \\ &\frac{\widehat{x^2}_k}{2}\left.\frac{\partial^2 U}{\partial x^2}\right|_{k\,ref} + \widehat{xy}_{k,j}\left.\frac{\partial^2 U}{\partial x \partial y}\right|_{k\,ref} + \frac{\widehat{y^2}_{k,j}}{2}\left.\frac{\partial^2 U}{\partial y^2}\right|_{k\,ref} + ...\end{aligned} \tag{4.31}$$

where

$$\begin{aligned}\widehat{x^n y^m}_{k,j} &= \frac{1}{A_j}\int_{A_j}((x - x_{j\,ref}) - (x_{k\,ref} - x_{j\,ref}))^n \cdot \\ &\quad ((y - y_{j\,ref}) - (y_{k\,ref} - y_{j\,ref}))^m dA \\ &= \sum_{r=0}^{m}\sum_{q=0}^{n}\frac{m!}{r!\,(m-r)\,!}\frac{n!}{q!\,(n-q)\,!}(x_{j\,ref} - x_{k\,ref})^q \cdot \\ &\quad (y_{j\,ref} - y_{k\,ref})^r\,\overline{x_{j\,ref}^{n-q}\,y_{j\,ref}^{m-r}}\end{aligned}$$

for simplicity, the term $x, y_{k\,ref}$ will be written as x, y_k, and $x, y_{j\,ref}$ will be replaced

with x, y_j. The resulting constrained least-squares system can be written as

$$\left[\begin{array}{cccccc|c} 1 & \overline{x}_k & \overline{y}_k & \overline{x_k^2} & \overline{xy_k} & \overline{y_k^2} & \cdots \\ \hline 1 & \widehat{x}_{k,1} & \widehat{y}_{k,1} & \widehat{x^2}_{k,1} & \widehat{xy}_{k,1} & \widehat{y^2}_{k,1} & \cdots \\ 1 & \widehat{x}_{k,2} & \widehat{y}_{k,2} & \widehat{x^2}_{k,2} & \widehat{xy}_{k,2} & \widehat{y^2}_{k,2} & \cdots \\ 1 & \widehat{x}_{k,3} & \widehat{y}_{k,3} & \widehat{x^2}_{k,3} & \widehat{xy}_{k,3} & \widehat{y^2}_{k,3} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \\ 1 & \widehat{x}_{k,N} & \widehat{y}_{k,N} & \widehat{x^2}_{k,N} & \widehat{xy}_{k,N} & \widehat{y^2}_{k,N} & \cdots\cdots \end{array}\right] \left(\begin{array}{c} U \\ \frac{\partial U}{\partial x} \\ \frac{\partial U}{\partial y} \\ \frac{1}{2}\frac{\partial^2 U}{\partial x^2} \\ \frac{\partial^2 U}{\partial x \partial y} \\ \frac{1}{2}\frac{\partial^2 U}{\partial y^2} \\ \vdots \end{array}\right)_k = \left(\begin{array}{c} \overline{U}_k \\ \overline{U}_1 \\ \overline{U}_2 \\ \overline{U}_3 \\ \vdots \\ \overline{U}_N \end{array}\right) \tag{4.32}$$

This constrained least squares system can be rewritten as

$$\begin{aligned} &\min \left\| [A_{Rec}] \left(\widetilde{P}_{Rec}\right) - \left(\overline{U}_{Rec} - \overline{U}_k\right) \right\| \\ &\quad \textit{satisfying } [B_{Rec}] \left(P_{Rec}\right) = \left(\overline{U}_k\right) \end{aligned}$$

where

$$[A_{Rec}] = \left[\begin{array}{cccccc} \widehat{x}_{k,1} - \overline{x}_k, & \widehat{y}_{k,1} - \overline{y}_k, & \widehat{x^2}_{k,1} - \overline{x_k^2}, & \widehat{xy}_{k,1} - \overline{xy_k}, & \widehat{y^2}_{k,1} - \overline{y_k^2}, & \cdots \\ \widehat{x}_{k,2} - \overline{x}_k, & \widehat{y}_{k,2} - \overline{y}_k, & \widehat{x^2}_{k,2} - \overline{x_k^2}, & \widehat{xy}_{k,2} - \overline{xy_k}, & \widehat{y^2}_{k,2} - \overline{y_k^2}, & \cdots \\ \widehat{x}_{k,3} - \overline{x}_k, & \widehat{y}_{k,3} - \overline{y}_k, & \widehat{x^2}_{k,3} - \overline{x_k^2}, & \widehat{xy}_{k,3} - \overline{xy_k}, & \widehat{y^2}_{k,3} - \overline{y_k^2}, & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \\ \widehat{x}_{k,N} - \overline{x}_k, & \widehat{y}_{k,N} - \overline{y}_k, & \widehat{x^2}_{k,N} - \overline{x_k^2}, & \widehat{xy}_{k,N} - \overline{xy_k}, & \widehat{y^2}_{k,N} - \overline{y_k^2}, & \cdots\cdots \end{array}\right]$$

$$[B_{Rec}] = \left[\begin{array}{cccccc} 1 & \overline{x}_k & \overline{y}_k & \overline{x_k^2} & \overline{xy_k} & \overline{y_k^2} & \cdots \end{array}\right]$$

and

$$P_{Rec} = \left(\begin{array}{c} U \\ \frac{\partial U}{\partial x} \\ \frac{\partial U}{\partial y} \\ \frac{1}{2}\frac{\partial^2 U}{\partial x^2} \\ \frac{\partial^2 U}{\partial x \partial y} \\ \frac{1}{2}\frac{\partial^2 U}{\partial y^2} \\ \vdots \end{array}\right)_k, \quad \widetilde{P}_{Rec} = \left(\begin{array}{c} \frac{\partial U}{\partial x} \\ \frac{\partial U}{\partial y} \\ \frac{1}{2}\frac{\partial^2 U}{\partial x^2} \\ \frac{\partial^2 U}{\partial x \partial y} \\ \frac{1}{2}\frac{\partial^2 U}{\partial y^2} \\ \vdots \end{array}\right)_k, \quad \left(\overline{U}_{Rec} - \overline{U}_k\right) = \left(\begin{array}{c} \overline{U}_1 - \overline{U}_k \\ \overline{U}_2 - \overline{U}_k \\ \overline{U}_3 - \overline{U}_k \\ \vdots \\ \overline{U}_N - \overline{U}_k \end{array}\right)$$

The least-squares system is solved to obtain the reconstruction polynomial coefficients using singular value decomposition method to find the pseudo inverse of the least-squares coefficient matrix as follows

$$\begin{aligned} A_{Rec} &= U\Sigma V^T \\ A^{\dagger}_{Rec} &= V\Sigma^{\dagger}U^T \end{aligned}$$

$$\begin{pmatrix} \frac{\partial U}{\partial x} \\ \frac{\partial U}{\partial y} \\ \frac{1}{2}\frac{\partial^2 U}{\partial x^2} \\ \frac{\partial^2 U}{\partial x \partial y} \\ \frac{1}{2}\frac{\partial^2 U}{\partial y^2} \\ \vdots \end{pmatrix}_k = \begin{bmatrix} \widehat{x}_{k,1} - \overline{x}_k, & \widehat{y}_{k,1} - \overline{y}_k, & \widehat{x^2}_{k,1} - \overline{x_k^2}, & \widehat{xy}_{k,1} - \overline{xy_k}, & \widehat{y^2}_{k,1} - \overline{y_k^2}, & \cdots \\ \widehat{x}_{k,2} - \overline{x}_k, & \widehat{y}_{k,2} - \overline{y}_k, & \widehat{x^2}_{k,2} - \overline{x_k^2}, & \widehat{xy}_{k,2} - \overline{xy_k}, & \widehat{y^2}_{k,2} - \overline{y_k^2}, & \cdots \\ \widehat{x}_{k,3} - \overline{x}_k, & \widehat{y}_{k,3} - \overline{y}_k, & \widehat{x^2}_{k,3} - \overline{x_k^2}, & \widehat{xy}_{k,3} - \overline{xy_k}, & \widehat{y^2}_{k,3} - \overline{y_k^2}, & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \\ \widehat{x}_{k,N} - \overline{x}_k, & \widehat{y}_{k,N} - \overline{y}_k, & \widehat{x^2}_{k,N} - \overline{x_k^2}, & \widehat{xy}_{k,N} - \overline{xy_k}, & \widehat{y^2}_{k,N} - \overline{y_k^2}, & \cdots\cdots \end{bmatrix}^{\dagger} \cdot \begin{pmatrix} \overline{U}_1 - \overline{U}_k \\ \overline{U}_2 - \overline{U}_k \\ \overline{U}_3 - \overline{U}_k \\ \vdots \\ \overline{U}_N - \overline{U}_k \end{pmatrix} \tag{4.33}$$

$$U_k = \overline{U}_k - \begin{bmatrix} \overline{x}_k & \overline{y}_k & \overline{x_k^2} & \overline{xy_k} & \overline{y_k^2} & \cdots \end{bmatrix} \cdot \begin{pmatrix} \frac{\partial U}{\partial x} \\ \frac{\partial U}{\partial y} \\ \frac{1}{2}\frac{\partial^2 U}{\partial x^2} \\ \frac{\partial^2 U}{\partial x \partial y} \\ \frac{1}{2}\frac{\partial^2 U}{\partial y^2} \\ \vdots \end{pmatrix} \tag{4.34}$$

The unlimited face flow properties can be found at face Gauss points $\left(x_{f_i}, y_{f_f}\right)$

as follows

$$
\begin{aligned}
U_{f_i} &= U_k + (x_{f_i} - x_k) \left.\frac{\partial U}{\partial x}\right|_k + (y_{f_i} - y_k) \left.\frac{\partial U}{\partial y}\right|_k + \\
&\quad \frac{(x_{f_i} - x_k)^2}{2} \left.\frac{\partial^2 U}{\partial x^2}\right|_k + (x_{f_i} - x_k)(y_{f_i} - y_k) \left.\frac{\partial^2 U}{\partial x \partial y}\right|_k + \\
&\quad \frac{(y_{f_i} - y_k)^2}{2} \left.\frac{\partial^2 U}{\partial y^2}\right|_k + \ldots
\end{aligned} \tag{4.35}
$$

To ensure monotonicity and solution convergence, a limiter is applied in order not to create a new extremum at the control volume face; the limited flow properties at the face can be written as

$$
\begin{aligned}
U_{f_i} &= U_k + \phi_k \left((x_{f_i} - x_k) \left.\frac{\partial U}{\partial x}\right|_k + (y_{f_i} - y_k) \left.\frac{\partial U}{\partial y}\right|_k + \right. \\
&\quad \frac{(x_{f_i} - x_k)^2}{2} \left.\frac{\partial^2 U}{\partial x^2}\right|_k + (x_{f_i} - x_k)(y_{f_i} - y_k) \left.\frac{\partial^2 U}{\partial x \partial y}\right|_k + \\
&\quad \left. \frac{(y_{f_i} - y_k)^2}{2} \left.\frac{\partial^2 U}{\partial y^2}\right|_k + \ldots \right)
\end{aligned} \tag{4.36}
$$

where ϕ_k is the limiter value in the control volume k. ANSLib uses two limiter functions to calculate ϕ: Venkatakrishnan's limiter [84] and a higher-order limiter [56] designed not to degrade the accuracy of high-order schemes for smooth flows. We will derive the face properties-mesh dependency for the limited case; the unlimited case can be obtained by simplification of the limited case by setting $\phi_k = 1$.

4.3.2 Mesh dependence of the face flow property reconstruction

The term $\partial U_{\text{face}}/\partial M$ for the limited case can be obtained using direct differentiation of Eq. 4.36

$$
\begin{aligned}
\frac{\partial U_{f_i}}{\partial M} \;=\; & \frac{\partial U_k}{\partial M} + \frac{\partial \phi_k}{\partial M}\left(\left(x_{f_i}-x_k\right) \left.\frac{\partial U}{\partial x}\right|_k + \left(y_{f_i}-y_k\right) \left.\frac{\partial U}{\partial y}\right|_k + \right. \\
& \frac{\left(x_{f_i}-x_k\right)^2}{2} \left.\frac{\partial^2 U}{\partial x^2}\right|_k + \left(x_{f_i}-x_k\right)\left(y_{f_i}-y_k\right) \left.\frac{\partial^2 U}{\partial x \partial y}\right|_k + \\
& \left. \frac{\left(y_{f_i}-y_k\right)^2}{2} \left.\frac{\partial^2 U}{\partial y^2}\right|_k + \ldots \right) + \\
& \phi_k \left(\frac{\partial \left(x_{f_i}-x_k\right)}{\partial M} \left.\frac{\partial U}{\partial x}\right|_k + \frac{\partial \left(y_{f_i}-y_k\right)}{\partial M} \left.\frac{\partial U}{\partial y}\right|_k + \right. \\
& \frac{\partial \frac{\left(x_{f_i}-x_k\right)^2}{2}}{\partial M} \left.\frac{\partial^2 U}{\partial x^2}\right|_k + \frac{\partial \left(x_{f_i}-x_k\right)\left(y_{f_i}-y_k\right)}{\partial M} \left.\frac{\partial^2 U}{\partial x \partial y}\right|_k + \\
& \left. \frac{\partial \frac{\left(y_{f_i}-y_k\right)^2}{2}}{\partial M} \left.\frac{\partial^2 U}{\partial y^2}\right|_k + \ldots \right) + \\
& \phi_k \left(\left(x_{f_i}-x_k\right) \frac{\partial \left.\frac{\partial U}{\partial x}\right|_k}{\partial M} + \left(y_{f_i}-y_k\right) \frac{\partial \left.\frac{\partial U}{\partial y}\right|_k}{\partial M} + \right. \\
& \frac{\left(x_{f_i}-x_k\right)^2}{2} \frac{\partial \left.\frac{\partial^2 U}{\partial x^2}\right|_k}{\partial M} + \left(x_{f_i}-x_k\right)\left(y_{f_i}-y_k\right) \frac{\partial \left.\frac{\partial^2 U}{\partial x \partial y}\right|_k}{\partial M} + \\
& \left. \frac{\left(y_{f_i}-y_k\right)^2}{2} \frac{\partial \left.\frac{\partial^2 U}{\partial y^2}\right|_k}{\partial M} + \ldots \right)
\end{aligned}
\tag{4.37}
$$

The unlimited face flow property-mesh dependency can be obtained by setting $\phi_k = 1$ and $\partial \phi_k / \partial M = 0$ in Eq. 4.37.

In Eq. 4.37, the term $\partial \phi_k / \partial M$ can be obtained by differentiating the limiter

expression, and the terms $\partial \left.\frac{\partial^n U}{\partial x^m \partial y^{n-m}}\right|_k / \partial M$ can be evaluated as follows:

$$\begin{aligned}
[A_{Rec}] \left\{ \widetilde{P}_{Rec} \right\} &= \left\{ \overline{U}_{Rec} \right\} \\
\frac{\partial [A_{Rec}]}{\partial M} \left\{ \widetilde{P}_{Rec} \right\} + [A_{Rec}] \frac{\partial \left\{ \widetilde{P}_{Rec} \right\}}{\partial M} &= \frac{\partial \left\{ \overline{U}_{Rec} \right\}}{\partial M} = \{0\} \\
\therefore \frac{\partial \left\{ \widetilde{P}_{Rec} \right\}}{\partial M} &= -[A_{Rec}]^{\dagger} \left\{ \frac{\partial [A_{Rec}]}{\partial M} \left\{ \widetilde{P}_{Rec} \right\} \right\} \\
\frac{\partial \left\{ \widetilde{P}_{Rec} \right\}}{\partial M} &= -[A_{Rec}]^{\dagger} \frac{\partial [A_{Rec}]}{\partial M} [A_{Rec}]^{\dagger} \left\{ \overline{U}_{Rec} \right\}
\end{aligned} \tag{4.38}$$

and

$$\begin{aligned}
\because U_k =& \overline{U}_k - \left[\begin{array}{ccccccc} \overline{x}_k & \overline{y}_k & \overline{x_k^2} & \overline{xy_k} & \overline{y_k^2} & \cdots \end{array} \right] \cdot \left\{ \widetilde{P}_{Rec} \right\} \\
\therefore \frac{\partial U_k}{\partial M} =& -\frac{\partial}{\partial M} \left[\begin{array}{ccccccc} \overline{x}_k & \overline{y}_k & \overline{x_k^2} & \overline{xy_k} & \overline{y_k^2} & \cdots \end{array} \right] \cdot \left\{ \widetilde{P}_{Rec} \right\} - \\
& \left[\begin{array}{ccccccc} \overline{x}_k & \overline{y}_k & \overline{x_k^2} & \overline{xy_k} & \overline{y_k^2} & \cdots \end{array} \right] \cdot \frac{\partial \left\{ \widetilde{P}_{Rec} \right\}}{\partial M}
\end{aligned} \tag{4.39}$$

where

$$\frac{\partial [A_{Rec}]}{\partial M} = \left[\begin{array}{cccccc}
\frac{\partial \widehat{x}_{k,1}}{\partial M} & \frac{\partial \widehat{y}_{k,1}}{\partial M} & \frac{\partial \widehat{x^2}_{k,1}}{\partial M} & \frac{\partial \widehat{xy}_{k,1}}{\partial M} & \frac{\partial \widehat{y^2}_{k,1}}{\partial M} & \cdots \\
\frac{\partial \widehat{x}_{k,2}}{\partial M} & \frac{\partial \widehat{y}_{k,2}}{\partial M} & \frac{\partial \widehat{x^2}_{k,2}}{\partial M} & \frac{\partial \widehat{xy}_{k,2}}{\partial M} & \frac{\partial \widehat{y^2}_{k,2}}{\partial M} & \cdots \\
\frac{\partial \widehat{x}_{k,3}}{\partial M} & \frac{\partial \widehat{y}_{k,3}}{\partial M} & \frac{\partial \widehat{x^2}_{k,3}}{\partial M} & \frac{\partial \widehat{xy}_{k,3}}{\partial M} & \frac{\partial \widehat{y^2}_{k,3}}{\partial M} & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots \\
\frac{\partial \widehat{x}_{k,N}}{\partial M} & \frac{\partial \widehat{y}_{k,N}}{\partial M} & \frac{\partial \widehat{x^2}_{k,N}}{\partial M} & \frac{\partial \widehat{xy}_{k,N}}{\partial M} & \frac{\partial \widehat{y^2}_{k,N}}{\partial M} & \cdots\cdots
\end{array} \right]$$

$$\frac{\partial \left\{ \widetilde{P}_{Rec} \right\}}{\partial M} = \frac{\partial}{\partial M} \left(\begin{array}{c} \frac{\partial U}{\partial x} \\ \frac{\partial U}{\partial y} \\ \frac{1}{2} \frac{\partial^2 U}{\partial x^2} \\ \frac{\partial^2 U}{\partial x \partial y} \\ \frac{1}{2} \frac{\partial^2 U}{\partial y^2} \\ \vdots \end{array} \right)$$

and

$$\frac{\partial}{\partial M}\left\{\left(x_{f_i}-x_k\right)^m\left(y_{f_i}-y_k\right)^n\right\} = m\left(x_{f_i}-x_k\right)^{m-1}\left(y_{f_i}-y_k\right)^n\left(\frac{\partial x_{f_i}}{\partial M}-\frac{\partial x_k}{\partial M}\right)+ n\left(x_{f_i}-x_k\right)^m\left(y_{f_i}-y_k\right)^{n-1}\left(\frac{\partial y_{f_i}}{\partial M}-\frac{\partial y_k}{\partial M}\right)$$

$$\begin{aligned}\frac{\partial\widehat{x^n y^m}_{k,j}}{\partial M} = & \sum_{r=0}^{m}\sum_{q=0}^{n}\left\{\frac{m!}{r!\,(m-r)\,!}\frac{n!}{q!\,(n-q)\,!}\,q\left(x_{j\,ref}-x_{k\,ref}\right)^{q-1}\cdot\left(y_{j\,ref}-y_{k\,ref}\right)\cdot\right. \\ & \overline{x_{j\,ref}^{n-q}\,y_{j\,ref}^{m-r}}\cdot\left(\frac{\partial x_{j\,ref}}{\partial M}-\frac{\partial x_{k\,ref}}{\partial M}\right)+ \\ & \frac{m!}{r!\,(m-r)\,!}\frac{n!}{q!\,(n-q)\,!}\,r\left(x_{j\,ref}-x_{k\,ref}\right)^{q}\left(y_{j\,ref}-y_{k\,ref}\right)^{r-1}\cdot \\ & \left.\overline{x_{j\,ref}^{n-q}\,y_{j\,ref}^{m-r}}\cdot\left(\frac{\partial y_{j\,ref}}{\partial M}-\frac{\partial y_{k\,ref}}{\partial M}\right)\right\}\end{aligned}$$

Chapter 5

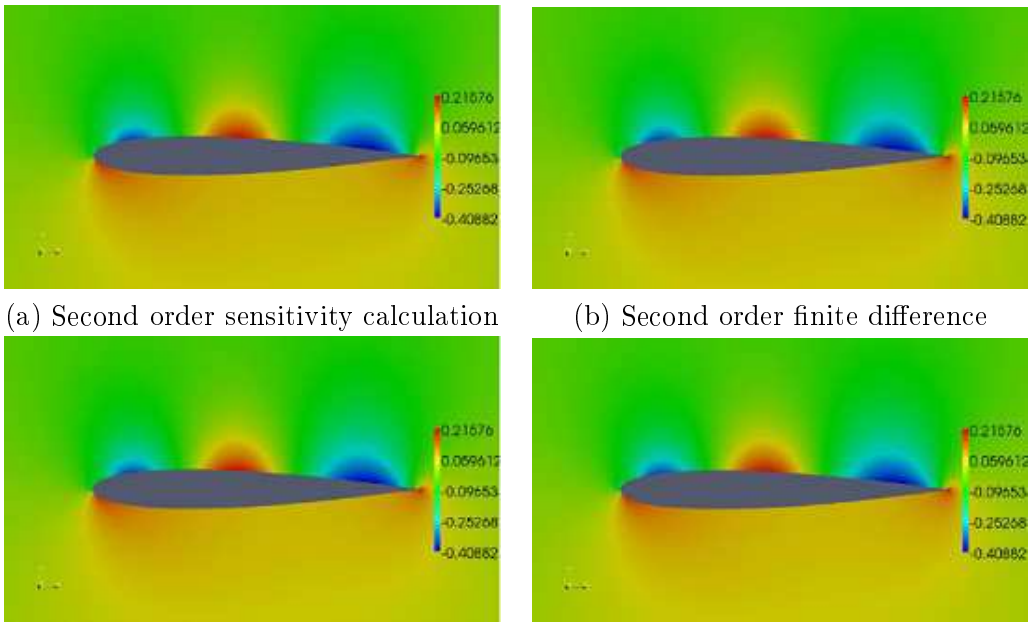
GRADIENT VALIDATION

In this section, the comparison of the gradient accuracy evaluated using second and fourth order schemes is carried out. Both second and fourth order gradients are evaluated using the adjoint approach and compared to their corresponding finite difference gradient. We present three test cases: subsonic, non-limited transonic and limited transonic test cases. In all test cases we use 18 design points to parametrize the airfoil geometry.

5.1 Subsonic Test Case

In this test case, the evaluation of the lift coefficient gradient with respect to the airfoil geometric design control points is presented for both second and fourth order schemes, including a comparison with a finite difference gradient calculated using the same order of accuracy in the flow solver. The airfoil used in this test case is NACA 0012 at subsonic conditions M = 0.5 and $\alpha = 2^o$. Figure 5.1 shows a representative comparison between the field of pressure sensitivity with respect to one of the geometry design points computed using finite differences and the solution sensitivity calculation of Eq. 4.6 for both second and fourth order accurate computations. Agreement is excellent in all cases; this is also true for the other design control points. The excellent matching of the pressure sensitivity when comparing the second order and the fourth order results indicates that the two schemes will give similar gradient vectors and similar optimization descent directions for subsonic flow optimization.

Table 5.1 shows quantitatively an excellent matching between the objective function gradient magnitude (less than half a percent difference) and direction (less than a half a degree difference) when comparing finite differences and the adjoint approach of Eq. 4.8 for both second and fourth order schemes. Figure 5.1 shows good



(a) Second order sensitivity calculation

(b) Second order finite difference

(c) Fourth order sensitivity calculation

(d) Fourth order finite difference

Figure 5.1: The pressure sensitivity with respect to one of the design control points computed for subsonic flow over NACA 0012, comparing the sensitivity calculation of Eq. 4.6 with finite difference results.

agreement between the objective function gradient components $\partial C_L/\partial y_i$, where y_i are the y locations of the design control points, computed using adjoint and finite difference approaches for both second and fourth order computations. The maximum error, normalized by the gradient magnitude, is only 0.005 which gives a high level of accuracy for the computed gradient. Taken together, these results imply that the order of discretization error has little effect on the computed gradient vector for subsonic flow.

5.2 Transonic Test Case with No Limiter

In this test case, the sensitivity analysis is carried out for NACA 0012 airfoil with M = 0.8 and $\alpha = 2°$. The drag coefficient gradient is evaluated without using limiters in the CFD simulation, so overshoot/undershoot at the shock location is expected. Figure 5.3 shows good agreement of the pressure sensitivity computed using Eq. 4.6

	Second order		Fourth order	
	Adjoint	FD	Adjoint	FD
Gradient vector magnitude	18.605	18.561	18.575	18.655
Angle with second order adjoint	0°	0.247°	1.117°	1.330°
Angle with fourth order adjoint	1.117°	0.917°	0°	0.249°

Table 5.1: The magnitude of second and fourth order C_L gradients and angles between the evaluated gradients for NACA 0012 in subsonic flow.

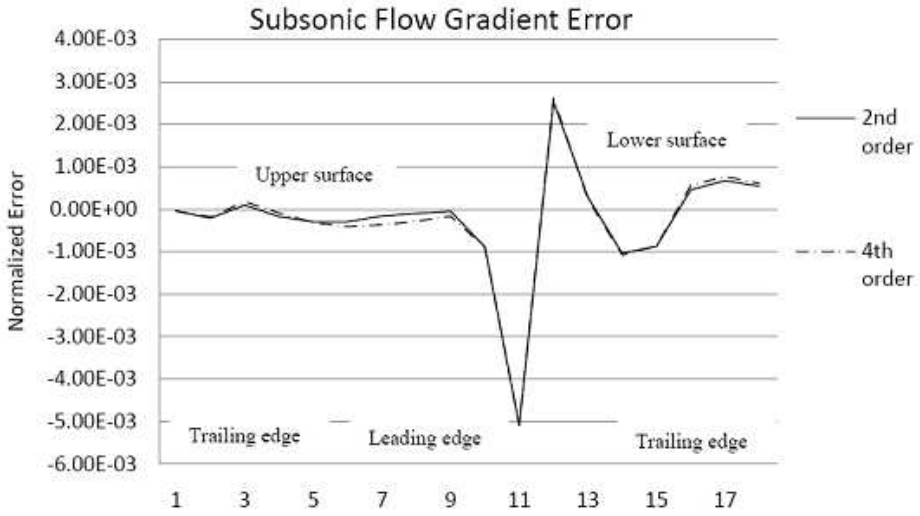
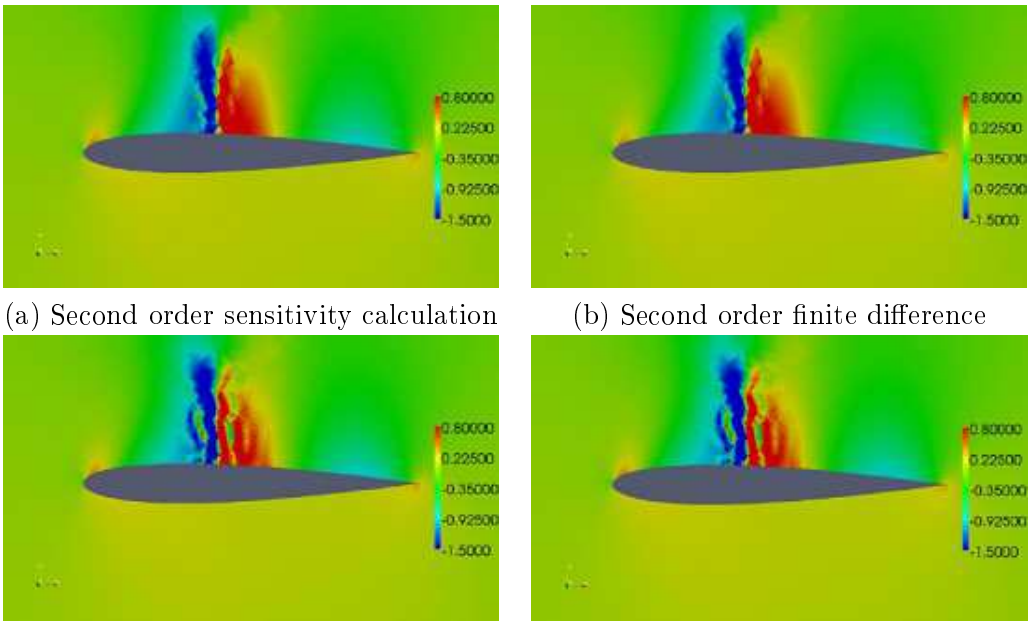


Figure 5.2: C_L gradient error in second and fourth order schemes with respect to the design points normalized by gradient magnitude.



(a) Second order sensitivity calculation (b) Second order finite difference

(c) Fourth order sensitivity calculation (d) Fourth order finite difference

Figure 5.3: The pressure sensitivity with respect to one of the design control points computed for an unlimited transonic flow around NACA 0012, comparing the sensitivity calculation of Eq. 4.6 with finite difference results.

and finite difference; the figure also shows that for transonic flows, the pressure sensitivity computed using second order and fourth order accurate adjoint scheme are different, especially near the shock location.

Table 5.2 shows again the excellent matching between the computed adjoint and finite difference gradient with a direction difference less than a degree and nearly identical magnitude. Figure 5.4 shows that for unlimited transonic flow, both second and fourth order adjoint gradients are an excellent match to the corresponding finite difference gradient, with the second order schemes matching more closely than the fourth order scheme. Comparison of the second and fourth order gradient vectors shows little difference between them (about 4% in magnitude and 2° in direction). Again, we expect that the second and fourth order schemes will give similar optimization search directions.

	Second order		Fourth order	
	Adjoint	FD	Adjoint	FD
Gradient vector magnitude	1.895	1.895	1.814	1.813
Angle with second order adjoint	0°	0.252°	2.236°	1.997°
Angle with fourth order adjoint	2.236°	2.354°	0°	0.518°

Table 5.2: The magnitude of second and fourth order C_D gradients and angles between the evaluated gradients for NACA 0012 in an unlimited transonic flow.

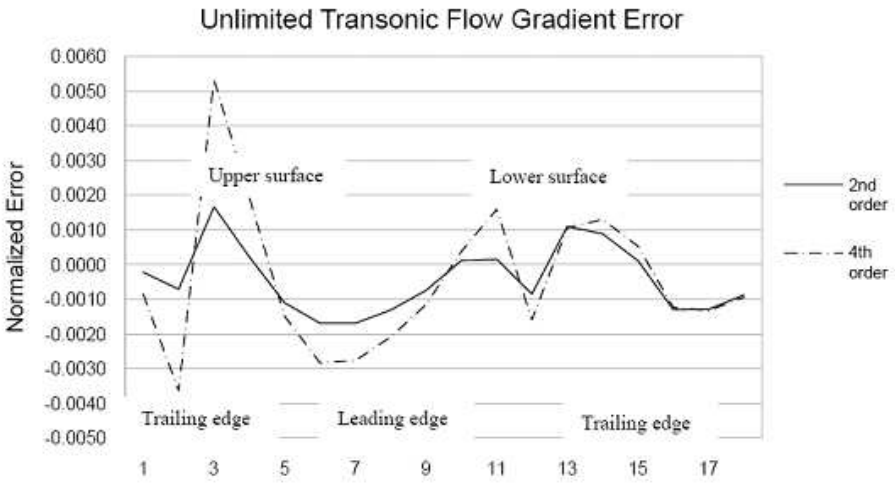
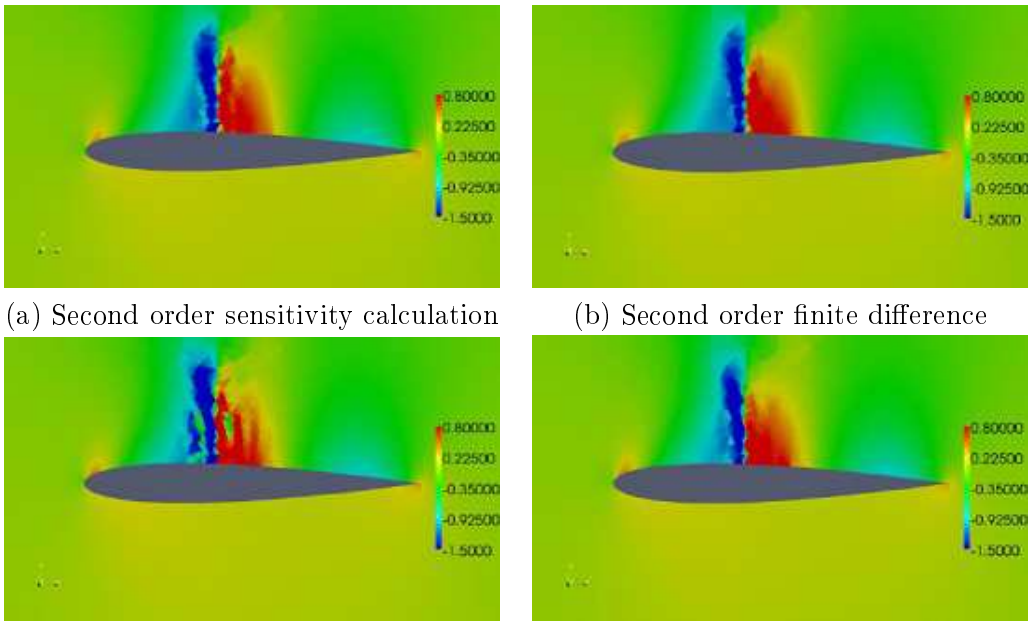


Figure 5.4: C_D gradient error in second and fourth order schemes with respect to the design points, normalized by gradient magnitude.



(a) Second order sensitivity calculation (b) Second order finite difference

(c) Fourth order sensitivity calculation (d) Fourth order finite difference

Figure 5.5: The pressure sensitivity with respect to one of the design control points computed for an unlimited transonic flow around NACA 0012.

5.3 Transonic Test Case with Limiter

In this test case, the impact of using a limiter in the CFD simulation on the accuracy of the computed gradient for second order and fourth order schemes is studied. Two different limiters are used, the Venkatakrishnan limiter [84] and the higher order limiter of Michalak and Ollivier-Gooch [56]. Figure 5.5 shows very good matching of the second order pressure sensitivity computed using adjoint and finite difference techniques with the use of Venkatakrishnan limiter.

Matching is less good between the pressure sensitivity and finite difference results for the fourth order accurate scheme; this lower level of pressure sensitivity matching will lead to less accurate gradient values when using the limited fourth order scheme. Table 5.3 shows that with the use of the Venkatakrishnan limiter, the second order gradient magnitude is a very good match with the corresponding finite difference gradient; the larger error in gradient value observed for the fourth order scheme is comparable to the difference in magnitude between second and fourth order schemes

	Second order		Fourth order	
	Adjoint	FD	Adjoint	FD
Gradient vector magnitude	1.897	1.895	1.887	1.812
Angle with second order adjoint	0°	3.612°	2.292°	3.334°
Angle with fourth order adjoint	2.292°	5.275°	0°	4.873°

Table 5.3: Magnitudes of second and fourth order C_D gradients and angles between the evaluated gradients for NACA 0012 in Venkatakrishnan limited transonic flow

	Second order		Fourth order	
	Adjoint	FD	Adjoint	FD
Gradient vector magnitude	1.676	1.753	1.576	1.672
Angle with second order adjoint	0°	2.845°	30.1°	18.8°
Angle with fourth order adjoint	30.1°	21.05°	0°	18.9°

Table 5.4: The magnitude of second and fourth order C_D gradients and angles between the evaluated gradients for NACA 0012 using higher order limiter in transonic flow.

for the unlimited transonic case. Also, the difference in finite difference and adjoint gradient direction grows to several degrees with the use of Venkatakrishnan limiter. Table 5.4 shows the same behavior with the higher order limiter of C. Michalak and C. Ollivier-Gooch. It shows also that the error in the gradient magnitude is larger compared to the Venkatakrishnan results. The matching in adjoint and finite difference gradient directions is good for the second order scheme but it is not for the fourth order scheme with the use of higher order limiter. It is well known that the use of limiters causes convergence problem due to the non-differentiability of the limiting procedure [55]. The limiter affects both the right and left hand sides of Eq. 1.6. We speculate that the poor matching between finite difference and adjoint gradients for the high-order limiter is related to the non-differentiability of the limiting procedure and that this effect will vary in strength from one limiting procedure to another.

To reduce the influence of $\partial R_i/\partial D$ in control volume i when calculating the adjoint gradient, the fourth order Jacobian was modified numerically by making the non zero structure of the fourth order Jacobian matrix the same as the non zero structure of the second order Jacobian, and dropping the rest of values in the fourth order Jacobian matrix. The right hand side is still constructed with fourth order

	Mod 4th order			
Limiter	Venkat.		HO	
Gradient scheme	Adj.	FD	Adj.	FD
Gradient vector magnitude	1.735	1.812	1.966	1.672
Angle with mod 4th order Venkat. adjoint	0°	2.845°	6.247°	3.338°
Angle with mod 4th order HO. adjoint	6.247°	7.256°	0°	7.207°

Table 5.5: The magnitudes and angles between the evaluated modified fourth order C_D gradients using adjoint, and finite difference for NACA 0012 using Venkatakrishnan and higher order limiters in transonic flow.

accuracy. The above modification doesn't affect the accuracy of the CFD simulation because the right hand side remains fourth order accurate. The computed gradient using this approach is presented in Table 5.5 and shows a reduction in the error in the fourth order computed adjoint gradient, especially in the direction of the gradient. The limiter used from now on is the Venkatakrishnan limiter as it produces an adjoint computed gradient with better matching to the finite difference gradient compared with the high order limiter.

5.4 Sensitivity of finite difference gradient to perturbation magnitude

In the previous sections, I have used perturbation amplitude $\epsilon = 10^{-6}$ and using the central difference formula to evaluate the finite difference gradient. In this section, the sensitivity of the evaluated gradient with respect to ϵ is presented for the hardest case, transonic flow with limiter. The limiter used in the transonic simulation is Venkatakrishnan's limiter. Table 5.6 shows the norm of the drag gradient for different values of perturbation amplitude ϵ; it shows also that decreasing ϵ to be less than 10^{-6} will not practically change the value of the computed gradient norm. Therefore, $\epsilon = 10^{-6}$ is chosen.

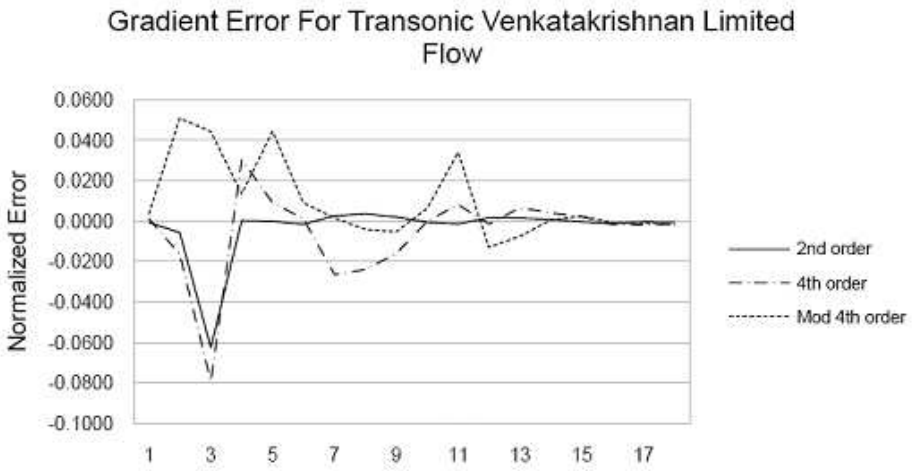


Figure 5.6: The normalized C_D gradient error in second, fourth, and modified fourth order schemes with respect to the design points in a limited transonic flow (Venkatakrishnan limiter)

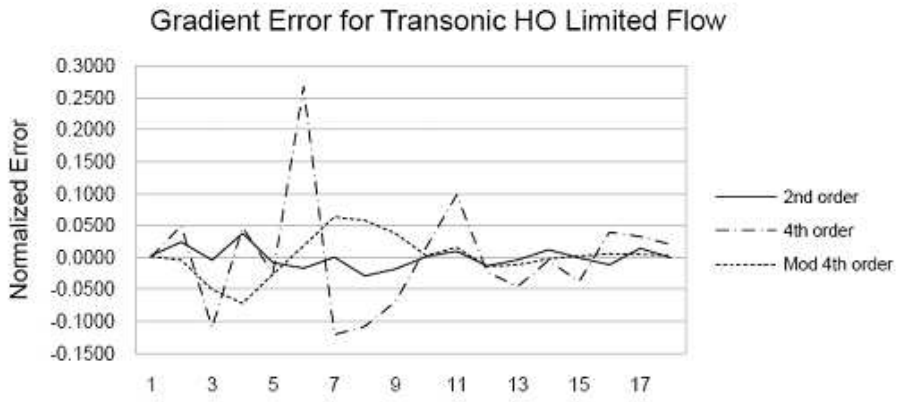


Figure 5.7: The normalized C_D gradient error in second, fourth, and modified fourth order schemes with respect to the design points in a limited transonic flow (higher order limiter).

ϵ	Second order $\left\|\frac{\partial C_D}{\partial D}\right\|$	Fourth order $\left\|\frac{\partial C_D}{\partial D}\right\|$
10^{-3}	1.90237	1.86814
10^{-6}	1.90275	1.87105
10^{-9}	1.09278	1.87114

Table 5.6: Finite difference drag gradient sensitivity with respect to perturbation amplitude ϵ

Chapter 6

GRADIENT BASED OPTIMIZATION TEST CASES

In this section we present four optimization test cases; in all of them we compare the optimal shape resulting from using second and fourth order schemes. The first two cases are inverse design problems, one subsonic and the other transonic. In both test cases, a target pressure distribution is obtained using CFD simulation of a parametrized NACA 2412 airfoil and the starting geometry is a NACA 0012 airfoil. The optimizer will try to find the geometry whose surface pressure distribution matches the target pressure distribution. Those two test cases are important, as the design space has only one solution point at which the resulting pressure distribution from the optimal geometry will match the target pressure distribution. The third test case is a transonic drag minimization with no lift constraint starting from the RAE 2822 airfoil. The objective of this test case is to minimize C_D at M = 0.73 and angle of attack = 2°. In this test case a strong shock wave is formed near mid chord of the initial airfoil geometry and we are seeking a shock free geometry, or at least a geometry that produces a much weaker shock wave. Geometric constraints are applied so the airfoil thickness will be positive all the way along the airfoil section. The fourth test case repeats test case three but adds the lift coefficient as a constraint; in this case, we compare the resulting optimal shape with the optimization results of Brezillon and Gauger [11] .

6.1 Subsonic Inverse Design

In this test case, the target pressure distribution is obtained for the parametrized NACA 2412 at a subsonic condition, $M = 0.5$ and $\alpha = 2°$, using second and fourth order CFD simulations. The starting geometry is the parametrized NACA 0012.

The objective function to be minimized is

$$F = \oint (P_T - P_i)^2 \, dS \tag{6.1}$$

The above objective function and its gradient can be expressed in discrete form as

$$F = \sum (P_T - P_i)^2 \, w_s \tag{6.2}$$

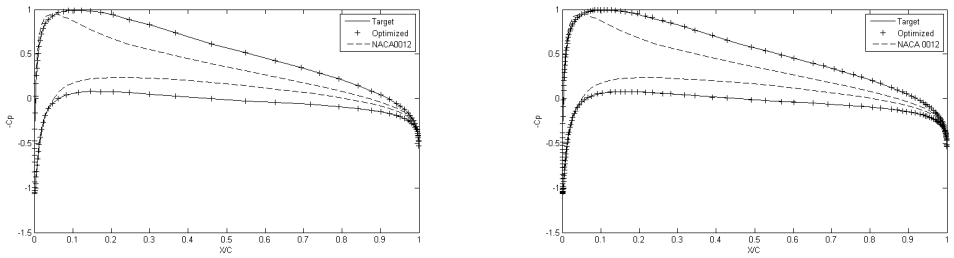
$$\frac{dF}{dx_d} = \sum 2 (P_T - P_i) \left(\frac{-\partial P_i}{\partial x_d} \right) w_s + (P_T - P_i)^2 \frac{\partial w_s}{\partial x_d} \tag{6.3}$$

where w_s is the arc length associated with the surface Gauss point.

Figure 6.1 shows the target pressure distribution of the NACA 2412, the initial pressure distribution of NACA 0012, and the optimized airfoil pressure distribution obtained by the second order and fourth order schemes; both schemes successfully reached the target pressure distribution. Figures 6.2 and 6.3 show the convergence history and gradient norm history for both schemes. Both schemes took about the same number of optimization iterations (28 for second order and 32 for fourth order) to drop the objective function value by eight orders of magnitude. Figures 6.4 and 6.5 show how close the optimized NACA 0012 is to the NACA 2412 using both second order and fourth order schemes. The error between the target geometry and the optimized profile is larger in the fourth order scheme due to inaccuracy of the pressure interpolation scheme used to evaluate the objective function; nevertheless the resulting geometry is an excellent match with the NACA 2412, with an error of less 0.1% of the surface movement.

6.2 Transonic Inverse Design

In this test case, the target pressure is obtained using CFD simulation of the flow over a NACA 2412 airfoil at transonic conditions, $M = 0.73$, $\alpha = 2^\circ$. The objective function to be minimized is again the integration of the square of the pressure difference between the target pressure and the optimized pressure as expressed in Eq. 6.1. The fourth order optimization is based on the modified fourth order gradient evaluation strategy. Figure 6.6 shows the initial, target, and the optimized



(a) Second order (b) Fourth order

Figure 6.1: Subsonic NACA 2412 inverse design pressure distributions for the initial, target, and optimized airfoil profiles

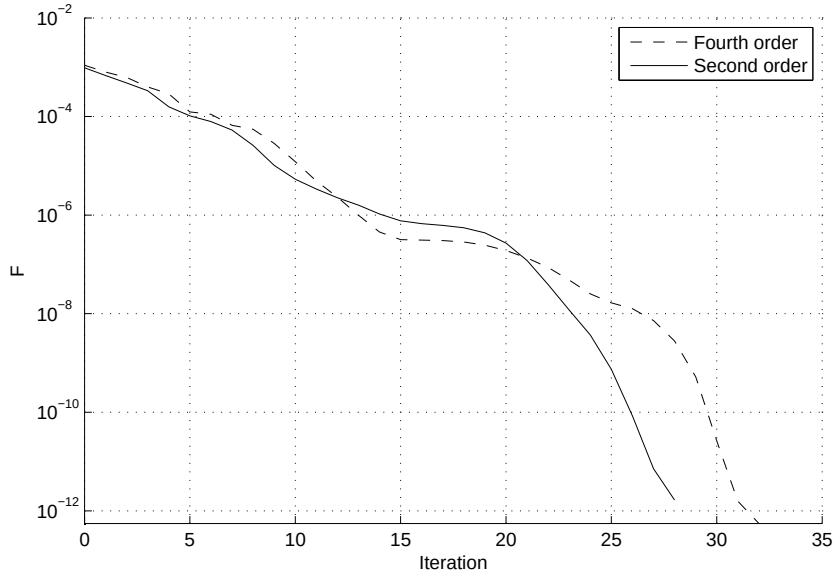


Figure 6.2: Second and fourth order optimization convergence history.

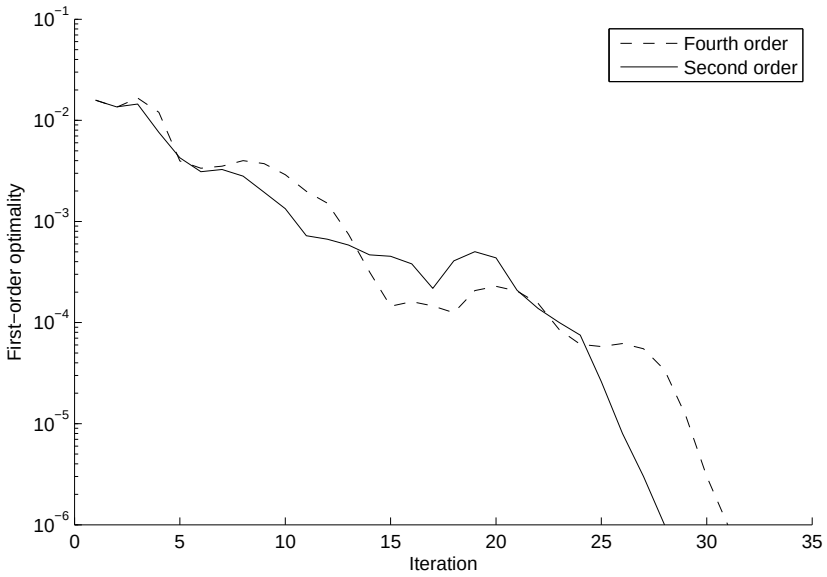


Figure 6.3: Gradient norm history

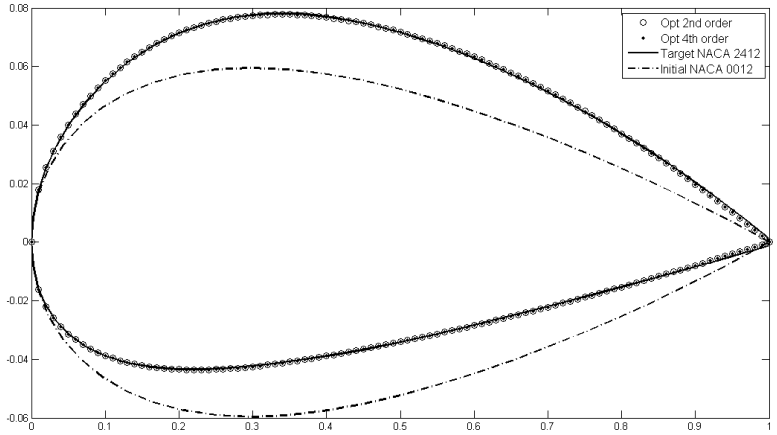
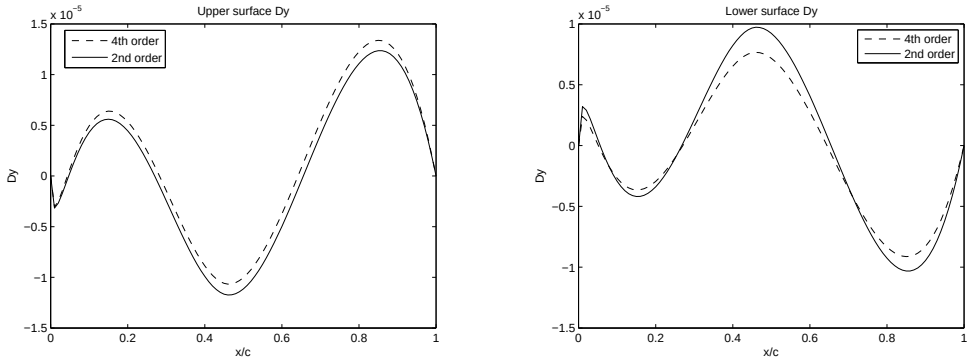


Figure 6.4: Subsonic inverse design optimal airfoil shapes



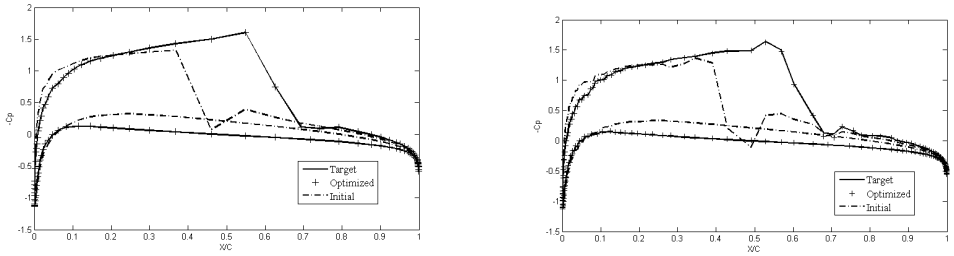
(a) Upper surface (b) Lower surface

Figure 6.5: The difference between the target profile and the optimized profiles, second and fourth order

pressure distribution. The convergence history and gradient norm history are shown in Figs. 6.7 and 6.8; the fourth order scheme is slower to converge compared to the second order scheme, due to the use of larger number of airfoil surface Gauss quadrature points used in the fourth order computations (double the number used for the second order scheme) which makes the minimizer slower to converge. The objective function dropped only three order of magnitudes before convergence stall. This stall is due to the high non-linearity in the target pressure distribution because of the existence of a strong shock wave in it. The noise in the pressure sensitivity generated by the shock wave in the target pressure distribution doesn't allow further convergence; however, the gradient magnitude dropped four order of magnitudes from its initial value. Figures 6.9 and 6.10 show that the optimal shapes for the two schemes differ by less than 10^{-4} of the chord length on the lower surface and less than 10^{-3} chord length on the upper surface (where the strong shock wave exists); both optimal shapes are in good agreement with the NACA 2412: the maximum deviation is about 5% of maximum surface movement.

6.3 Drag Minimization without Lift Constraint

In this test case, minimization of the drag coefficient will be carried out with no lift constraint applied. The airfoil to be optimized is an RAE 2822 at transonic



(a) Second order (b) Fourth order

Figure 6.6: Subsonic NACA 2412 inverse design pressure distributions for the initial, target, and optimized airfoil profiles

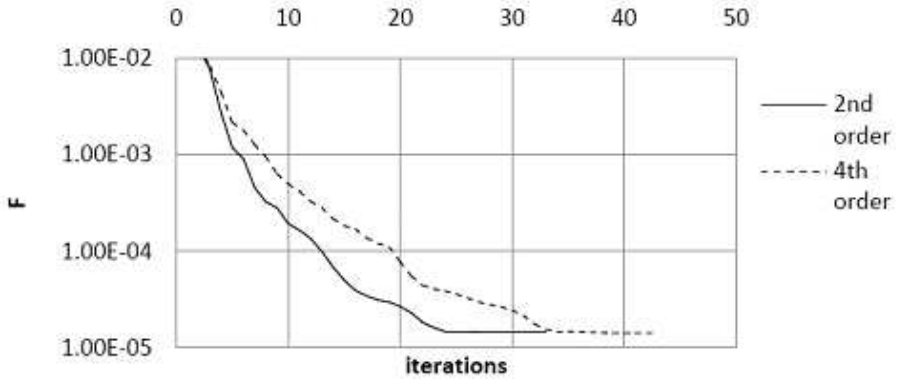


Figure 6.7: Second and fourth order optimization convergence history.

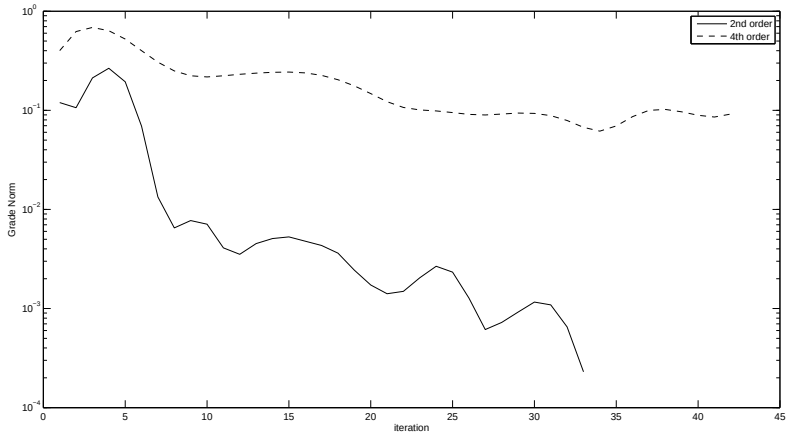
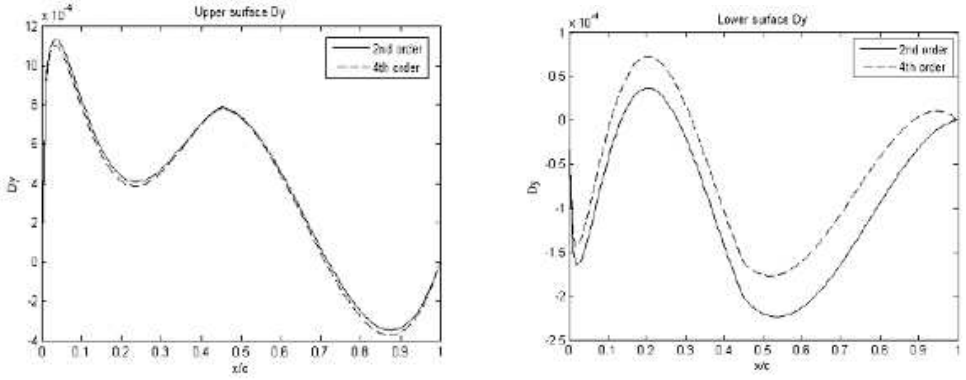


Figure 6.8: Gradient norm history



(a) Upper surface (b) Lower surface

Figure 6.9: The difference between the target profile and the optimized profiles, second and fourth order

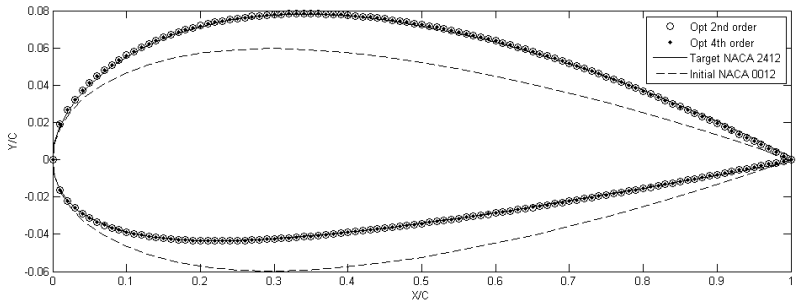


Figure 6.10: Transonic inverse design optimal airfoil shapes.

	C_L	C_L optimized airfoil	C_D	C_D optimized airfoil
Second order	0.865	0.765	0.0081	0.0046[6]
Fourth order	0.849	0.759	0.0099	0.0047

Table 6.1: Aerodynamic coefficients of original and optimized RAE 2822 airfoil at transonic conditions.

conditions: $M = 0.73$ and $\alpha = 2°$. Fig. 6.11 shows the initial solution with a strong shock wave standing at 70% chord. Geometric constraints are added to insure that there is no intersection between the airfoil upper and lower surfaces along the airfoil. Figure 6.12 shows the optimal shapes and the optimized pressure fields resulting from using second and fourth order schemes.

Figures 6.12 and 6.13 show that the difference between the second and fourth order optimal profiles is notable on both upper and lower surfaces. The surface pressure distribution of the original RAE 2822 airfoil and of the optimized airfoil for second and fourth order schemes is shown in Fig. 6.14. Both schemes successfully produced a similar shock free pressure distribution but with different profiles. The original and optimized C_D are shown in Table 6.1. A drag reduction of about 50% is achieved by both schemes. The value of the C_D of the second order optimized airfoil is obtained from fourth order accurate CFD simulation over the optimized second order airfoil profile to eliminate the differences in discretization error in the final results.

Figure 6.16 shows the convergence history of the optimization. Both schemes reached their optimal solution in about the same number of optimization iterations.

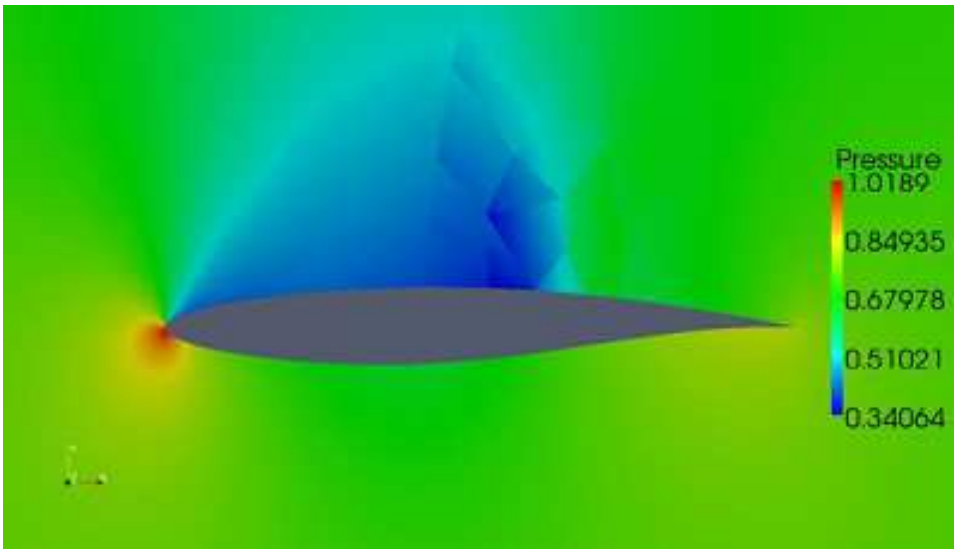


Figure 6.11: Pressure contours of RAE 2822 at Mach 0.73 and angle of attack 2

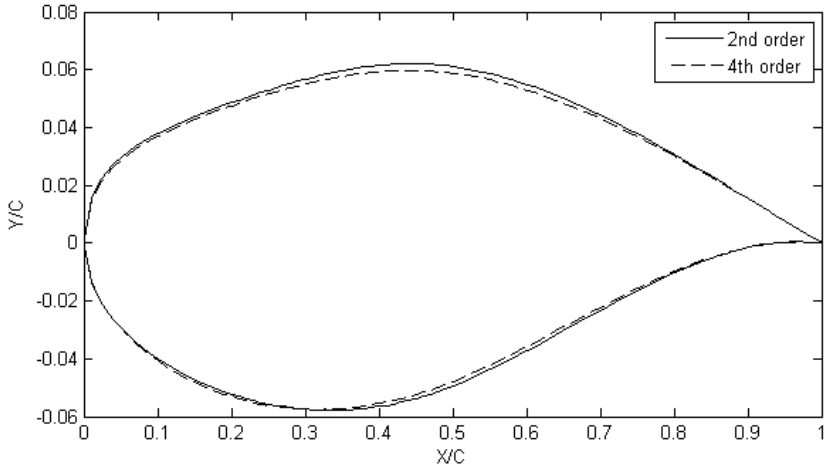
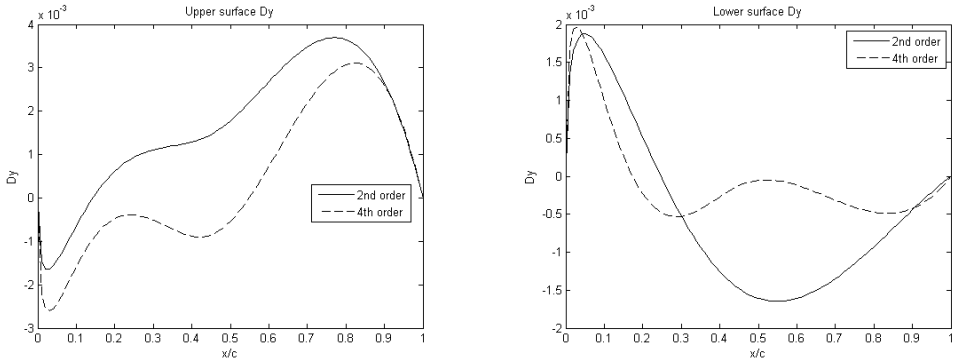


Figure 6.12: Optimized RAE 2822.



(a) Upper surface (b) Lower surface

Figure 6.13: Optimization surface displacements of the original RAE2822 surfaces.

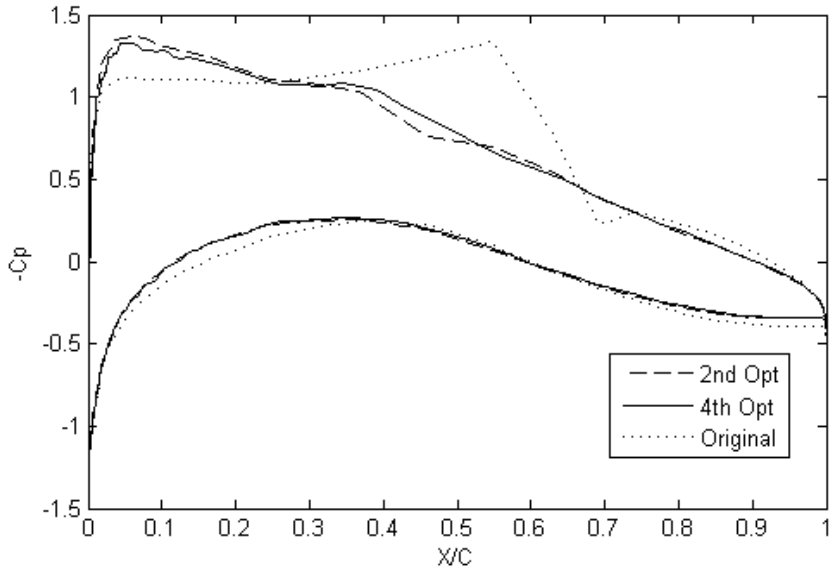
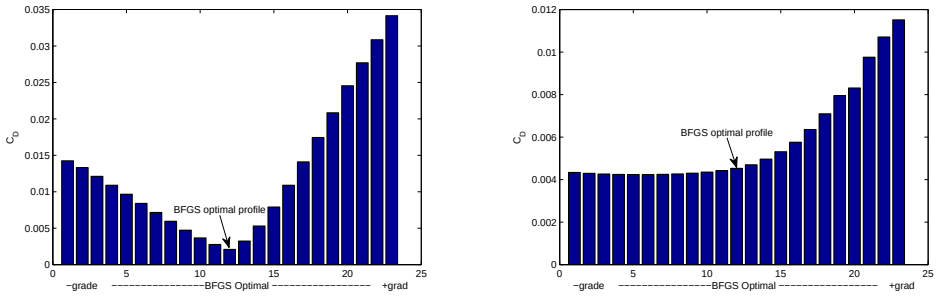


Figure 6.14: Pressure distribution comparison for the original and the optimized geometries.



(a) Second order response surface (b) Fourth order response surface

Figure 6.15: Objective function response surface along positive and negative gradient directions centered at the optimal profile of RAE 2822 transonic drag minimization without lift constraint.

The number of CFD simulations for the second order scheme is 50 compared to 45 for the fourth order scheme, yet the overall computational cost of the second order scheme is 40% less than the fourth order computations. The gradient of the objective function is not zero at the optimal point; to understand the reasons for that, I have plotted the objective function values along positive and negative gradient direction at the optimal solution point as shown in Fig 6.15. The second order response surface shows that the obtained optimal solution is a possible local optimum and the shape of the objective function shows that the gradient computed using finite difference will not be zero at the optimal point. For the fourth order computations, the figure shows that the optimizer reached a near optimal solution. While a slightly lower value is available,the small difference between the objective function value at the obtained optimal point and the minimum value along the negative gradient direction (on the order of the discretization error) may cause a line search failure due to insufficient decrease in the objective function.

6.4 Drag Minimization with Lift Constraint

This test case represents a typical optimization task required in aerospace industry, as it is required to minimize the drag while the lift is unchanged. Therefore the lift coefficient C_L will be an aerodynamic constraint. The original RAE 2822 geometry

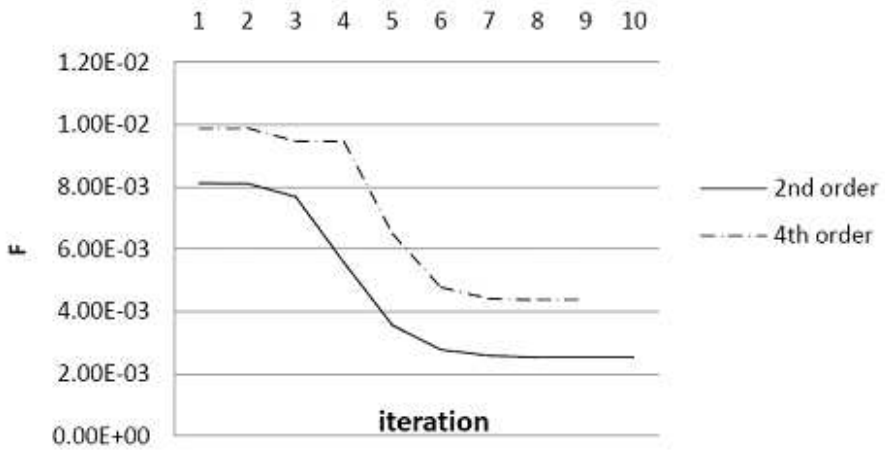


Figure 6.16: Second and fourth order optimization convergence history.

will be used as a starting shape; the objective function to be minimized is

$$F = C_D + 10 \cdot (C_L - C_{L_C})^2 \tag{6.4}$$

where C_{L_C} is the original lift coefficient of RAE 2822 at $M = 0.73$ and $\alpha = 2°$. In this test case both geometric and aerodynamic constraints are applied to avoid getting a non-feasible airfoil geometry, while not affecting airfoil lift coefficient. The nonlinear constraint, the lift coefficient, is added as a penalty term in the objective function as shown in Eq. 6.4. Figure 6.17 show the optimized airfoil's pressure distribution for both second and fourth order schemes. Both second and fourth order schemes travel 99% of the way towards their optimal objective function after 7 iterations; the second order scheme then becomes very slow to reach its optimal value. The second order scheme costs 112 CFD simulations to reach its optimum compared to 49 CFD simulations for the fourth order. Figure 6.20 shows a noticeable difference between the optimized profiles, including a notably larger nose radius and less aft chamber for the fourth order scheme. A shock free geometry is obtained with weak compression waves on the airfoil upper surface. Drag is reduced by about 50%, as shown in Table 6.2, while the lift coefficient is about 1% higher than its original value.

	C_L	C_L optimized airfoil	C_D	C_D optimized airfoil
Second order	0.865	0.871	0.0081	0.0047[7]
Fourth order	0.849	0.853	0.0099	0.0051

Table 6.2: Aerodynamic coefficients of original and optimized RAE 2822 airfoil at transonic conditions.

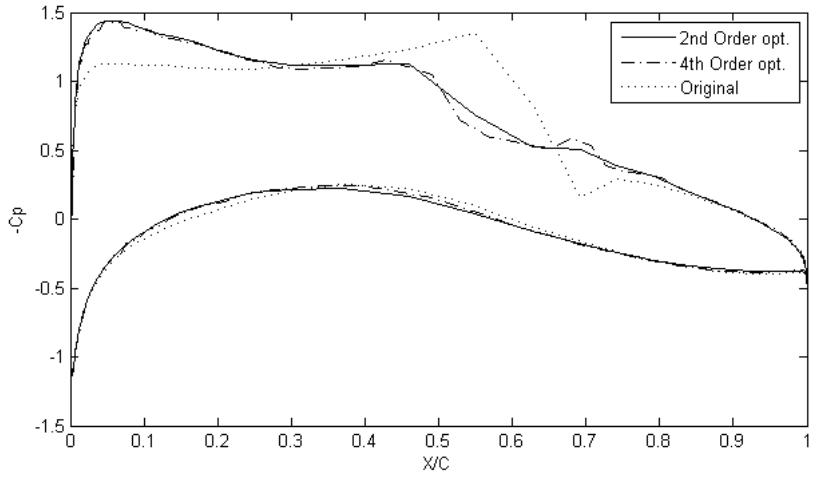


Figure 6.17: Surface pressure distribution, of the initial and optimized geometries

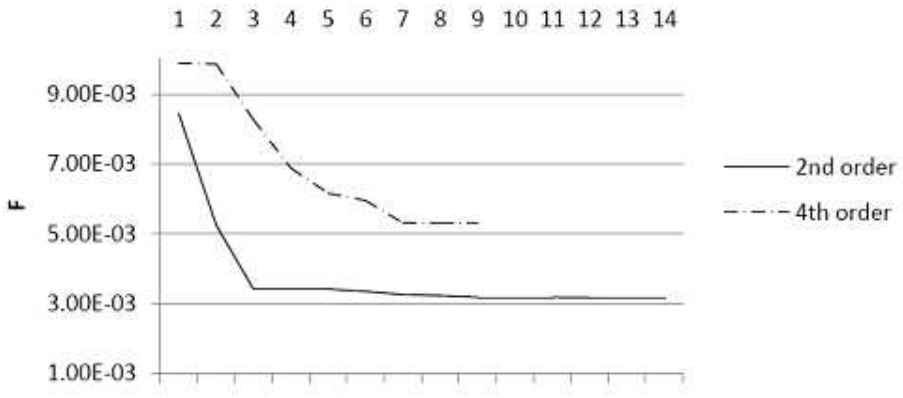


Figure 6.18: Second and fourth order optimization convergence history.

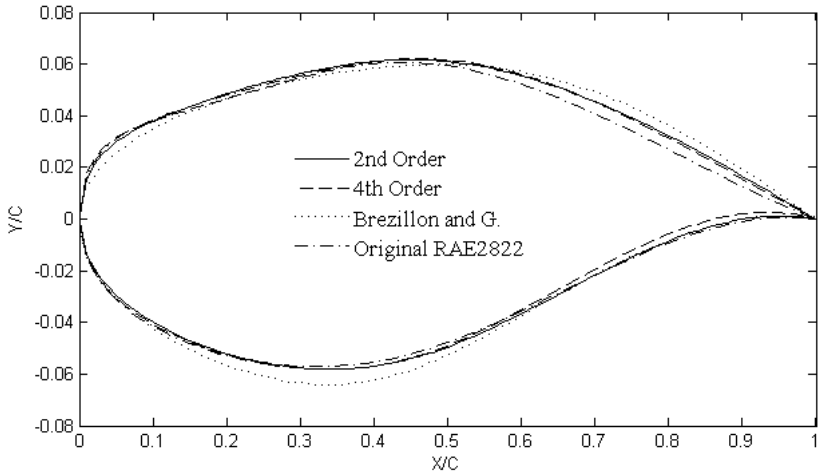
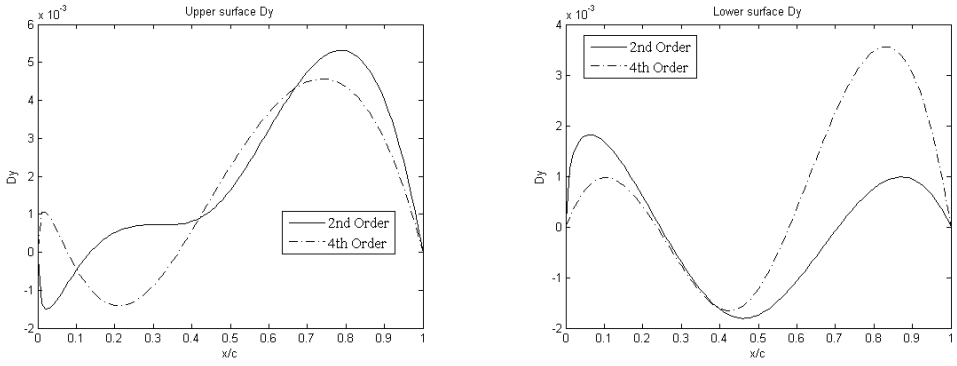
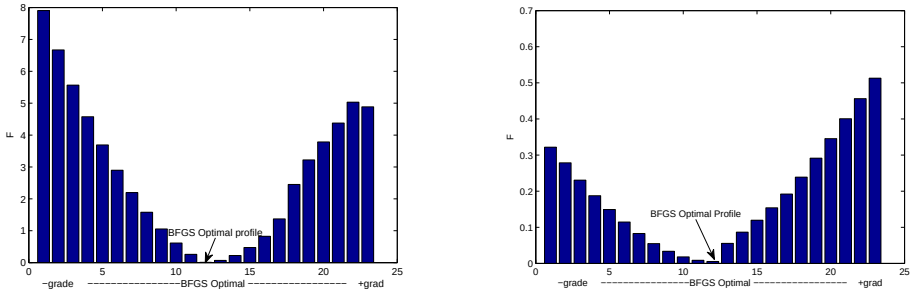


Figure 6.19: Optimal shapes comparison: second order, fourth order, and optimized profile by Brezillon and Gauger compared with the original RAE2822.



(a) Upper surface (b) Lower surface

Figure 6.20: Second and fourth order optimized profile surface displacements from the initial shape.



(a) Second order response surface (b) Fourth order response surface

Figure 6.21: Objective function response surface along positive and negative gradient directions centered at the optimal profile of RAE 2822 transonic drag minimization without lift constraint.

At the optimization breakdown point (optimal solution found), the gradient is not zero, possibly due to the existence of the shock wave which causes high non-linearity in the objective function and lift constraint behavior. Figure 6.21_shows plots of the objective function response along negative and positive gradient direction. It shows also that the obtained optimal solutions are local minima along the gradient direction, and shows also that the gradient computed by finite difference is not zero. It is also worth noting that the gradient of the objective function at the breakdown point is dominated by the gradient of the lift constraint.

Figures 6.19 and 6.20 show a comparison between the optimized airfoil profiles based on second and fourth order computations. Differences in shape are especially noticeable near the leading edge, and in the reflexed part of the lower surface near the trailing edge. The second order optimized profile C_d is four drag counts less than the fourth order profile drag coefficient, nearly 10%. The same problem was studied by Brezillon and Gauger [11]; they used the second order solver MEGAFLOW of the German Aerospace Center and obtained their optimal profile by controlling the camber line of the RAE 2822 via 20 control points. Comparing the optimal pressure distribution of our second order computations shown in Fig 6.17 and that of Brezillon, Fig 6.22, shows that both solvers tend to accelerate the flow near the leading edge upper surface more than the original airfoil shape, followed by a gradual deceleration till the trailing edge; on the other hand, the surface pressure on the lower

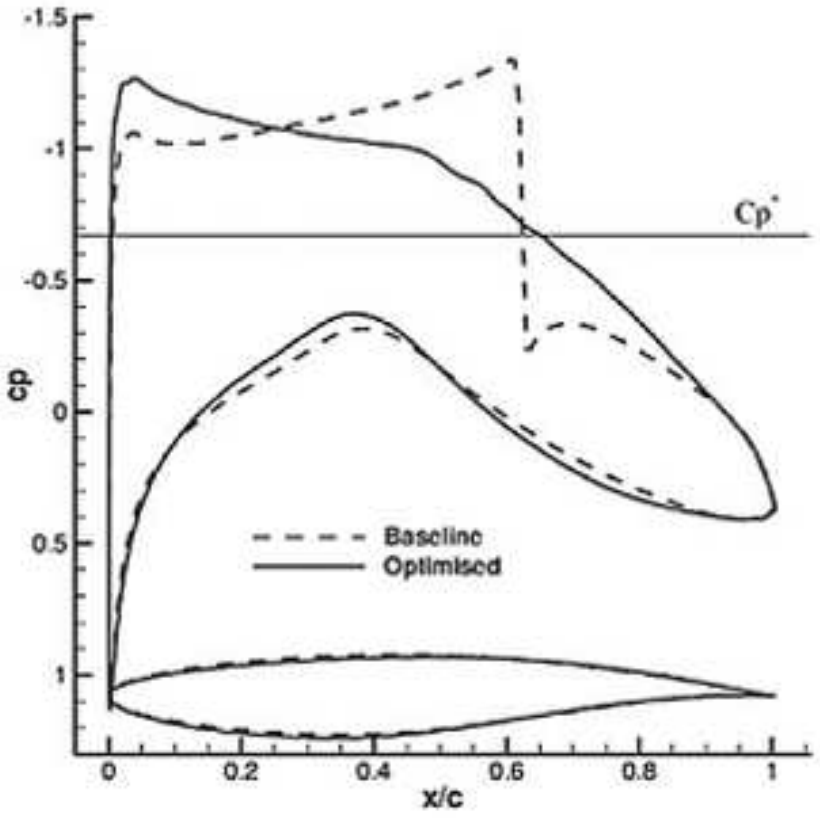


Figure 6.22: initial and optimal pressure distribution obtained by Brezillon and Gauger [11] (presented with permission)

airfoil surface changes only slightly from the original surface pressure distribution of the RAE2822. This similarity in the optimized pressure distribution suggests that whatever geometry parametrization technique is used, the surface pressure changes will be qualitatively similar.

The lift penalty factor used in this test case is selected such that violating the lift constraint by one lift count (10^{-2}) causes 10 drag counts (10^{-3}) increase in the objective function value. Table 6.3 shows the effect of the lift constraint penalty weighting factor on the optimal objective function value, as well as the optimal drag and lift coefficients; these results are based on second order computations.

Lift penalty weight	F_{opt}	$C_{D_{\text{opt}}}$	$C_{L_{\text{opt}}}$	Lift penalty
5	0.0052	0.0047	0.873	3.2×10^{-4}
10	0.0051	0.0047	0.871	3.6×10^{-4}
20	0.0082	0.0080	0.868	1.8×10^{-4}

Table 6.3: Lift penalty weight effect on Drag minimization of RAE 2822 with lift constraint

6.5 Mesh Refinement Study of a Drag Minimization with Lift Constraint

In the last test case, Table 6.2 shows that the optimal value of C_D is different when comparing second and fourth order scheme results. This subsection examines the impact of mesh refinement on the optimal value of C_D. Three mesh grid sizes are used, with 5000, 11000, and 15000 triangles, respectively. In all test cases, $C_{L_C} = 0.84$ is used as a lift constraint value for both second and fourth order computations. Figure 6.23 compares the difference in the optimal C_D value for second and fourth order schemes with mesh refinement; the plotted value of second order C_D optimal are obtained using fourth order CFD simulation on the optimal shape obtained using second order scheme.

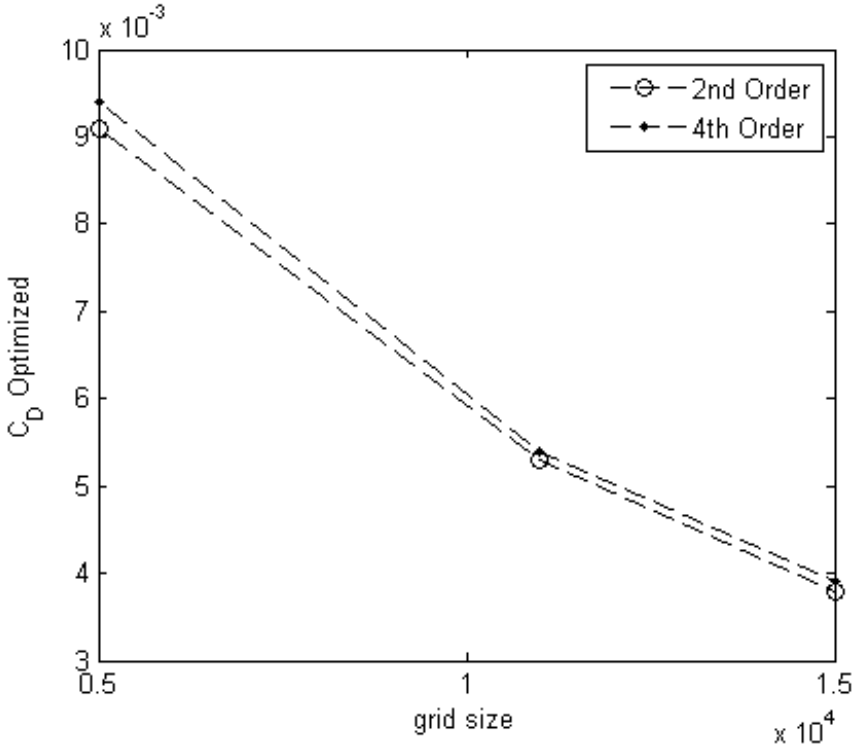


Figure 6.23: Optimal C_D value with mesh refinement

The figure shows that the two schemes tend to reach the same value of the optimal C_D with mesh refinement with only one drag count difference (which is on order of the discretization error). This behavior is expected as transonic CFD simulations using second and fourth order schemes for the same geometry produce almost the same surface pressure distribution and hence objective function value. The root mean square of difference between the second and fourth order optimized airfoil profiles of drops from $3 \cdot 10^{-5}$ on the coarsest mesh to $2 \cdot 10^{-5}$ on the finest grid.

Chapter 7

PARTICLE SWARM OPTIMIZATION AND A NEW HYBRID OPTIMIZATION METHOD

7.1 Introduction to Gradient Free Optimization

Gradient based optimization results depend on the optimization starting point. To find a global optimal solution, gradient independent methods are an obvious candidate. Perhaps the best-known non-gradient based optimization technique is the genetic algorithm [30], a biologically-inspired approach which can find the global optimal point of an objective function with multiple local optima. In genetic algorithm optimization, a randomly generated population of candidate solutions are generated; their fitness is determined by the value of the objective function [29]. Solution points with the best fitness are selected to be parents of the next generation. A new generation of offspring is generated by crossover and mutation of the parents' genetic characteristics (that is, the parents' solution point locations in the design space). This procedure is repeated until the globally optimal solution is obtained. The number of individuals in each generation should not be less than the number of design variables, otherwise poor optimization convergence will result. Figure 7.1 shows a flowchart of a simple genetic algorithm optimizer. The dependency of the population size on the number of design variables and the large number of generations required to converge to the global optimum combine to make genetic algorithm impractically expensive for large aerodynamic optimization problems. Genetic algo-

rithm optimization has been used for aerodynamic optimization problems by many researchers. A genetic algorithm was used for transonic wing drag minimization with a lift constraint by Gage and Kroo [29]. Anderson used GA for wing aerodynamic shape optimization with structural constraints [1]. Jang and Lee used GA to maximize lift to drag ratio of an airfoil using the Euler flow model with the NACA 0012 as a starting geometry [44]. Oyama et al applied a genetic algorithm with a Navier-Stokes solver for transonic wing optimization [70]. They also explored the use of fractal analysis in GA aerodynamic optimization [71].

Particle swarm optimization is a random search technique first introduced in 1995 as a technique that simulate the behavior of a herd of predators hunting for food [46]. This technique has proved to be faster than genetic algorithm to reach the global optimal by many researchers (for example, [34, 59]); swarm intelligence has the advantage of not being highly sensitive to problem dimensions [79], which is not true for evolutionary optimization like genetic algorithm. The particle swarm optimization method (described in Section 7.2) can be made more efficient by hybridizing it with sequential quadratic programming, as described in Section 7.3. The effectiveness of this hybrid approach for unconstrained and constrained aerodynamic optimization is demonstrated in Section 7.4.

7.2 Swarm Intelligence

Particle swarm optimization is a random search method that searches for the global optimal solution of an objective function. It uses a number of swarm particles that scans the design space looking for the global optimal solution. Each swarm particle represents a point in the design space and it moves through the design space according to a simple formula. A particle's velocity depends on its inertia, its own experience, and social experience gained by communicating with other swarm particles. The particle swarm technique keeps a record of location of the best objective function value for each particle ($\overrightarrow{p_i}$) as well the global best location ($\overrightarrow{g}$), which is the best location among the $\overrightarrow{p_i}$'s. Algorithm 7.1 is the general pseudo code of the particle swarm technique.

The velocity of the particle in an optimization iteration depends on

- Its velocity in the previous iteration (momentum part)

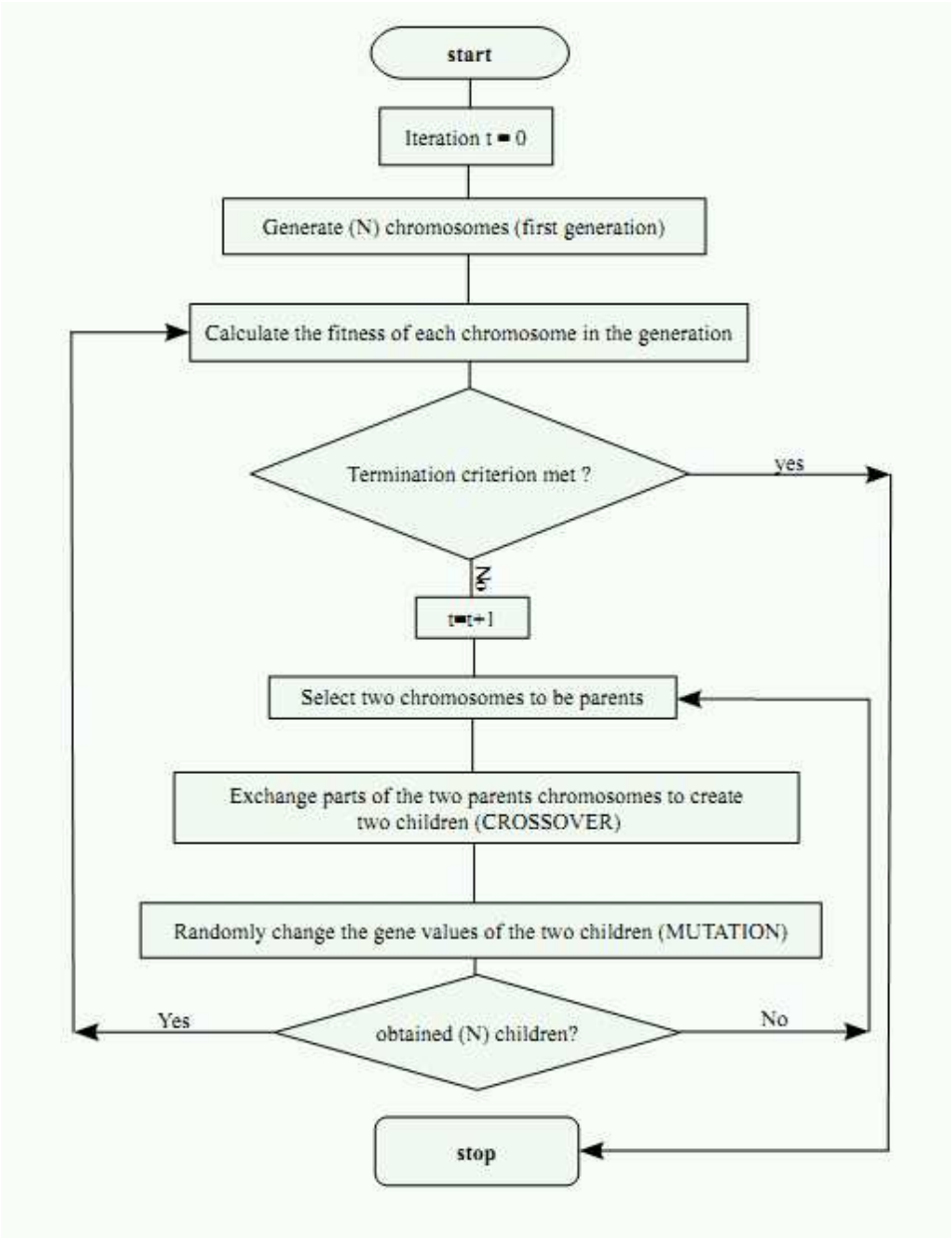


Figure 7.1: Genetic Algorithm optimization

Algorithm 7.1 Particle swarm pseudo code

For each particle i in the swarm
  Initialize particle position $\overrightarrow{x_i}$
  Set its best position to be its initial position: $\overrightarrow{p_i} = \overrightarrow{x_i}$
  Initialize its velocity $\overrightarrow{V_i} = \vec{0}$
End
Set the global best position: $\overrightarrow{g} = \overrightarrow{p_i}$ where $F(\overrightarrow{p_i}) < F(\overrightarrow{p_j})$ for all $i \neq j$

Do
  For each particle i
    Calculate its velocity$\overrightarrow{V_i}$
    Update its position $\overrightarrow{x_i}$
    Calculate its objective function value (fitness) $F(\overrightarrow{x_i})$
    If the current fitness value is better than its best fitness ($F(\overrightarrow{x_i}) < F(\overrightarrow{p_i})$)
      Set the current particle position to be its best position: $\overrightarrow{p_i} = \overrightarrow{x_i}$
    End if
  End for
  Select the best particle objective function and position to be the global best if it is better than the stored value

While max iteration count not reached or convergence criterion not met.

- Its distance from its known best value $\overrightarrow{p_i}$ (cognitive part)
- Its distance from the swarm global best value $\overrightarrow{g}$ (social part)

Figure 7.2 shows schematically how the above three factors affect the particle velocity, the velocity of each particle and its position update are obtained using the following formulas

$$\overrightarrow{V_i}(k+1) = w\overrightarrow{V_i}(k) + c_1\,[\overrightarrow{r_1}] \cdot \{\overrightarrow{p_i} - \overrightarrow{x_i}(k)\} + c_2\,[\overrightarrow{r_2}] \cdot \{\overrightarrow{g} - \overrightarrow{x_i}(k)\} \tag{7.1}$$

$$\overrightarrow{x_i}(k+1) = \overrightarrow{x_i}(k) + \overrightarrow{V_i}(k+1), \quad i = 1..N \tag{7.2}$$

where $w = 0.73$ is the inertial weight factor, $c_1 = c_2 = 1.49$ are the cognitive and social acceleration factors respectively; these constants are selected according to Clerc's constriction models [16]. $\overrightarrow{r_1}$ and $\overrightarrow{r_2}$ are two random vectors of size N whose elements are independent and uniformly randomly selected from the uniform distribution on the interval $[0, 1]$. The "$\cdot$" operation is a component by component multiplication not a dot product and this is true from this point through the rest of the chapter.

Particle swarm optimization suffers from solution stagnation once the particles have converged to a high quality locally optimal solution which is not a true globally optimal point [23]. To avoid this problem, a restart strategy is suggested when premature convergence is detected [9]. At each restart, swarm particles start from new positions in the design space and converge to an optimal solution; restarting is carried out m times, and the optimal solution is the best of all global bests. This technique is called the multi-start particle swarm optimization (MPSO). A drawback of this mechanism is that restarting may cause repetition of the search computations. Evers and Ben Ghalia suggested a more efficient mechanism to escape from high quality local wells. They re-scatter the particles around the convergence point in a region with radius smaller than the design space norm, and large enough some at least some particles to be outside the basin of attraction of the local optimum [24, 23]; their scheme is called the regrouped particle swarm optimization (RegPSO). The main three elements in their scheme are

- Prematurity detection.

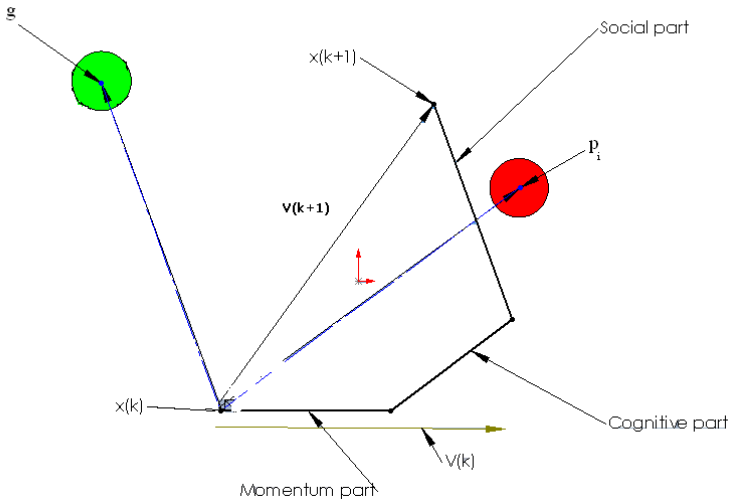


Figure 7.2: Velocity components and position update of a particle

- Redefining upper and lower boundaries for decision (design) variables.
- Regrouping the particles by scattering them on a search domain defined by the new upper and lower boundaries.

In the following subsections, the above three elements of the RegPSO strategy will be presented; interested readers are encouraged to consult Evers and Ben Ghalia [23] for more detail. All constants presented in the next subsections were found by Evers and Ben Ghalia using numerical optimization experiments on standard benchmark optimization problems.

7.2.1 Premature convergence detection

Convergence is detected when all particles converge to an optimal solution and start to lose their momentum; this loss of momentum prevents escaping from the neighborhood of a high quality local optimal solution. Premature convergence can be detected when the greatest distance between a swarm particle and the global best point $\overrightarrow{g}$ falls below a certain threshold. This was suggested by Van den Bergh and adopted in Ever's work with success [9, 23].

For multidimensional optimization problem of size n and m particles, the decision variables $\overrightarrow{x} = \begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_n \end{bmatrix}^T$ have initial upper and lower limits $\overrightarrow{x_U^0} = \begin{bmatrix} x_{1U} & x_{2U} & x_{3U} & \cdots & x_{nU} \end{bmatrix}^T$ and $\overrightarrow{x_L^0} = \begin{bmatrix} x_{1L} & x_{2L} & x_{3L} & \cdots & x_{nL} \end{bmatrix}^T$. First, the range is defined as

$$\overrightarrow{\text{range}}\,(\Omega) = \overrightarrow{x_U} - \overrightarrow{x_L} \tag{7.3}$$

Also the normalized swarm radius can be found according to

$$\delta_{\text{norm}} = \frac{\max_{i=1\ldots m} \|\overrightarrow{x_i}\,(k) - \overrightarrow{g}(k)\|}{\|\overrightarrow{\text{range}}\,(\Omega)\|} \tag{7.4}$$

Convergence is detected when the normalized swarm radius δ_{norm} drops below $\varepsilon = 1.1 \cdot 10^{-4}$.

7.2.2 Redefining decision variables range

Once convergence is detected, the upper and lower limits of the decision variables for the regrouped swarm are computed. The initial range of particle j can be found using its upper limit x_{jU}^0 and lower limit x_{jL}^0as

$$\text{range}_j\left(\Omega^0\right) = x_{jU}^0 - x_{jL}^0 \tag{7.5}$$

Upper and lower limits of the decision variables are changed when convergence is detected in order to scatter swarm particles in a ball surrounding the converged solution. Changing the range of each decision variable j in a regrouping stage r is done according to the following formula [23]:

$$\text{range}_j\left(\Omega^r\right) = \max\left(\text{range}_j\left(\Omega^0\right),\, 1.2 \times 10^4 \max_{i=1\ldots m}\left|\overrightarrow{x_{i,j}^{r-1}} - \overrightarrow{g_j^{r-1}}\right|\right) \tag{7.6}$$

When the current range of design variables range_j is sufficiently small, the optimization problem is considered converged.

7.2.3 Regrouping and position clipping

Regrouping particles in a region surrounding the best position $\overrightarrow{g}$ not only moves them away from the local minimum but allows them to recover their momentum,

which had been lost due to convergence. Regrouping is done according to the following formula

$$\overrightarrow{x_i^r} = \overrightarrow{g} + [\overrightarrow{r_i}]^T \cdot \overrightarrow{\text{range}\,(\Omega^r)} - \frac{1}{2}\overrightarrow{\text{range}\,(\Omega^r)} \tag{7.7}$$

where $\overrightarrow{\text{range}\,(\Omega^r)} = [\text{range}_1\,(\Omega^r)\,, \text{range}_2\,(\Omega^r)\,, \ldots, \text{range}_n\,(\Omega^r)]^T$.

Equation 7.7 requires position clipping to keep the particles within the constrained design space, i.e $x_{j,L}^r \leqq x_{i,j} \leqq x_{j,U}^r$. The upper and lower limits of the components of particle's decision vector can be found as

$$\begin{aligned} x_{j,L}^r &= \max\left(x_{j,L}^0,\, g_j^{r-1} - \frac{1}{2}\text{range}_j\,(\Omega^r)\right) \\ x_{j,U}^r &= \min\left(x_{j,U}^0,\, g_j^{r-1} + \frac{1}{2}\text{range}_j\,(\Omega^r)\right) \end{aligned} \tag{7.8}$$

7.3 Hybrid SQP-RegPSO Technique

Swarm intelligence starts with a set of randomly generated position points in the design space and defines its initial best location, $\overrightarrow{g}$, as the point of best fitness among the swarm particles; this best point act as a gravity center in the design space and attracts other particles to it; if the initial best point is of high quality, this reduces the total computational effort to find the global optimal solution. The technique we are proposing here for hybridization of sequential quadratic programing (SQP) and particle swarm optimization is to use SQP to find the initial high quality $\overrightarrow{g}$ before using RegPSO technique. The pseudo-code of the hybrid scheme is summarized in Algorithm 7.2.

7.4 Optimization Test Cases

This section will examine the efficiency of the proposed hybrid scheme and compare it with the original RegPSO technique. To do so, a drag optimization that seeks a shock free profile with $C_L = 0.4$ at $M = 0.8$ and angle of attack $\alpha = 1.5°$ is carried out. Two distinct starting points were selected to explore the existence of a unique

Algorithm 7.2 Hybrid SQP-RegPSO pseudo code

For each particle in the swarm
  Initialize particle position $\overrightarrow{x_i}$
  Set its best position to be its initial position: $\overrightarrow{p_i} = \overrightarrow{x_i}$
  Initialize its velocity $\overrightarrow{V_i} = 0$
End
Set the global best position: $\overrightarrow{g} = \overrightarrow{p_i}$ where $F(\overrightarrow{p_i}) < F(\overrightarrow{p_j})$ for all $i \neq j$, or the SQP optimal solution, whichever is better
Set $\delta_{\text{norm}} = 1$
Set $t = 1$

Do
  For each particle i
    Calculate its velocity$\overrightarrow{V_i}$
    Update its position $\overrightarrow{x_i}$
    Clip position such that $x_{L,j} \leq x_{i,j} \leq x_{U,j}$
    Calculate its objective function value (fitness) $F(\overrightarrow{x_i})$
    If the current fitness value is better than its best fitness ($F(\overrightarrow{x_i}) < F(\overrightarrow{p_i})$)
      Set the current particle position to be its best position: $\overrightarrow{p_i} = \overrightarrow{x_i}$
    End if
  End for
  Select the best particle objective function and position to be the global best if it is better than the stored value

$t = t + 1$
Compute δ_{norm} using Eq 7.4
If $\delta_{\text{norm}} \leq \varepsilon$
  Compute the range of each decision variable range $\left(\Omega_j^r\right)$
  Scatter the particles around $\overrightarrow{g}$ according to Eq 7.7
  Apply position clipping with limits defined according to Eq 7.8
End if
While ($\delta_{\text{norm}} > \varepsilon$ and $t \leq$ max iterations).

global best[8]; the first is to start from NACA 0012, the second one is to start from NACA 00083 airfoil.[9] These cases were chosen because experiments showed that SQP optimization with these two starting points converged to distinct optimal solutions with significantly different objective function values. All the computations here are second order as we aiming to find a global optimization method with reasonable computational cost, not to compare second and fourth order optimal results.

7.4.1 Constrained drag optimization of the NACA 0012

In this test case, drag optimization of the NACA 0012 at a lift coefficient of $C_L = 0.4$, $M = 0.8$ and angle of attack $\alpha = 2^\circ$ is presented. The objective function can be written as

$$F = C_D + 10 \cdot (C_L - 0.4)^2 \tag{7.9}$$

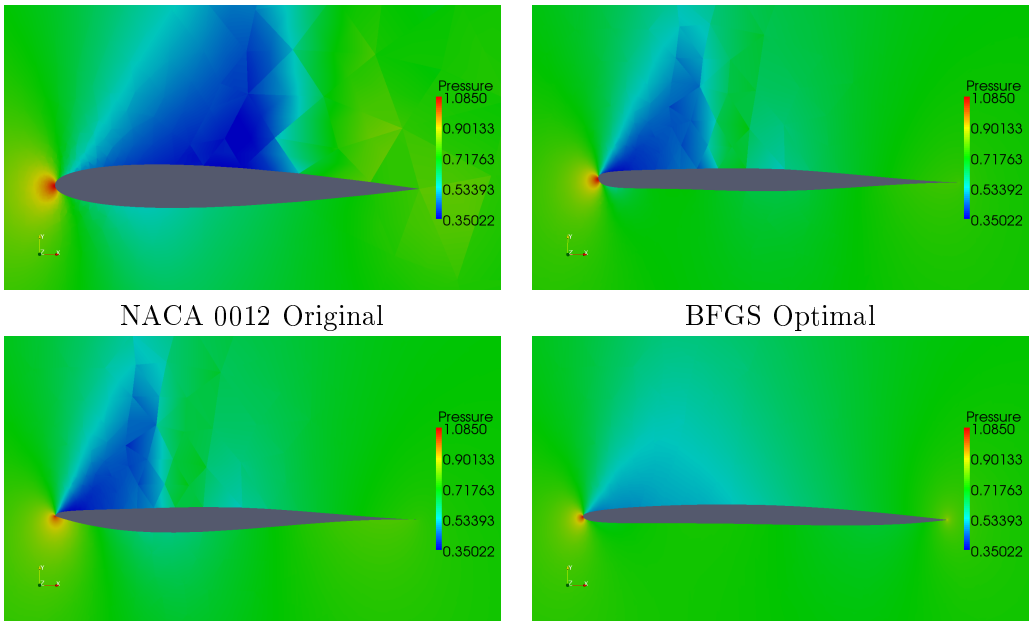
SQP optimization is used first to find a local optimum and thus provide an initial high quality $\overrightarrow{g}$ as a starting point for the RegPSO scheme. The optimal profile found by SQP optimization with BFGS Hessian approximation was not shock free. RegPSO is used after SQP and successfully reached a shock free profile. In this test case, the swarm size was 10 particles and the number of decision variables is 18. The first phase of the hybrid scheme (SQP optimization) costs 92 CFD simulations while the second phase costs 490 CFD simulations which makes the total cost 582 CFD simulations to reach shock free profile.

Using RegPSO alone as an optimization technique with a maximum of 1000 CFD simulations does not result in a shock free profile, as the optimization has not yet converged to a global optimum. Fig 7.4.1 shows the pressure distribution of the original NACA 0012 at transonic conditions, the SQP optimal profile, the RegPSO optimal shape, and the hybrid shock free profile. Drag has been reduced from 150 drag counts at the initial local optimum found by SQP optimization to 8.89 drag counts at the final global optimum.

Figure 7.4 shows the surface pressure distribution of the original NACA 0012 airfoil, and the optimized profile using SQP with BFGS approximate Hessian, RegPSO,

[8] In the case of a finite region in the design space for which the flow is shock free, there may be a region of global "best" solutions differing only in the amount of discretization error present.

[9] That is, an airfoil from the 4-digit NACA series with 8.3% maximum thickness.



NACA 0012 Original | BFGS Optimal

RegPSO 1000 CFD simulations | Hybrid 492 Total CFD simulations

Figure 7.3: Pressure Field of the original NACA 0012 and the optimized profiles

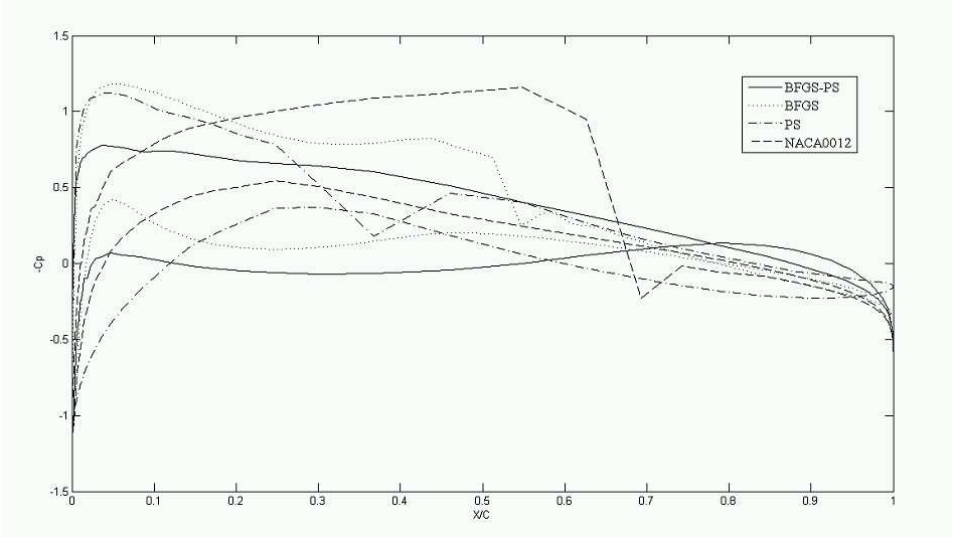


Figure 7.4: Surface pressure distribution of original NACA 0012 and the optimized profiles

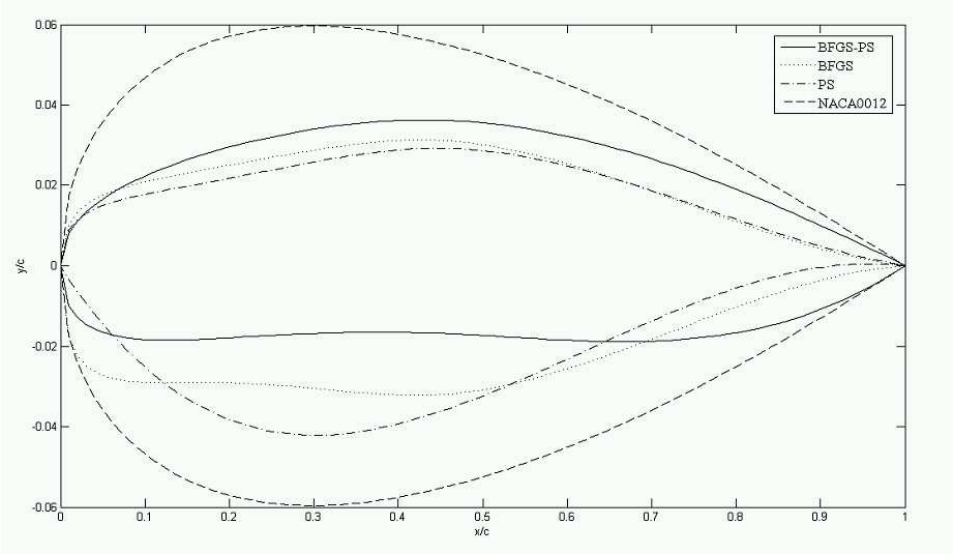


Figure 7.5: Airfoil shapes of original NACA 0012 and the optimized profiles

and hybrid BFGS-RegPSO techniques. Figure 7.5 shows the different optimized airfoil profiles resulted from the three optimization techniques, note that the initial global best point of the hybrid scheme is the SQP optimal shape. To study the function variation between the initial best known shape, which results from SQP optimization, and the final best shape of the hybrid scheme, the vector that connects these two points in design space was divided into 100 steps and the objective function value at each of these points computed. Figure 7.7 shows that there is a descent direction (the vector that connects the initial and final $\overrightarrow{g}$) at the SQP optimal point but the SQP optimizer can not find it. Possible reasons for that include:

1. An error in the computed adjoint gradient.

2. The SQP gradient and the vector that connects the initial and final $\overrightarrow{g}$ are perpendicular due to the influence of numerical noise and of the behavior of the Euler flow model physics in the SQP gradient as the descent direction resulting from adjoint computations tends to minimize the drag by shifting the shock wave forward so it occurs at lower Mach number to reduce its strength, while this mechanism to reduce the drag is not strictly followed by RegPSO scheme .

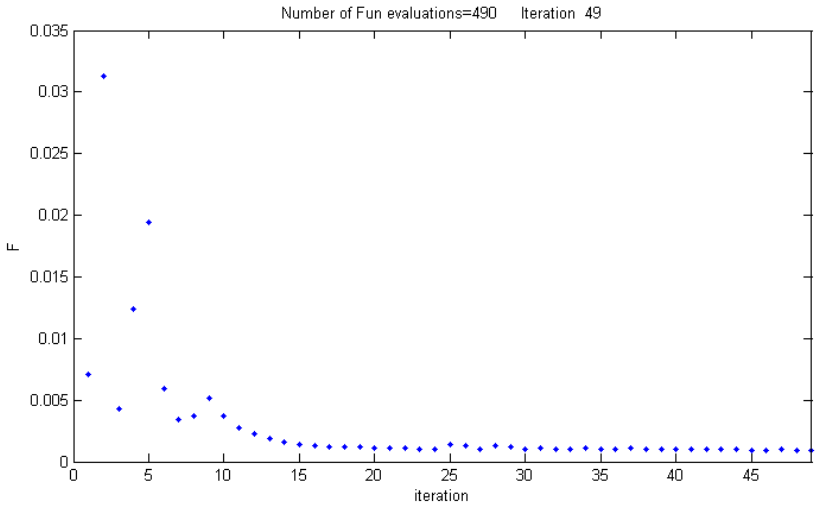


Figure 7.6: Function minimization iterations in the second phase of the hybrid scheme (RegPSO) with a NACA 0012 airfoil as the starting point.

To determine the real reason, we computed the adjoint gradient and the finite difference gradient at the SQP local optimum, then computed the angles between the

- Adjoint computed gradient vector.
- Finite difference computed gradient vector.
- Vector that connects the SQP (local) optimum and the hybrid (global) optimum.

Table 7.1 shows the angles between those three vectors. The adjoint and finite difference gradient directions are in excellent agreement but are almost perpendicular to the third vector that connects local and global optima. This observation supports the hypothesis that the influence of numerical noise and Euler flow model physics prevents finding that descent direction; however, this needs to be verified by another test case.

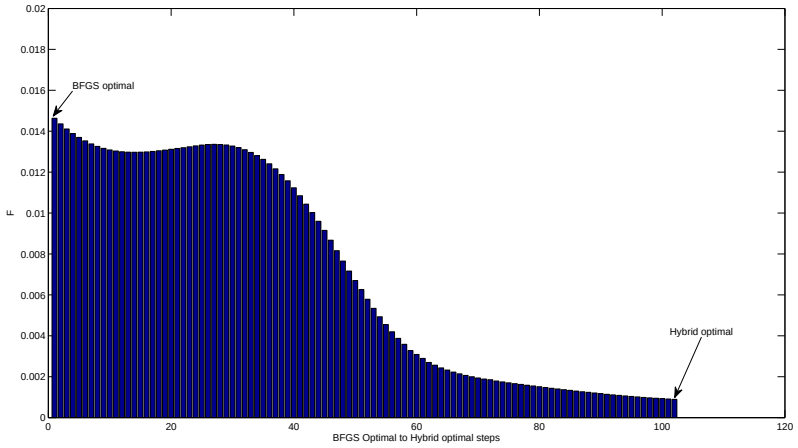


Figure 7.7: Objective function values between SQP and hybrid optimal points (starting geometry: NACA 0012).

	Angle in degrees
Adjoint and finite difference gradients	1.78
Adjoint gradient and SQP-hybrid vector	86.6
Finite difference gradient and SQP-hybrid vector	86.7

Table 7.1: Angles between adjoint, FD., and SQP-hybrid vectors

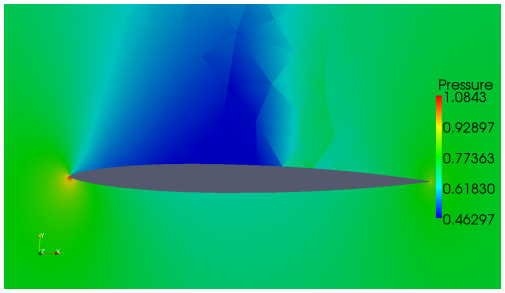
7.4.2 Constrained drag minimization of NACA 00083

In this test case, the same drag optimization problem is studied at the same transonic conditions, but starting from a thinner airfoil. Again, SQP could not eliminate the shock wave on the airfoil's upper surface, but a shock free optimal shape is obtained using RegPSO in the second phase as shown in Fig 7.8.

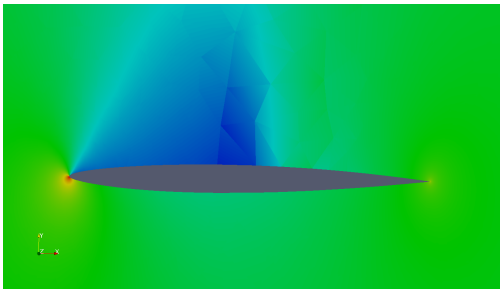
Figure 7.9 shows a comparison of the surface pressure of the original NACA 00083, SQP with BFGS approximate Hessian (local) optimum, and hybrid BFGS-RegPSO (global) optimum results. The BFGS optimal profile weakens the shock strength to reduce the drag, but the hybrid scheme eliminates the shock wave completely. Figure 7.10 shows a profile comparison of the original NACA 00083, BFGS optimal, and the hybrid optimal profile.

The drag coefficient is reduced from 32 drag counts at the BFGS optimal solution to 8.89 drag counts at the hybrid optimal solution. The optimal value of the drag coefficient found by the hybrid scheme in this test case is the same as the value obtained in the previous case and in both cases, it is of order of the discretization error. The total number of CFD simulations is 1080, 80 for the SQP phase and 1000 for the RegPSO phase.

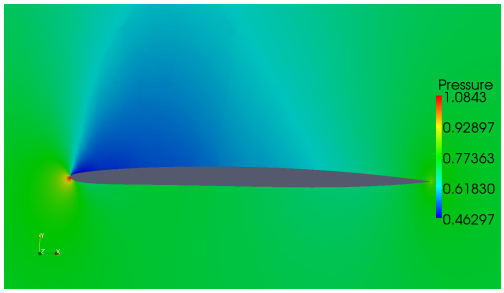
Again, to study the objective function variation between the BFGS (local) optimum, and the final (global) optimum of the hybrid scheme, we took 100 steps on the vector that connects these points and compute the objective function values. Figure 7.12 shows the variation in the objective function value along that vector, which is not a descent vector at the BFGS optimal point. Also, there is numerical noise in the objective function values at some stations along that vector, in addition to an appearance then vanishing of a shock wave. As in the previous test case, we computed the objective function gradient at the initial gBest using adjoint and finite difference strategy and calculated its angle with the vector that connects the initial and the final best position vectors $\overrightarrow{g}$ obtained using the hybrid scheme. Table7.2 shows that the vector connect the initial and final best solutions is almost perpendicular to the adjoint and finite difference gradients; it shows also that adjoint and finite difference gradients are a very good match. The orthogonality of these vectors was previously noted in the last test case as well. This strongly suggests that numerical noise in the gradient and the influence of the flow model physics prevents finding a descent



NACA 00083 original



BFGS Optimal



Hybrid Optimal

Figure 7.8: Pressure surface of NACA 00083 and the optimized profiles

	Angle in degree
Adjoint and Finite difference	0.27
Adjoint and SQP-hybrid vector	86.93
Finite difference and SQP-hybrid vector	86.87

Table 7.2: Angles between adjoint, FD, and BFGS-hybrid best solutions vectors

direction that leads from the SQP local optimum to the shock free global optimum.

7.4.3 Aerodynamic and thickness constraint drag minimization of NACA 0012

In this test case thickness is constrained to be $t_c = 0.1$ at $x_{t_c} = 0.35$, in addition to the lift constraint $C_{Lc} = 0.4$ at transonic conditions, $M = 0.8$ and $\alpha = 1.5°$. SQP with BFGS Hessian approximation optimization technique is first used, the objective

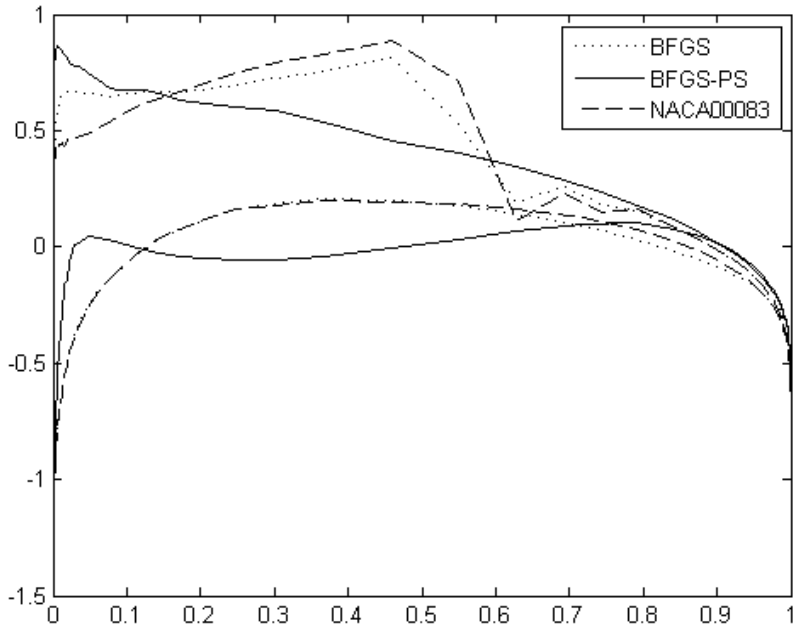


Figure 7.9: Surface pressure of NACA 00083 and the optimized profiles

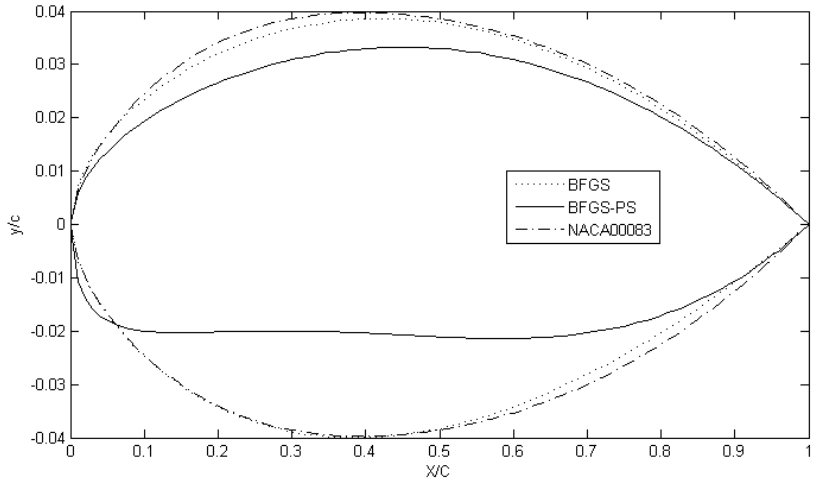


Figure 7.10: Profile comparison of NACA 00083, BFGS optimal, and hybrid optimal airfoils

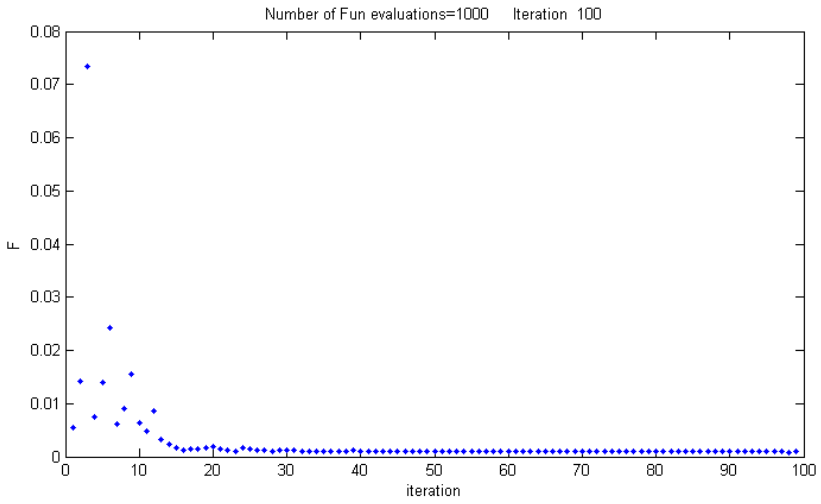


Figure 7.11: Function minimization iterations in the second phase of the hybrid scheme (RegPSO) of NACA 00083 start

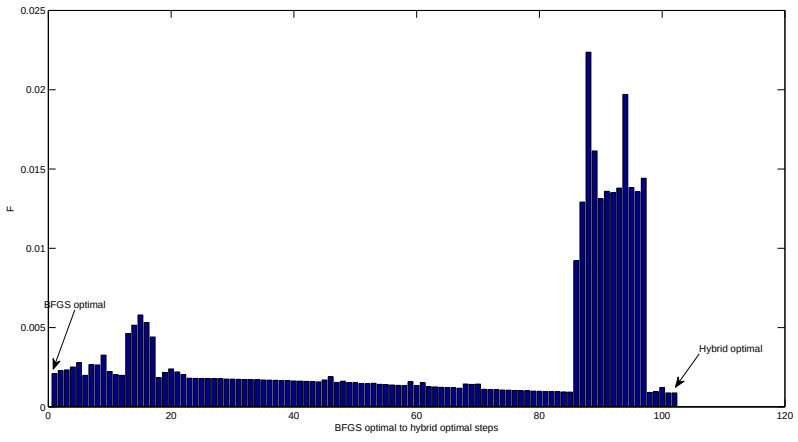


Figure 7.12: Objective function values between BFGS and hybrid optimal points, starting geometry NACA 00083

function to be minimized is

$$F = C_D + 10\,(C_L - 0.4)^2 \qquad (7.10)$$
$$\text{subject to} t\,(x_{t_c}) = t_c$$

The thickness constraint can be cast as an equality constraint equation using the surface parametrization matrix $Q_2\,[AQ_1]^\dagger$; for this test case, $x_c < L_1$, so the thickness constraint equation can be written as

$$t\,(x_{t_c}) = P_1(x_{t_c}) - P_3(x_{t_c}) = t_c \qquad (7.11)$$
$$\therefore \left(a_0\sqrt{x_c} + a_1 x_c + a_2 x_c^2 + a_3 x_c^3\right) - \left(c_0\sqrt{x_c} + c_1 x_c + c_2 x_c^2 + c_3 x_c^3\right) = t_c$$

Recall that the relationship of polynomial coefficients and the design variables can be found using $Q_2\,[AQ_1]^\dagger$ as

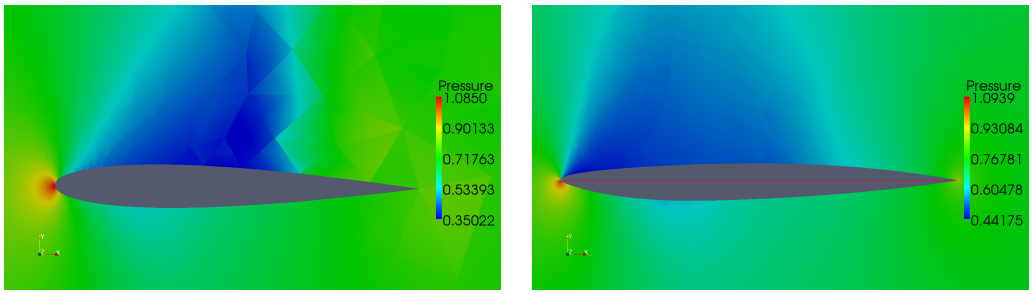
$$a_i = \sum \left(Q_2\,[AQ_1]^\dagger\right)_{i,j} y_j \qquad (7.12)$$

Equation (7.12) can be used with Eq (7.11) to derive an algebraic linear equation that relates the thickness constraint t_c to design variables directly; this last equation can be used as a constraint equation added to the optimization problem and this constraint will be satisfied using a Lagrange multiplier.

For the second phase of the hybrid scheme, the thickness constraint is satisfied using a penalty term in the objective function; the objective function for RegPSO phase can be written as

$$F = C_D + 10\,(C_L - 0.4)^2 + (t\,(x_{t_c}) - t_c)^2 \qquad (7.13)$$

The penalty method is used to approximately satisfy the thickness constraint term due to the lack of Lagrange multiplier in non-gradient based optimization techniques. Drag coefficient have been reduced from 230 to 120 drag counts in the first phase, then reduced to 9 drag counts using the hybrid scheme, a shock free profile is obtained as shown in Figs (7.13, 7.14) ; this shock free profile is obtained with 900 CFD simulations in total.



NACA 0012 original Hybrid Optimal

Figure 7.13: Pressure surface of NACA 0012 and the optimized profiles with thickness constraint

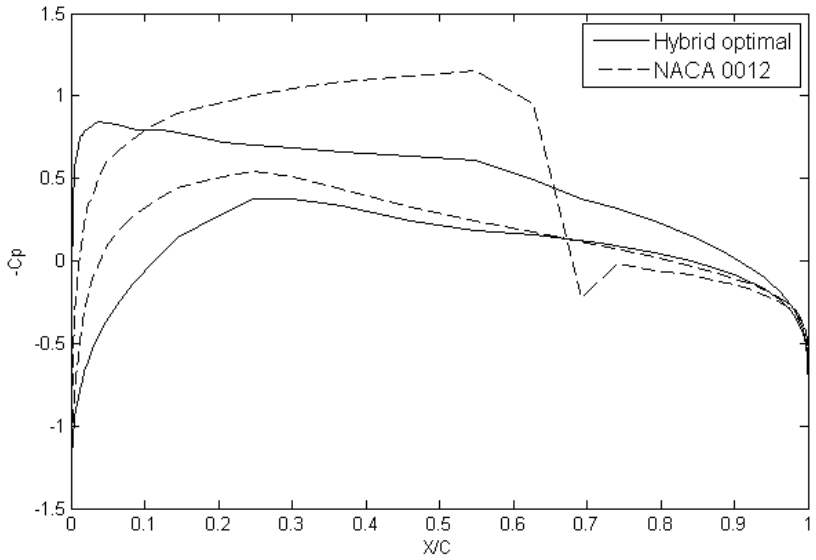
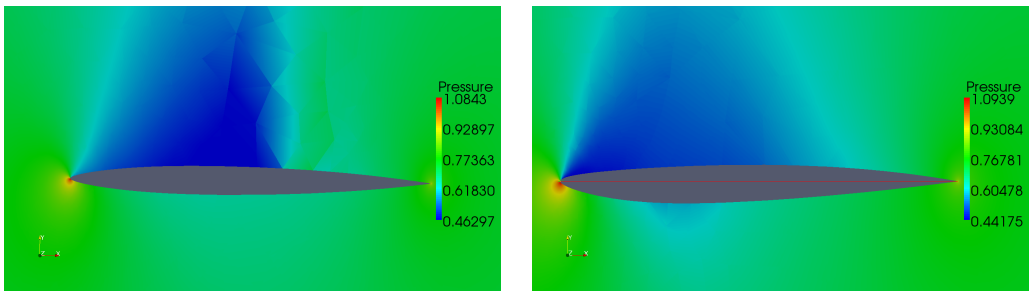


Figure 7.14: Surface pressure of NACA 0012 and the optimized profile with thickness constraint



NACA 00083 original Hybrid Optimal

Figure 7.15: Pressure surface of NACA 00083 and the optimized profiles with thickness constraint

7.4.4 Aerodynamic and thickness constraint drag minimization of NACA 00083

This test case is a repeat of the last one, but the starting geometry is different. The starting geometry is NACA 00083 which does not satisfy the thickness constraint. The SQP optimizer is used first;it satisfies the thickness constraint after the first few iterations, then it starts to minimize the drag. Drag is reduced from 60 drag counts to 40 drag counts in the first phase. The second (RegPSO) phase reduced the drag to 9 drag counts and produced a shock free optimal shape, as shown in Figs. 7.15 and 7.16.

Comparison of the optimized profiles of the last two test cases are shown in Fig. 7.17. The two profiles are very close in upper surface shape, which is tightly constrained by the need to eliminate the shock wave; there is more difference in the lower surface on which the flow is subsonic. The last observation suggests that for inviscid flow simulations, there is a region of global optimal solution rather than one point global optimum. More precisely, once a shock-free solution is obtained, the objective function is too flat to distinguish reliably between the drag of shock-free solutions; this is not surprising, considering that the drag for shock-free inviscid flows is due solely to discretization error.

In all test cases, the new hybrid scheme was able to find a shock free airfoil profile regardless of the starting points. Only ten particles were using in the swarm to scan the design space, which is roughly half of the number of design variables. This is lower than the recommended population size required by GA to converge well. The

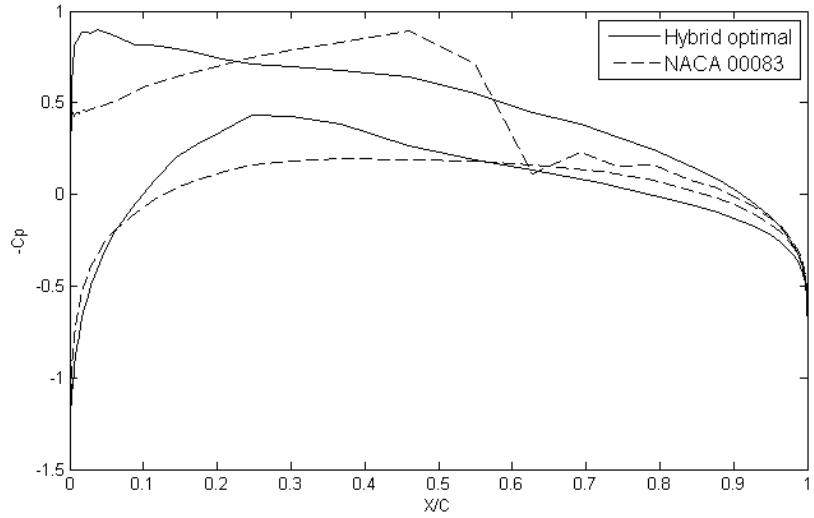


Figure 7.16: Surface pressure of NACA 00083 and the optimized profile with thickness constraint

optimization computational cost is four to nine times the cost of SQP optimization. Using SQP optimization with ten different starting geometries might lead to a shock free profile, but there is no guarantee for that. As the results above show, there is a high chance of getting trapped in local minima or suffering a breakdown in the optimization process because of solution noise.

Table 7.3 shows the effect of the thickness penalty weight on the optimization results using second order accurate computations. It shows that a thickness penalty weight of 10 is sufficient to satisfy the thickness geometric constraint, but it will increase the value of $C_{D_{\text{opt}}}$ and may prevent obtaining a shock free optimal profile. Figure Shows a comparison of the optimal profiles using two different thickness penalty weights; the maximum thickness of the optimized airfoils were 0.077 and 0.095 respectively for thickness penalty of 1 and 10. Therefore it is recommended to use a thickness penalty of order of 10 in order to satisfy a maximum thickness constraint. Figure 7.18 shows a comparison of the optimal pressure distribution of the two optimal profiles; a shock free pressure distribution is obtained using thickness penalty weight of 1. The drag coefficient for this profile is 9 drag counts which is

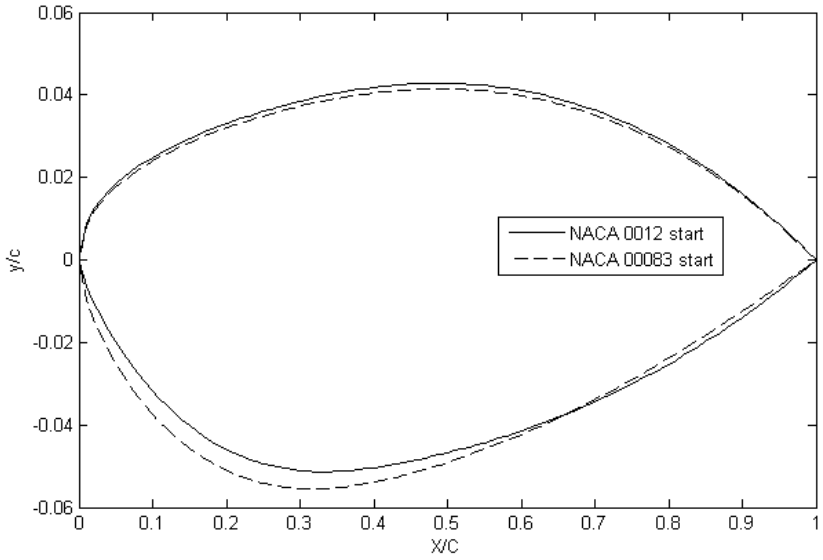


Figure 7.17: Profile comparison of the hybrid scheme optimal profiles, starting geometries NACA 0012 and NACA 00083

of order of discretization error. It shows also that with thickness penalty weight 10, there is a weak compression wave on the airfoil upper surface which causes an increase in the drag coefficient to 17 drag counts.

Thick. penalty weight	F_{opt}	$C_{D_{\text{op}_t}}$	$C_{L_{\text{opt}}}$	$t\,(x_{t_c})$	Lift penalty	Thick. penalty
1	0.00134	0.00088	0.3997	0.077	0.000001	0.00048
10	0.00208	0.0017	0.4005	0.095	0.0000026	0.00025

Table 7.3: Thickness penalty weight impact on the optimization results of NACA 00083 using hybrid scheme

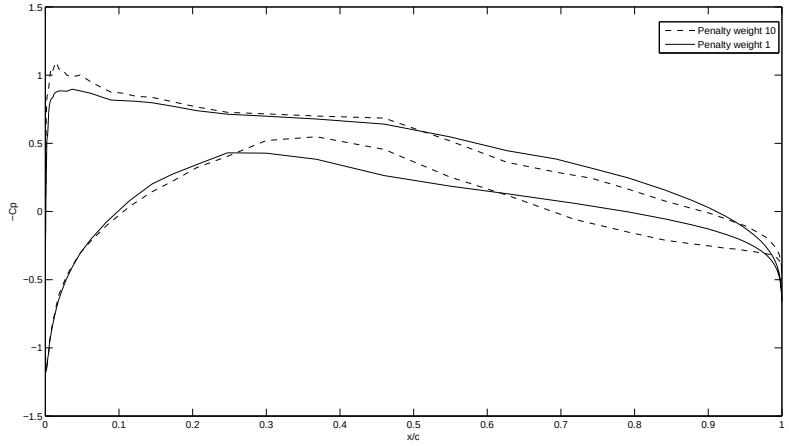


Figure 7.18: Thickness penalty weight impact on the Optimal pressure distribution of NACA 00083 using hybrid optimization scheme.

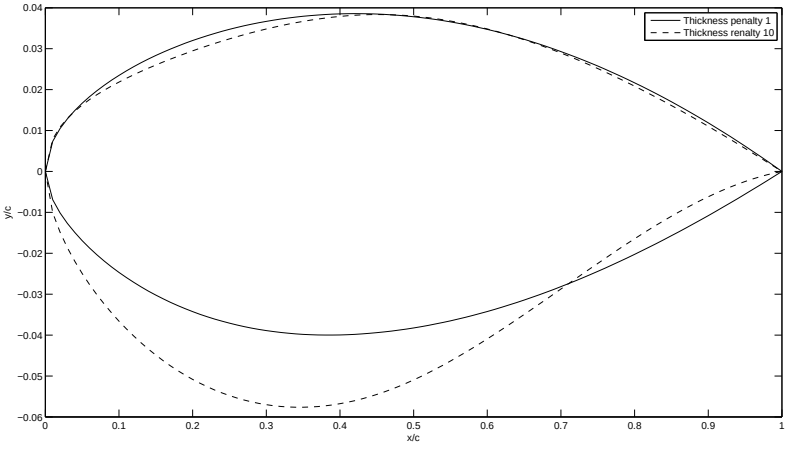


Figure 7.19: Thickness penalty weight impact on the Optimal profile with NACA 00083 starting geometry using hybrid optimization scheme.

Algorithm 7.3 Pseudo code of RegPSO-SQP "hybrid$^{\mathrm{T}}$" optimization scheme

1. Use the initial geometry as the initial global best $\overrightarrow{g}$.
2. Randomly generate swarm.
3. Add the initial geometry as a swarm member.
4. Use RegPSO strategy with stopping criterion of max 900 CFD simulations to find an estimated global best $\overrightarrow{g}$.
5. Use SQP with starting point $\overrightarrow{g}$ to find better global best solution.

7.5 Comparing SQP-RegPSO with RegPSO-SQP optimization strategies

I have presented results of the proposed hybrid optimization scheme in which SQP optimization is used to obtain a reasonably good upper bound on the global optimum. Based on the results of previous test cases, this scheme seems to be a promising global optimization scheme. Another hybrid scheme in which RegPSO precedes SQP optimization is possible. In this section I will compare the optimization results of the last five optimization problems using the proposed hybrid scheme with the results of RegPSO-SQP scheme; I will refer to the latter scheme as hybrid$^{\mathrm{T}}$. Algorithm 7.3 shows a pseudo code of hybrid$^{\mathrm{T}}$ optimization.

Both schemes has two phases, SQP phase and RegPSO phase, but they differ in the order of the phases. Tables 7.4 and 7.5 summarize the optimization results for both schemes. Figures 7.20–7.24 show a comparison of the obtained pressure distribution of the optimal shape obtained using these schemes. These comparisons show that the hybrid scheme is better than the hybrid$^{\mathrm{T}}$ for all cases; in Case II, the schemes are nearly identical in drag coefficient though different in final shape. This was expected because in hybrid$^{\mathrm{T}}$ if the first (RegPSO) phase could not reach the basin of attraction of the global minimum, the second (SQP) phase will not be able to find it. Tables 7.4, 7.5 show a comparison of Phase I and Phase II optimal objective function value for both hybrid and hybrid$^{\mathrm{T}}$ for the five test cases in this chapter. As a hybridization quality measure, the ratio of the optimal objective function phase

Case	Phase I Opt (SQP)	Phase II Opt (RegPSO)	Phase II/I
I[10]	0.0150	0.0009	0.0600
II[11]	0.0032	0.0009	0.2813
III [12]	0.0120	0.0013	0.1083
IV [13]	0.0040	0.0013	0.3250
V[14]	0.0040	0.0021	0.525

Table 7.4: Optimization results of the hybrid scheme in Phase I, II

Case	Phase I Opt (RegPSO)	Phase II Opt (SQP)	Phase II/I
I	0.0137	0.0049	0.358
II	0.0019	0.0011	0.579
III	0.0125	0.0125	1.0
IV	0.0017	0.0017	1.0
V	0.0052	0.0052	1.0

Table 7.5: Optimization results of the hybrid$^{\mathrm{T}}$ scheme in Phase I, II

II to Phase I optimal is used and the smaller this value the more successful the hybridization sequence is. The tables show also that the sequence used in hybrid is better than hybrid$^{\mathrm{T}}$.

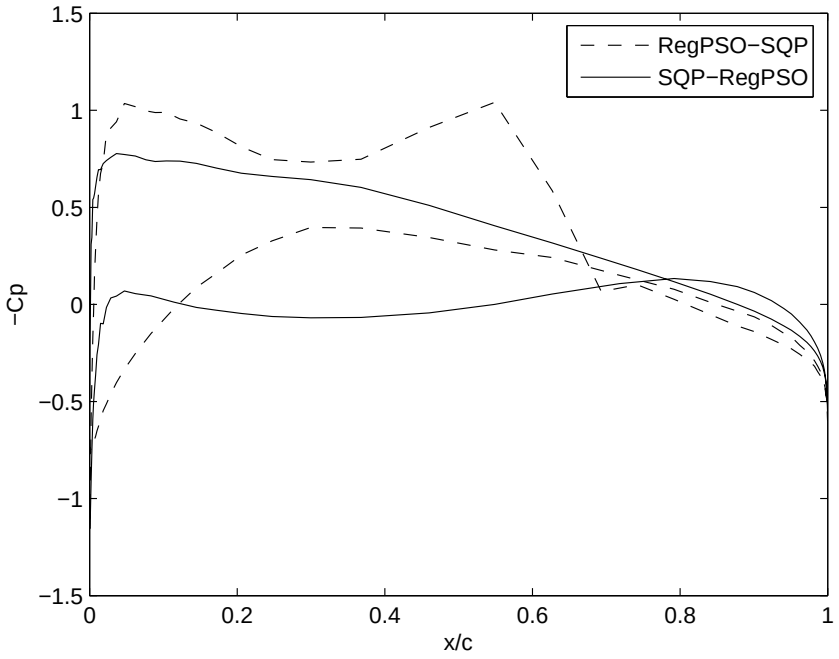


Figure 7.20: Hybrid and hybrid$^{\mathrm{T}}$ optimal pressure distribution for NACA 0012 (Case I).

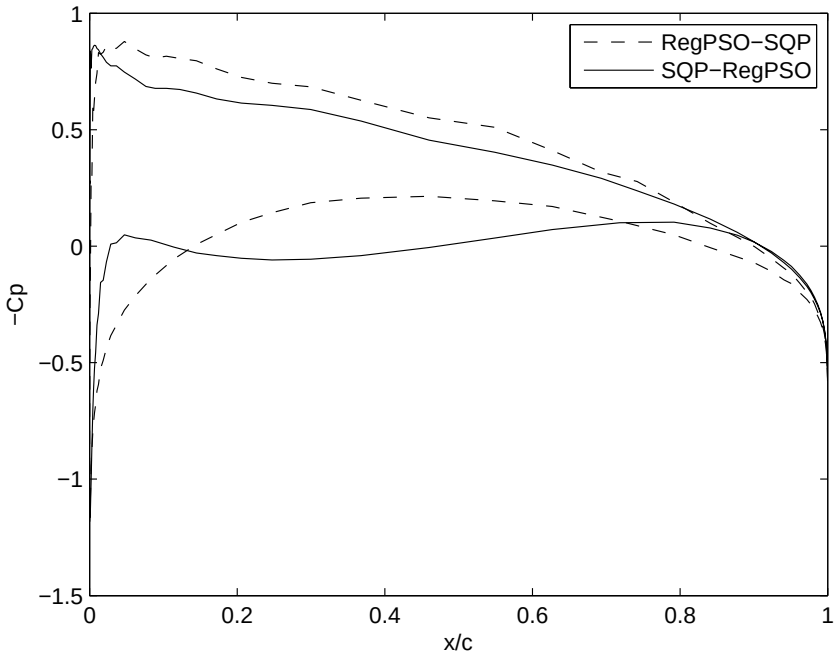


Figure 7.21: Hybrid and hybrid$^{\mathrm{T}}$ optimal pressure distribution for NACA 00083 (Case II).

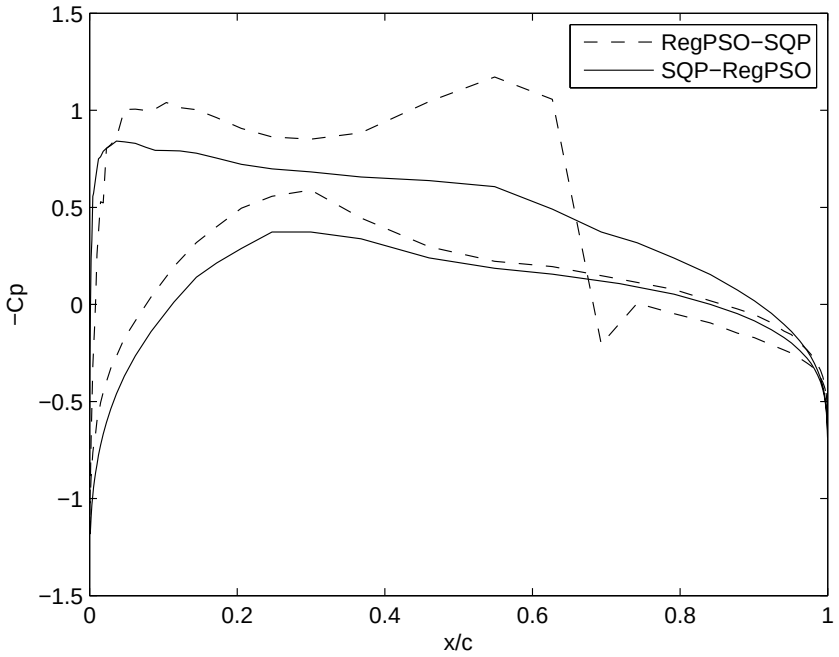


Figure 7.22: Hybrid and hybrid$^{\mathrm{T}}$ optimal pressure distribution for NACA 0012 with 10% thickness constraint and unit thickness penalty weight (Case III).

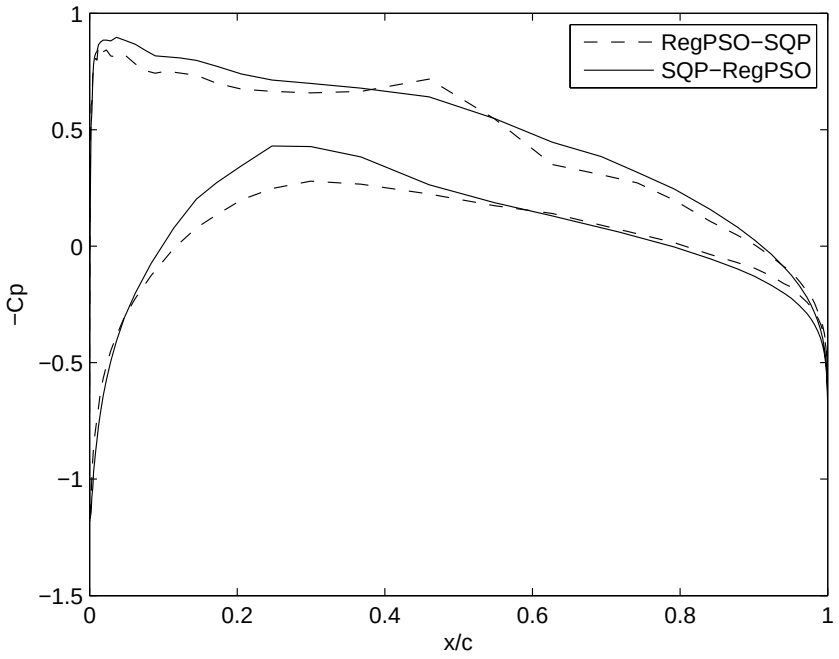


Figure 7.23: Hybrid and hybrid$^{\mathrm{T}}$ optimal pressure distribution for NACA 00083 with 10% thickness constraint and unit thickness penalty weight (Case IV).

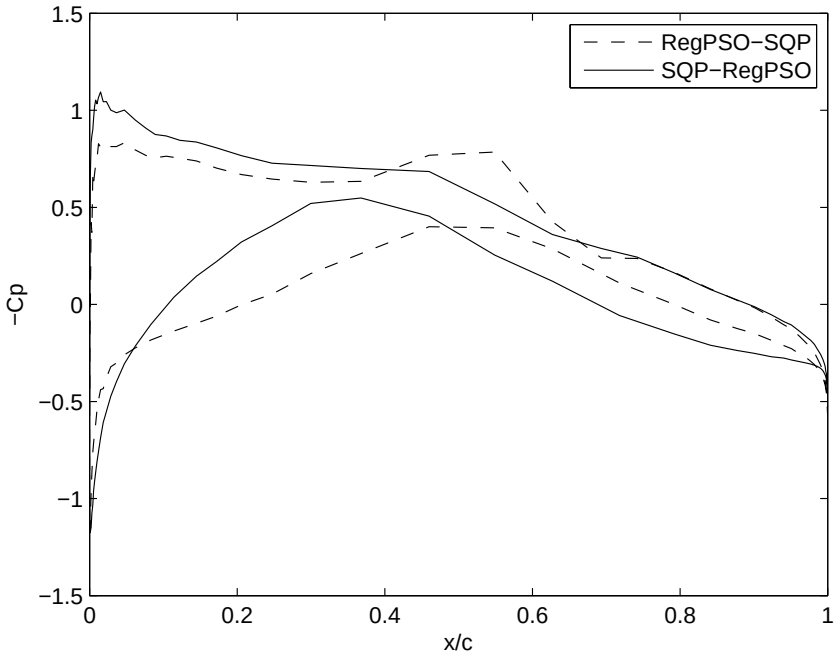


Figure 7.24: Hybrid and hybrid$^{\mathrm{T}}$ optimal pressure distribution for NACA 00083 with 10% thickness constraint and ten thickness penalty weight (Case V).

Chapter 8

CONCLUSIONS AND RECOMMENDATIONS

8.1 Contributions and Conclusions

This thesis presents the first study of the use of high order finite volume CFD methods in aerodynamic optimization. I have developed the first finite volume based optimization code that uses high order Jacobian in gradient computations using the adjoint technique. A new geometry parametrization technique has been developed and presented; this new technique is based on least squares surface fitting. To develop an efficient global optimizer, a new hybrid SQP-RegPSO scheme has been developed to find the global optimal solution in the feasible design space.

I presented both constrained and unconstrained aerodynamic optimization using sequential quadratic programming with the use of adjoint method to compute the objective function gradient. The computations were based on higher order finite volume method on unstructured grids. I took advantage of evaluating the exact Jacobian matrix to compute the flow adjoint field and flow solution sensitivity. The computed gradient based on the adjoint method is an excellent match with the corresponding finite difference gradient in subsonic flow for both second and fourth order schemes. For transonic flow, the second order Jacobian matches well with the corresponding order of accuracy finite difference gradient; on the other hand, the fourth order Jacobian does not match the finite difference gradient well when using a limiter. This error in the fourth order gradient is attributed to the high sensitivity of the fourth order Jacobian matrix to the use of the limiter in transonic flow which tends to affect the diagonal dominance of the Jacobian matrix. To tackle this problem and to enhance the diagonal dominance of the fourth order Jacobian matrix, we used the same non-zero structure of the second order Jacobian matrix

and dropped the rest of the non-zero elements; this tends to increase the diagonal dominance of the modified fourth order Jacobian matrix and improves its condition number.

I have developed a new approach for smooth parametrization of airfoil shapes, based on a least-squares fit of the parameters of two polynomials to a set of control points. Compared with using all mesh points as design variables, this approach reduces the size of the design space and eliminates oscillations in the shape. We use a semi-torsional spring analogy to deform the mesh grid in the entire flow field when the surface shape evolves during optimization. It is used also to calculate the mesh sensitivity terms in the adjoint gradient.

To test the developed gradient based optimizer; I have used the optimizer to tackle inverse design problems in which only one optimal solution point in the design space exists. Additional transonic drag minimization test cases have been presented with and without lift constraint. As a summary of test cases observations: A subsonic inverse design test case shows that both second and fourth order schemes reached their target geometry after almost the same number of iterations. The same behavior is observed for transonic inverse design test case. This indicates that the convergence is independent of the order of accuracy of the spatial discretization. In a drag minimization test case without lift constraint, both second and fourth order scheme reached their optimum shape after almost the same number of optimization iterations. The difference of the resulting optimal airfoil shape is small. Both schemes reduced the drag by almost 50% of its original value but the lift also went down.

In a drag minimization test case with lift constraint, the fourth order scheme was faster to reach its optimal shape with 8 optimization iterations (which cost 49 CFD simulations), while the second order scheme took 13 iterations (112 CFD simulations) to reach its optimum. Both schemes reached their optimum with almost the same wall clock run time and found nearly identical airfoil optimal shapes. Based on all the test cases we presented, we conclude that the spatial discretization error produces a systematic error in the objective function and has a little effect on the final optimal shape.

A mesh refinement study shows that both second and fourth order schemes tend to give the same optimal value for the objective function as we refine the mesh.

Based on the results of this research, I recommend the use of a high order method for objective function value computations and for accurate prediction of forces and shock wave capture in transonic flows, and the use of a second order method (but based on the obtained forth order solution) to compute the objective function gradient needed for gradient based optimizer. If the mesh resolution makes second and fourth order simulation results similar, using the second order scheme for optimization will be economical.

I have developed also a new hybrid gradient/non-gradient based optimization technique that uses sequential quadratic programming with BFGS Hessian approximation technique in an initial gradient based optimization phase, followed by the regrouped particle swarm optimization method in the non-gradient random search phase. The SQP optimization phase leads to a high quality initial best-solution point that reduces the total computational effort of the RegPSO to reach a shock free profile.

Unlike with the SQP scheme, the transonic drag optimization with lift constraint of the NACA 0012 and NACA 00083 tackled with the hybrid scheme lead to shock free optimal shapes; in both cases, the objective function gradient vector is almost perpendicular to the vector that connects the SQP optimum and the hybrid optimum. This explains why the gradient based optimization can not reach the global optimal, and also shows that its optimal solution depends on the starting geometry. The objective function behavior along the vector that connects the SQP and the hybrid optima shows that numerical noise may have a harmful impact on gradient based optimization as well.

Transonic drag optimization of NACA 0012 and NACA 00083 have been tackled with both aerodynamic and geometric constraints. For RegPSO, the geometric constraint is satisfied using penalty method; in both cases, shock free profiles have been obtained. The last test case (drag reduction of NACA 00083 with a thickness constraint) shows that the hybrid scheme was able to find a shock free profile despite *increasing* the thickness during optimization. The computational expense of the hybrid scheme is four to nine times more than SQP optimization, but in all cases, a global optimal (shock free) profile obtained, and in aerospace industry, this is worth the increase in computational effort.

8.2 Future Work

We intend to apply non gradient-based optimization techniques like particle swarm algorithm to study the effect of CFD scheme order of accuracy on the final optimal shape in the near future. Another obvious extension of the current work is aerodynamic optimization using RANS with turbulence models. The flow solver should solve the turbulence equations coupled with the mean flow equations; otherwise, the flow sensitivity values will be less inaccurate. Thus, for a three dimensional flow simulation with $k - \epsilon$ turbulence model, the block size of the Jacobian matrix is 7×7. Fully coupled solvers have also been shown to converge more efficiently.

For three-dimensional problems, the geometry parametrization should be replaced with a perturbation parametrization (parametrization of the change in the shape). This will reduce the required number of design variables which is very important for three dimensional optimization problems. Importantly, perturbation parametrization also prevents the change in the initial pressure distribution when parametrizing the geometry itself.

Exploring the effect of the geometry parametrization technique used on SQP optimization results is needed. We speculate that using the right geometry parametrization technique may lead to a design space in which aerodynamic constraints are simply connected and an SQP optimizer can find a globally optimal solution.

Bibliography

[1] Anderson, M.B. The Potential of Genetic Algorithms for Subsonic Wing Design. In *AIAA Paper 95-3925, presented at the 1st AIAA Aircraft Engineering, Technology, and Operations Congress, Los Angeles, CA*, 1995.

[2] Anderson, Murray B. Genetic Algorithms In Aerospace Design: Substantial Progress, Tremendous Potential. In *NATO/Von-Karmen-Institute Workshop on Intelligent System*, 2002.

[3] Anderson, W. Kyle, and Bonhaus, Daryl L. Aerodynamic Design on Unstructured Grids for Turbulent Flows. Technical report, NASA Langley Research Center, Hampton, Virginia, 1997.

[4] Avriel, M. *Nonlinear Programming Analysis and Methods.* Dover Publishing. ISBN 0-486-43227-0., 2003.

[5] Azab, M. B., and Ollivier-Gooch, C. F. Higher Order Two Dimensional Aerodynamic Optimization Using Unstructured Grids and Adjoint Sensitivity Computations. In *AIAA-2010-1430, 48th AIAA Aerospace sciences meeting, Orlando, FL, USA, 4-7Jan 2010*, Orlando FL, 2010.

[6] Azab, M. B., and Ollivier-Gooch, Carl. Constrained and Unconstrained Aerodynamic Quadratic Programming Optimization Using High Order Finite Volume Method and Adjoint Sensitivity Computations. In *49th Aerospace science meeting, Orlando FL, AIAA-2011-0183*, 2011.

[7] Batina, J. T. Unsteady Euler Airfoil Solutions Using Unstructured Dynamic Meshes. *AIAA Journal ,*, 28 No 8.:1381–1388, 1990.

[8] Beall, M. W., Walsh, J., and Shephard, Mark S. Accessing CAD Geometry for Mesh Generation. In *Proceedings of the 12th International Meshing Roundtable, Sandia National Laboratories*, pages 2003–3030, 2003.

[9] Bergh, F., and Engelbrecht, A. A New Locally Convergent Particle Swarm Optimiser. In *Conference on Systems, Man and Cybernetics*, 2002.

[10] Borrel, P., and Rappoport, A. Simple Constrained Deformations for Geometric Modeling and Interactive Design. *ACM Transactions on Graphics*, 13:137–155, 1994.

[11] Brezillon, J. and Gauger, N. R. 2D and 3D Aerodynamic Shape Optimisation Using the Adjoint Approach. *Aerospace Science and Technology*, 8:715–727, 2004.

[12] Broyden, C. G. The Convergence of a Class of Double-rank Minimization Algorithms. *Journal of the Institute of Mathematics and Its Applications*, 6:76–90, 1970.

[13] Byrd,R. H., Lu, P., and Nocedal, J. A Limited Memory Algorithm for Bound Constrained Optimization. *SIAM Journal on Scientific and Statistical Computing*, 16, 5:1190–1208., 1995.

[14] Cavallo, P. A., Hosangadi, A., Lee, R. A., and Dash, S. M. Dynamic Unstructured Grid Methodology with Application to Aero/Propulsive Flow Field. In *AIAA Paper 1997-2310*, 2002.

[15] Chandrashekarappa, P., and Duvigneau, R. Metamodel-Assisted Particle Swarm Optimization and Application to Aerodynamic Shape Optimization. Research Report RR-6397, INRIA, 2007.

[16] Clerc, M., and Kennedy, J. The Particle Swarm - Explosion, Stability, and Convergence in a Multidimensional Complex Space. *Evolutionary Computation, IEEE Transactions on*, 6(1):58–73, 2002.

[17] Consentino, G., and Holst, T. Numerical Optimization Design of Advanced Transonic Wing Configurations. In *AIAA 85-0424*, 1985.

[18] Cook, P.H., McDonald M.A., and Firmin, M.C.P. . Aerofoil RAE 2822: Pressure Distributions, and Boundary Layer and Wake Measurements. In *Experimental Data Base for Computer Program Assessment, AGARD Report AR 138.* AGARD, 1979.

[19] Dadone, Andrea, Mohammadi, Bijan, and Petruzzelli, Nicola. Incomplete Sensitivities and BFGS Methods for 3D Aerodynamic Shape Design. Research Report RR-3633, INRIA, 1999. Projet M3N.

[20] Duvigneau, R. Aerodynamic Shape Optimization with Uncertain Operating Conditions using Metamodels. Research Report RR-6143, INRIA, 2007.

[21] Eberhart, R. C., and Kennedy, J. A New Optimizer Using Particle Swarm Theory. In *Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, pp. 39-43.*, 1995.

[22] Engelbrecht, A. P. *Fundamentals of Computational Swarm Intelligence.* John Wiley and Sons, 2006.

[23] Evers, G. I., and Ben Ghalia, M. Regrouping Particle Swarm Optimization a New Global Optimization Algorithm with Improved Performance Consistency Across Benchmarks. In *SMC IEEE International Conference*, pages 3901–3908, oct. 2009.

[24] Evers, George. An Automatic Regrouping Mechanism to Deal with Stagnation in Particle Swarm Optimization. Master's thesis, University of Texas-Pan American, 2009.

[25] Farhat, C., Degand, C., Koobus, B., and Lesoinne, M. Torsional Springs for Two-Dimensional Dynamic Unstructured Fluid Meshes. *Computer methods in applied mechanics and engineering*, 163(1-4):231–245, 1998.

[26] Frink, N. T. *Three-Dimensional Upwind Scheme for Solving The Euler Equations on Unstructured Tetrahedral Grids.* PhD thesis, Virginia Polytechnic Institute and State University, 1991.

[27] Frink, N. T., Parikh, P., and Pirzadeh, S. A Fast Upwind Solver for The Euler Equations on Three-Dimensional Unstructured Meshes. In *AIAA paper 91-0102*, 1991.

[28] Gage, P., and Kroo, I. Development of the Quasi-Procedural Method for Use in Aircraft Configuration Optimization. In *AIAA Paper No. 92-4693, Presented at the Fourth AIAA/USAF/NASA/OAI Symposium on Multidisciplinary Analysis and Optimization*, 1992.

[29] Gage, P., and Kroo, I. A Role of Genetic Algorithms in a Preliminary Design Environment. In *AIAA Paper No. 93-3933*, 1993.

[30] Goldberg, David E. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Publishing Company, 1989.

[31] Goldfarb, D. Factorized Variable Metric Methods for Unconstrained Optimization. *Mathematics of Computation*, 30(136):796–811, 1976.

[32] Gregg, R.D., and Misegades, K.P. Transonic Wing Optimization Using Evolution Theory. In *presented at the AIAA 25th Aerospace Sciences Meeting, AIAA 87-0520*, 1987.

[33] Hart, Willian E. . *Adaptive Global Optimization with Local Search*. PhD thesis, University of California, San Diego, 1994.

[34] Hassan, R., Cohanim, A. , and de Weck,O. A Comparison of Particle Swarm Optimization and The Genetic Algorithm. In *AIAA-2005-1897*, page 1897, 2005.

[35] Hicks, R. M. , and Henne, P. A. . Wing Design by Numerical Optimization. *Journal of Aircraft*, 15:407–412., 1978.

[36] Ira H. Abbott and Albert Edward Von Doenhoff. *Theory of Wing Sections*. Dover, 1959.

[37] Jameson, A. Aerodynamic Design via Control Theory. *Journal of Scientific Computing*, 3:233–260, 1988. 10.1007/BF01061285.

[38] Jameson, A. A Perspective on Computational Algorithms for Aerodynamic Analysis and Design. *Progress in Aerospace Sciences*, 37(2):197–243, 2001. doi: DOI: 10.1016/S0376-0421(01)00004-5.

[39] Jameson, A. *Aerodynamic Shape Optimization Using The Adjoint Method*. VKI Lectur seriese, 2003.

[40] Jameson, A. Optimum Transonic Wing Design Using Control Theory. In *IUTAM Symposium held in Guttingen, Germany 2-6 September 2002*, page 253. Springer Netherlands, 2003. IUTAM Symposium Transsonicum IV: proceedings of the IUTAM Symposium held in Guttingen, Germany 2-6 September 2002.

[41] Jameson, A. , and Reuther, J. Control theory based airfoil design using Euler equations. In *AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 1994.

[42] Jameson, A., and Kim, S. Reduction of The Adjoint Gradient Formula in The Continuous Limit. *AIAA paper*, 40:2003, 2003.

[43] Jameson, A., Pierce,N. A., and Martinelli, L. Optimum Aerodynamic Design Using The Navier-Stokes Equations. In *35th AIAA Aerospace Sciences Meeting & Exhibit, Reno, NV*, 1997.

[44] Jang, M., and Lee, J. Genetic Algorithm Based Design of Transonic Airfoils Using Euler Equations. In *AIAA Paper 2000-1584, Presented at the 41st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials*, 2000.

[45] Karthik, M. *Application Of The Discrete Adjoint Method To Coupled Multidisciplinary Unsteady Flow Problems For Error Estimation And Optimization*. PhD thesis, Department of Mechanical Engineering, The University of Wyoming, 2009.

[46] Kennedy, J., and Eberhart, R. Particle Swarm Optimization. In *Proceedings of IEEE International Conference on Neural Networks. IV. pp. 1942-1948*, 1995.

[47] Kroo, I. Drag Due to Lift: Concepts for Prediction and Reduction. *Annual Review of Fluid Mechanic*, 33:587–617, 2001.

[48] Kulfan, B. M. Recent Extensions and Applications of the CST Universal Parametric Geometry Representation Method. In *7th AIAA Aviation Technology, Integration and Operations Conference*, 2007.

[49] Kulfan, B. M., Bussoletti, J. E. . Fundamental Parametric Geometry Representations for Aircraft Component Shapes. In *11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2006.

[50] Li, W. , Huyse, L. and Padula, S. Robust airfoil optimization to achieve drag reduction over a range of mach numbers. *Structural and Multidisciplinary Optimization*, 24:38–50, 2002. 10.1007/s00158-002-0212-4.

[51] Martineau, D. G., Georgala, J. M. A Mesh Movement Algorithm for High Quality Generalised Meshes. In *42nd AIAA Fluid Dynamics Conference and Exhibit, AIAA Paper 2004-0614, Reno, NV*, 2004.

[52] Masuda, H., Yoshioka, Y., and Furukawa, Y. Interactive Mesh Deformation Using Equality-Constrained Least Squares. *Computers and Graphics*, 30(6):936–946, 2006.

[53] Mengistu, Temesgen, and Ghaly, Wahid. Aerodynamic Optimization of Turbomachinery Blades Using Evolutionary Methods and ANN-Based Surrogate Models. *Optimization and Engineering*, 9:239–255, 2008. 10.1007/s11081-007-9031-1.

[54] Michalak, C. *Effcient High-Order Accurate Unstructured Finite-Volume Algorithms for Viscous and Inviscid Compressible Flows*. PhD thesis, The University of British Columbia, 2009.

[55] Michalak, C., and Ollivier-Gooch, C. Differentiability of Slope Limiters on Unstructured Grids. In *Fourteenth Annual Conference of the Computational Fluid Dynamics Society of Canada*, 2006.

[56] Michalak, C., and Ollivier-Gooch, C. Accuracy Preserving Limiter for The High-Order Accurate Solution of The Euler Equations. *Journal of Computational Physics*, 228(23):8693–8711, 2009.

[57] Michalak, Christopher, and Ollivier-Gooch, Carl. Globalized Matrix-Explicit Newton-GMRES for The High-Order Accurate Solution of The Euler Equations. *Computers and Fluids*, 39:1156–1167, 2010.

[58] Mousavi, A., Castonguay, P., and Nadarajah, S. K. Survey of Shape Parameterization Techniques and its Effect on Three-Dimensional Aerodynamic Shape Optimization. In *Proceedings of the AIAA 18th Computational Fluid Dynamics Conference*. AIAA 2007-3837, 25 - 28 June 2007 2007.

[59] Mouser, C. R., and Dunn, S. A. Comparing Genetic Algorithms and Particle Swarm optimisation for an Inverse Problem Exercise. In Rob May and A. J. Roberts, editors, *Proc. of 12th Computational Techniques and Applications Conference CTAC-2004*, volume 46, pages C89–C101, March 2005. `http://anziamj.austms.org.au/V46/CTAC2004/Mous` [March 21, 2005].

[60] Nadarajah, S., and Jameson, A. A Comparison of The Continuous and Discrete Adjoint Approach to Automatic Aerodynamic Optimization. *AIAA paper*, 667:2000, 2000.

[61] Nejat, A., and Ollivier-Gooch, C. A High-Order Accurate Unstructured Finite Volume Newton-Krylov Algorithm for Inviscid Compressible Flows. *Journal of Computational Physics*, 227(4):2582–2609, 2008.

[62] Nemec, M., and Aftosmis, M. J. Adjoint Sensitivity Computations for an Embedded-Boundary Cartesian Mesh Method. *Journal of Computational Physics*, 227(4):2724–2742, 2008.

[63] Nemec, M., and Zingg, D. W. . Newton-Krylov Algorithm for Aerodynamic Design Using the Navier-Stokes Equations. *AIAA Journal*, 40:1146–1154, 2002.

[64] Nielsen, E., and Anderson, K. Aerodynamic Design Optimization on Unstructured Meshes Using the Navier-Stokes Equations. Technical report, NASA Langley Technical Report Server, 1998.

[65] Nielsen, Eric J., and Kleb, Bil. Efficient Construction of Discrete Adjoint Operators on Unstructured Grids by Using Complex Variables. In *43rd AIAA Aerospace Sciences Meeting and Exhibit Reno, Nevada*, 2005.

[66] Nocedal, J., and Wright Stephen, J. *Numerical Optimization*. Springer New York, 2006.

[67] Obayashi, S. Genetic Algorithm for Aerodynamic Inverse Optimization Problems. In *Genetic Algorithms in Engineering Systems: Innovations and Applications*, 1995.

[68] Ollivier-Gooch, C. and Van Altena, M. A High-Order-Accurate Unstructured Mesh Finite-Volume Scheme for The Advection-Diffusion Equation. *Journal of Computational Physics*, 181(2):729–752, 2002.

[69] Ollivier-Gooch, C., Nejat, A., and Michalak, C. On Obtaining and Verifying High-Order Finite-Volume Solutions to the Euler Equations on Unstructured Meshes. *AIAA Journal*, 47(9):2105–2120, 2009.

[70] Oyama, A., Obayashi, S., and Nakahashi, K. Transonic Wing Optimization Using Genetic Algorithm. In *AIAA Paper 97-1854, 13th Computational Fluid Dynamics Conference*, 1997.

[71] Oyama, A., Obayashi, S., and Nakahashi, K. Fractional factorial design of genetic coding for aerodynamic optimization. In *AIAA Paper 99-3298, 3rd AIAA Weakly Ionized Gases Workshop*, 1999.

[72] Piegl, L., and Tiller, W. *The NURBS Book*. Springer, Berlin, 1995.

[73] Reuther, J., Jameson, A., Farmer, J., Martinelli, L., and Saunders, D. Aerodynamic Shape Optimization of Complex Aircraft Configurations via an Adjoint Formulation. In *34rd AIAA Aerospace Sciences Meeting and Exhibit, no. AIAA 1996-0094*, 1996.

[74] Roe, P. L. Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes. *Journal of Computational Physics*, 43(2):357–372, 1981. doi: DOI: 10.1016/0021-9991(81)90128-5.

[75] Saad, Youcef, and Schultz, Martin H. GMRES: a Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems. *SIAM J. Sci. Stat. Comput.*, 7:856–869, July 1986.

[76] Samareh, Jamshid A. A Survey of Shape Parameterization Techniques. In *CEAS/AIAA/ICASE/NASA Langley International Forum on Aeroelasticity and Structural Dynamics Williamsburg, VA, Also, NASA/CP-1999-209136, June 1999, pp. 333-343.*, 1999.

[77] Samareh, Jamshid A. Geometry and Grid/Mesh Generation Issues for CFD and CSM Shape Optimization. *Optimization and Engineering*, 6:21–32, 2005. 10.1023/B:OPTE.0000048535.08259.a8.

[78] Sederberg, Thomas W., and Parry, Scott R. Free-Form Deformation of Solid Geometric Models. *SIGGRAPH Comput. Graph.*, 20:151–160, August 1986.

[79] Shi, Y., and Eberhart, R. C. Empirical Study of Particle Swarm Optimization. In *Proceedings of Congress onEvolutionary Computation, CEC 99.*, volume 3, page 1950 Vol. 3, 1999.

[80] Stein, K., Tezduyar, T., and Benney, R. Mesh Moving Techniques for Fluid-Structure Interactions with Large Displacements. *J. Appl. Mech.*, 70:58–63, 2003.

[81] Tautges,Timothy J. The Common Geometry Module (CGM): A Generic, Extensible Geometry Interface. In *Proceedings of the 9th International Meshing Roundtable*, pages 337–348, 2000.

[82] Tezduyar, T. E., Behr, M., Mittal, S., and Johnson, A. A. Computation of Unsteady Incompressible Flows with The Stabilized Finite Element Methods: Space-Time Formulations, Iterative Strategies and Massively Parallel Implementations. *New Methods in Transient Analysis*, 246:0, 1992.

[83] Truong, Anh H., Oldfield, Chad A., and Zingg, David W. Mesh Movement for a Discrete-Adjoint Newton Krylov Algorithm for Aerodynamic Optimization. *AIAA Journal*, 46, issue 7:1695–1704, 2008.

[84] Venkatakrishnan, V. Convergence to Steady State Solutions of The Euler Equations on Unstructured Grids with Limiters. *Journal of Computational Physics*, 118(1):120–130, 1995.

[85] Venter, G., and Sobieszczanski-Sobieski, J. Multidisciplinary Optimization of a Transport Aircraft Wing Using Particle Swarm Optimization. *Struct Multidisciplinary Optimization*, 26:pp 121–131, 2004.

[86] Watt, A., and Watt, M. *Advanced Animation and Rendering Techniques.* Addison-Wesley Publishing Company, New York, 1992.

[87] Zhu, C., Byrd, R. H., Lu, P., and Nocedal, J. Algorithm 778: L-BFGS-B: Fortran Subroutines for Large-Scale Bound-Constrained Optimization. *ACM Transactions on Mathematical Software (TOMS)*, 23(4):550–560, 1997.

[88] Zymaris,A.S., Papadimitriou, D.I., Giannakoglou, K.C., and Othmer,C. Feasibility Study of Constant Eddy-Viscosity Assumption in Gradient-Based Design Optimization. *Journal of Computational Physics*, 229:5228–5245, 2010.