RIACS

NASA-CR-201064

60641

Research Institute for Advanced Computer Science
NASA Ames Research Center

Aerodynamic Shape Optimization Using Control Theory

James Reuther

RIACS Technical Report 96.09 May 1996

Published as Ph.D. Dissertation, University of California Davis, Davis CA 95616, April 1996.

Aerodynamic Shape Optimization Using Control Theory

James Reuther

The Research Institute of Advanced Computer Science is operated by Universities Space Research Association, The American City Building, Suite 212, Columbia, MD 21044, (410) 730-2656

Work reported herein was sponsored by NASA under contract NAS 2-13721 between NASA and the Universities Space Research Association (USRA).

Aerodynamic Shape Optimization Using Control Theory

By

James John Reuther

B.S. (University of California, Davis) 1989
M.S. (University of California, Davis) 1991

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Engineering

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Mohamed Hafez

[illegible]

[illegible]

[illegible]

Committee in Charge

1996

Abstract

Aerodynamic shape design has long persisted as a difficult scientific challenge due its highly nonlinear flow physics and daunting geometric complexity. However, with the emergence of Computational Fluid Dynamics (CFD) it has become possible to make accurate predictions of flows which are not dominated by viscous effects. It is thus worthwhile to explore the extension of CFD methods for flow analysis to the treatment of aerodynamic shape design.

Two new aerodynamic shape design methods are developed which combine existing CFD technology, optimal control theory, and numerical optimization techniques. Flow analysis methods for the potential flow equation and the Euler equations form the basis of the two respective design methods. In each case, optimal control theory is used to derive the adjoint differential equations, the solution of which provides the necessary gradient information to a numerical optimization method much more efficiently then by conventional finite differencing. Each technique uses a quasi-Newton numerical optimization algorithm to drive an aerodynamic objective function toward a minimum. An analytic grid perturbation method is developed to modify body fitted meshes to accommodate shape changes during the design process. Both Hicks-Henne perturbation functions and B-spline control points are explored as suitable design variables. The new methods prove to be computationally efficient and robust, and can be used for practical airfoil design including geometric and aerodynamic constraints. Objective functions are chosen to allow both inverse design to a target pressure distribution and wave drag minimization. Several design cases are presented for each method illustrating its practicality and efficiency. These include non-lifting and lifting airfoils operating at both subsonic and transonic conditions.

Contents

accounts and provided me with a conducive and supportive research environment at Ames without which I could not have succeeded in this effort. Mr. Bencze deserves the greatest recognition and respect from fellow scientists for his unwavering campaign to continue to fund and promote important research in the field of aeronautics. I would also like to thank Mr. David Saunders, Ms. Susan Cliff, Dr. John Gallman, Dr. Kalpana Chawla and Professor Joe Oliger for their support and assistance from NASA Ames Research Center and RIACS.

Although there are many other people that have also contributed to the completion of this research, those warranting special recognition are my fellow students and close friends at both universities. These include but are not limited to: Dr. Todd Mitty, Mr. Juanjo Alonso, Mr. David Banks, and Dr. David Kinney. Most important amongst my student friends has been Dr. James Farmer who spent many patient hours assisting me in this research.

Finally, I owe a great deal to my friends and family outside of the technical arena for withstanding the rigors with me during this effort. Special thanks in this regard goes to Ms. Mary Kate Horrigan, and Mr. John Lyons.

Acknowledgments

The work contained within this thesis has been accomplished through the support and assistance of individuals at both Princeton University and the University of California, Davis, as well as at NASA Ames Research Center.

First and foremost I am deeply indebted to Professor Antony Jameson, my thesis advisor from Princeton University. I am grateful to him not only for accepting me as one of his graduate students but also for his continual support and assistance during the course of this research, and for providing his flow analysis and adjoint based design codes that served as a starting point for this work. I also owe deep gratitude to the other members of Professor Jameson's research group, Dr. Timothy J. Baker and Professor Luigi Martinelli, who supported and encouraged both this endeavor and my gratifying one year stay as a visiting student at Princeton University.

At the University of California at Davis I owe much to Professor Mohamed Hafez who not only gave me my early inspiration within this field of research, but also has continually prompted me to engage in more challenging research. Much thanks also go to Professor Cornelis P. van Dam who advised my Masters thesis, suggested I continue research in aerodynamic design, and is a dissertation reader for this thesis. I also thank my other dissertation reader, Professor Jean J. Chattot.

Instrumental in the completion of this work has been the unwavering support and funding that has come from NASA Ames Research Center over the past several years. Very special thanks go to my mentor at Ames, Mr. Raymond Hicks who has given me constant supervision and education. It was through his suggestions that I was eventually led to the pursuits contained within this work. I feel deeply fortunate to have worked with Ray for all these years since it is quite evident that he stands out as one of the greatest practical aerodynamicists of our age. I would further like to pay special thanks to Mr. Daniel Bencze who has defended my research on many

List of Figures

List of Tables

Nomenclature

English Symbols

A	any symmetric positive definite matrix
A_{11}, A_{12}, A_{22}	components of $\mathcal{A}$
$\mathbf{A}_{11}, \mathbf{A}_{12}, \mathbf{A}_{22}$	linearized coefficients of A_{11}, A_{12}, A_{22}
$\mathbf{A}_1, \mathbf{A}_2$	flux Jacobians for the Euler equations
$A^{(\xi)}, \bar{A}^{(\eta)}$	terms of the rotated difference operator
$\bar{A}$	wave propagation matrix
$\mathcal{A}$	coordinate transformation matrix
b	an arbitrary vector
$\hat{b}$	band width of global flux Jacobian $\left[\frac{\partial R}{\partial w}\right]$
$\mathbf{b}$	vector of design variables, also $\mathbf{u}$
$\bar{\mathbf{b}}$	Bézier control points
B	outer boundary
c	speed of sound
c_a	speed of sound calculated at the outer boundary
c_e	speed of sound extrapolated from the flow domain
c_∞	speed of sound in the freestream
$\mathbf{c}$	arbitrary constant
c_p	constant pressure specific heat
c_v	constant volume specific heat
$\hat{c}$	chord length
C_a	coefficient of axial force
C_n	coefficient of normal force
C_d	coefficient of drag
C_l	coefficient of lift

C_m	coefficient of pitching moment
C	surface boundary
C_{1-3}	frozen coefficients of an Lapacian-like operator
$\mathbf{C}$	contravariant flux Jacobian matrices
$\mathbf{d}^2$	1st order dissipation flux
$\mathbf{d}^4$	3rd order dissipation flux
D	domain of the flow solutions
$\hat{\mathcal{D}}$	diagonal factorization update matrix
$\tilde{\mathcal{D}}$	diagonal factorization of $\tilde{\mathcal{H}}$
$\mathcal{D}$	the total discrete dissipation for the Euler equations
$\mathcal{D}^2$	2nd order discrete dissipation
$\mathcal{D}^4$	4th order discrete dissipation
$\mathbf{e}$	error term of the optimization algorithm
E	total energy
f	arbitrary function
$\mathbf{f}, \mathbf{g}$	Cartesian flux components of the Euler equations
F, G	contravariant Euler fluxes
$\mathbf{F}$	Euler flux matrix
$\mathcal{F}$	boundary control vector
g_μ	a Gaussian distribution to limit adjoint boundary terms through shock structures
$\mathcal{G}$	gradient of the cost function with respect to the control function or design variables
h	modulus of the conformal mapping function
$\bar{h}$	step size taken for gradient calculations
$\hat{h}$	characteristic length
H	total enthalpy
H_∞	total freestream enthalpy
$\mathcal{H}$	Hessian matrix
$\tilde{\mathcal{H}}$	approximate Hessian

i, j	indices of the computational domain
I	cost function
$\mathcal{I}$	identity matrix
$\mathcal{I}_f^c$	collection transfer operator from fine to coarse meshes
$\mathcal{I}_c^f$	interpolation transfer operator from coarse to fine meshes
J	determinant of the coordinate transformation
$\mathbf{J}$	linearized coefficients of J
K	coordinate transformation matrix
$\mathcal{K}$	the number of previous updates that are stored in a limited memory quasi-Newton method
$\bar{L}$	a characteristic length
L	operator for the variational of the potential flow equation
$\mathcal{L}$	discrete operator for the potential equation
$\bar{\mathcal{L}}$	an arbitrary linear difference operator on a fine mesh
$\hat{\mathcal{L}}$	lower triangular factorization update matrix
$\tilde{\mathcal{L}}$	lower triangular factorization of $\tilde{\mathcal{H}}$
m	order of a Bézier curve
$\bar{m}$	number of design points
M	Mach number
$\mathcal{M}$	coefficients of the discrete adjoint equation fluxes
M_c	critical Mach number
M_∞	freestream Mach number
n	time step parameter
$\bar{n}$	number of design variables
$\bar{n}_x, \bar{n}_y$	Cartesian components of the unit normal
$\bar{n}_\xi, \bar{n}_\eta$	computational coordinate components of the unit normal
$\hat{n}$	number of unknowns in the flow field
$\mathbf{n}$	unit normal vector
$\mathbf{n}_t$	unit tangent vector

$\mathcal{N}_e$	integrals along projection lines from the surface contributing to the Euler adjoint gradient
$\mathcal{N}_p$	integrals along projection lines from the surface contributing to the potential flow adjoint gradient
p	pressure
p_d	desired pressure
$\mathbf{p}$	optimization search direction and distance ($\lambda\mathbf{s}$)
$\tilde{P}, \tilde{Q}$	terms containing only variation in the mesh metrics for the potential equation
P, Q	dissipation flux terms for the potential flow equation
P_ψ, Q_ψ	dissipation flux terms for the potential flow adjoint equation
$\bar{P}, \bar{Q}$	linearized approximations of P and Q
$\hat{P}, \hat{Q}$	unswitched P and Q
$\hat{P}_\psi, \hat{Q}_\psi$	unswitched P_ψ and Q_ψ
$\mathcal{P}_p$	potential flow residual correction for coarse meshes
$\mathcal{P}_e$	Euler equation residual correction for coarse meshes
q	speed, magnitude of velocity
q_d	desired speed
q_C	speed at the surface
q_0	zero lift surface speed
q_∞	speed at far-field
$\mathbf{q}$	heat flux vector
$\mathcal{Q}$	discrete Euler residual
r	forcing function
$\mathbf{r}$	vector to define Hessian updates
$\tilde{\mathbf{r}}$	an arbitrary vector to define Hessian updates
R	arbitrary governing equation
R_e	Riemann invariant extrapolated from the flow domain
R_∞	Riemann invariant from the freestream
$\mathbf{R}$	gas constant

$\mathcal{R}$	arc length along η lines normalized to 1 at the surface and 0 at the far field
$\boldsymbol{\mathcal{R}}$	discrete Euler residuals including artificial dissipation
$\bar{\boldsymbol{\mathcal{R}}}$	smoothed $\boldsymbol{\mathcal{R}}$
s	arc length along airfoil surface
$\mathbf{s}$	optimization search direction
$\mathbf{S}$	surface projected area – directed face areas
$\mathbf{S}_x, \mathbf{S}_y$	components of the directed face areas
$\mathcal{S}$	control volume surface
$\mathcal{S}$	unit control volume surface
t	time
$\mathbf{t}$	surface traction vector
$\hat{T}$	a characteristic time
T	temperature
$\mathbf{T}$	stress tensor
$\mathbf{T}_0$	eigenvector matrix of the linearization of $\frac{\partial G}{\partial \mathbf{w}}$
u, v	Cartesian velocity components
$\mathbf{u}$	design variable vector
U, V	contravariant velocity components
$\mathcal{U}$	quasi-Newton update matrix
$\mathbf{v}$	velocity vector
$\mathbf{v}_a$	velocity vector calculated at the outer boundary
$\mathbf{v}_e$	velocity vector extrapolated from the flow domain
$\mathbf{v}_\infty$	velocity vector in the freestream
$\mathbf{V}$	discrete control volume
$\mathcal{V}$	control volume domain
$\mathcal{V}$	unit control volume domain
w	arbitrary flow field variable
$\mathbf{w}$	vector of Cartesian Euler unknowns
W	vector of contravariant Euler unknowns

x, y	Cartesian coordinate system
$\mathcal{X}$	mesh point location vector
$\mathbf{y}$	difference in the gradient vector between design cycles
$\hat{\mathbf{y}}$	transformed $\mathbf{y}$
$\mathbf{z}$	vector to define the Hessian update
$\tilde{\mathbf{z}}$	vector to define the Hessian update orthogonal to $\mathbf{s}$
$\bar{\mathbf{z}}$	scaled $\mathbf{z}$
$\hat{\mathbf{z}}$	transformed $\bar{\mathbf{z}}$
$\mathcal{Z}$	difference operator for the alternating direction implicit scheme

GREEK SYMBOLS

α	angle of attack
$\bar{\alpha}_0, \bar{\alpha}_1, \bar{\alpha}_2$	coefficients of a generalized alternating direction scheme
$\tilde{\alpha}$	control parameter for enthalpy damping
β	coefficient to subtract out non-A-orthogonal components of an arbitrary vector
$\beta_0, \beta_1, \beta_2$	terms related to $\bar{\alpha}_0, \bar{\alpha}_1, \bar{\alpha}_2$ for the equivalent time dependent scheme
δ	first variation operator
δ	differencing operator
$\bar{\delta}_{\xi}, \delta_{\bar{\eta}}$	one-sided difference operators
$\tilde{\delta}$	variations independent of variations in α
$\hat{\delta}$	correction operator
δ	step size in the optimization algorithm to eliminate the error component
Δt	time scale estimate
$\bar{\Delta} t$	(time scale estimate) x (CFL) condition number
ϵ_i, ϵ_j	control parameters for residual smoothing
$\varepsilon^2, \varepsilon^4$	coefficients of the dissipation fluxes
γ	ratio of specific heats c_p/c_v
$\bar{\gamma}_1, \bar{\gamma}_1$	scale factors for the rank two Hessian update components
Γ	circulation

∇	grad operator
κ_x, κ_y	components of the wave propagation direction
$\boldsymbol{\kappa}$	wave propagation direction
λ	Courant-Friedrich-Lewey (CFL) condition number
$\lambda_1 - \lambda_4$	eigenvalues of $\tilde{A}$
$\tilde{\lambda}$, $\tilde{\lambda}_1$, $\tilde{\lambda}_2$	ratio of accuracy of the cost function to the accuracy of the flow or adjoint convergence
$\bar{\lambda}$	step size along the optimization search direction
Λ_1, Λ_2	weights for combined cost functions
$\hat{\mu}$	dissipation coefficient for the potential equation
μ^2, μ^4	user-defined scale factors for 2nd and 4th order dissipation
ν	normalized second derivative of pressure used to limit dissipation
$\bar{\nu}$	averaging operator
ψ	adjoint variable (scalar) for the potential flow formulation
$\boldsymbol{\psi}$	adjoint variable (vector) for the Euler formulation
π	pi
ϕ	velocity potential
ϕ_s	velocity along the surface $- \frac{\partial \phi}{\partial s}$
ϕ_E	circulation potential
ϕ_G	perturbation potential
ϕ_{U_∞}	freestream potential
$\boldsymbol{\Phi}$	mesh metric relation
ω	relaxation factor for the alternating direction implicit scheme
Ω	factor in the adjoint equation boundary forcing term to specify fixed lift mode
ρ	density
ρ_1	residual of discrete flow field equation
ρ_2	residual of discrete adjoint field equation
ϱ	characteristic scaling quantity
σ	coefficient of the rotated difference operator

ς	small positive constant
θ	angle around the unit circle
Θ	smoothed adjoint boundary forcing variable
$\tau_0 - \tau_5$	pseudo time stages of the Runge-Kutta like scheme
υ_1, υ_2	smoothing parameters for the boundary forcing terms of the adjoint equation
Υ	first stage of ADI scheme
ξ, η	computational coordinate system
ξ_l, η_l	local cell computational coordinate system

Chapter 1

INTRODUCTION

Even before the success of the first powered flight at the turn of the 20th century, the importance of aerodynamic design was realized, leading to the introduction of wind tunnels in 1884. Since that time aerodynamicists have sought to develop better tools to facilitate the aerodynamic design process. The recent advances in computer technology have opened up the possibility for developing new methods to treat the problem of aerodynamic shape design. However, even with today's computer systems, detailed aerodynamic design has proven to necessitate a balance between the need for an accurate representation of the physical phenomena and limitations in the available computational resources.

1.1 PROBLEM STATEMENT

The goal of all aerodynamic design methods, be they experimental, analytical, or computational, is to find a shape which improves an aerodynamic measure of merit while adhering to appropriate constraints. The particular goal of this research is the development of accurate, efficient and versatile computational tools capable of automated design of aerodynamic shapes subject to both geometric and aerodynamic constraints. The two-dimensional design of airfoil sections has been selected as a representative problem to test alternative approaches. This still allows much variation in the possible methods and their capabilities. These capabilities can be classified by the level of flow physics that is used in the design process:

1. Viscous Methods

- Compressible Navier-Stokes equations
- Incompressible Navier-Stokes equations
- Euler + boundary layer equations
- Potential flow + boundary layer equations
- Small disturbance equation + boundary layer equations
- Linear potential flow equation + boundary layer equations

2. Inviscid Methods
   - Euler equations
   - Potential flow equation
   - Small disturbance equation
   - Linear potential flow equation

3. Source and Vortex Element Methods
   - Panel methods
   - Thin-airfoil theory

1.1.1 Group 1

The first group of analysis methods are all capable of treating some degree of viscous phenomena. For the design of airfoil sections subject to strictly subsonic flow, the ability to treat viscous effects becomes important since they dominate the production of drag seen as skin friction and pressure losses. Further, if it is desired to design airfoils for high lift, again the viscous effects must be given high consideration due to their determination of stall behavior.

The choice of which of the governing equations within this group to use for a viscous flow airfoil analysis or design is determined by a further understanding of the problem. If the problem consists entirely of low Mach number flow (<.2) then either the incompressible Navier-Stokes equations or the linear potential flow equation coupled with a boundary layer analysis can be used. The choice between these two is settled by the degree of coupling between the inviscid and viscous (boundary layer) portions of the flow. If coupling is pronounced, such as in flows that are dominated by large separation regions, then the incompressible Navier-Stokes equations should be used. For flows that exhibit compressibility effects, again either the compressible Navier-Stokes equations or one of the others remaining in group (1) must be used depending on the degree of inviscid-viscous coupling. If very strong coupling is not present, the choice can follow from the discussion to be presented for group (2).

1.1.2 GROUP 2

If the problem of interest is transonic cruise design, the main source of concern becomes wave drag. While the compressible Navier-Stokes equations may be used to analyze this problem it is often not critical to include viscous effects since they are of secondary consideration for determining cruise point wave drag. Group (2) above lists a hierarchy of inviscid methods with decreasing ability to treat compressibility effects. At the bottom of this list are the linear potential methods which have no such capability. The small disturbance equation is a nonlinear potential flow equation that corrects for compressibility effects provided these effects are small. It is usually appropriate for thin airfoils or bodies where local Mach numbers are only slightly greater than 1. Both the Euler and the potential flow equations remove these small disturbance restrictions and allow for arbitrary geometry and Mach numbers. The primary difference between the two is in their treatment of entropy and vorticity. The potential flow equation does not admit the production of either vorticity or entropy. Further, it does not allow for the convection of vorticity. The Euler equations on the other hand, properly allow for the production of entropy and vorticity along shock boundaries, as well as the convection of vorticity. These differences between the two have significant implications for when each system is appropriate and how each system must be formulated.

Independent of the production of vorticity through shock waves, the presence of vorticity in the flow domain is necessary in the case of two-dimensional airfoils for the Kutta condition to be enforced at the trailing edge. This problem is addressed in the potential flow formulation by enforcing a constant circulation in the flow field such that the Kutta condition is satisfied. In the case of the Euler equations, which allow for the convection of vorticity but not its production at the surface, the artificial viscosity required to obtain a stable solution to the discrete system is often enough to allow the Kutta condition to be satisfied as a by-product of the pseudo time dependent iteration process.

The difference in the treatment of entropy between the two may also affect both the results and the solution process. For viscous flows over an airfoil, the entropy production is limited to two regions: boundary layers and shock waves. Since both methods are inviscid, neither can account for the entropy production in the boundary layer. However, since the Euler equations capture the correct shock jump properties, they will predict the entropy production through the shocks and consequently the correct shock strengths. In contrast, the potential flow equation models shock waves as isentropic compressions and hence deviates from the correct solution as the shock strength increases. While corrections to this difficulty in the potential flow equation have been devised by Hafez [33], they are by no means easy to implement within the framework of design methods. In most potential flow methods the formulation is left as isentropic. Without this correction these methods can still be used with reasonable accuracy so long as the local Mach number of the flow around the airfoil does not exceed about 1.3. For flows with strong shocks where the isentropic assumption fails, the Euler equations are necessary to obtain accurate inviscid solutions.

The final group of methods (3) bridges the gap between analytical methods and the first computational methods. They are of historical and academic interest but are no longer used for transonic airfoil analysis, or design.

1.2 Developments in Aerodynamic Design

The success of powered flight resulted from the maturation of many required elements, including aerodynamics. The Wright brothers developed airfoils through the painful and laborious process of building and testing countless models [114]. Their eventual success caused the subject, referred to as "a dream of madmen" [73], to become a worthy scientific endeavor.

In the 1920s, fueled by the value of flight as a weapon of war, wind tunnel experiments as well as flight testing were conducted throughout the globe. In the United States, NACA—the National Advisory Committee for Aeronautics (precursor to NASA)—was formed. One of its initial goals was the systematic development of efficient airfoils through intensive wind tunnel testing.

Theoretical aerodynamics matured in parallel with experimental aerodynamics, and similarly gained acceptance through the success of powered flight. The first practical theories in aerodynamics were developed by Ludwig Prandtl and his co-workers, with their Thin Airfoil and Lifting Line Theories, both developed in the period 1912-1918 [82, 88]. Both of these theories were developed through the insight provided by the earlier work of Lanchester [73], who first proposed the idea of circulation and vortex shedding. Thin Airfoil Theory showed what experimenters already knew: airfoil sections have a lift curve slope close to 2π per radian. But this theory gave strong theoretical support to the early vague notions of how lift was generated. It also gave credibility to the aerodynamicist and placed him on the level of a true scientist and not a "madman." Lifting Line Theory was even more important. This theory fostered the development of such concepts as induced drag, wing efficiency, and optimal elliptic planforms. The aviation world quickly adopted these ideas, as was evidenced by the highly refined wing planforms of aircraft developed during the Second World War.

As aircraft became more refined, enhancing their aerodynamics grew more complex. The experimentalists built larger and more capable wind tunnels. With these new facilities, NACA followed its highly successful four-digit airfoils with first the five-digit airfoils and eventually the six-series airfoils. These latter airfoils were intended to take advantage of increased laminar flow to reduce drag. Starting in the 1950s work began in earnest to understand and develop aircraft capable of routine transonic and supersonic flight. At NASA Ames Research Center, the Unitary Wind Tunnel complex was built [84, 85] to provide experimentalists with the capability of studying aerodynamics over a large range of Mach numbers.

Since about 1960 there has been rapid progress in the field of CFD. Especially in the last decade with substantial improvements in both computer performance and numerical methods, CFD has been used extensively in parallel with experimental methods to aid in the aerodynamic design process. While much research continues in the CFD field, accurate and robust solutions for many flow conditions are now routinely obtained over complete aircraft configurations. Modern aircraft designers hope to benefit from this capacity in order to refine existing designs at transonic conditions and develop new designs at supersonic conditions. These highly nonlinear flow regimes require a design fidelity for which only CFD may provide the answers within practical time and cost constraints. Thus far, however, CFD—like wind tunnel testing—has not had as much success in direct aerodynamic shape design. Since the inception of CFD, researchers have sought not only to accurately predict the flow fields about given configurations, but also to formulate design methods capable of creating new optimum configurations. Yet while flow analysis can now be carried out over quite complex configurations using the Navier-Stokes equations with a high degree of confidence, direct CFD-based design is still limited to very simple two-dimensional and three-dimensional configurations, usually without including viscous effects. The CFD-based aerodynamic design methods that do exist can be grouped into three basic categories: inverse surface methods, inverse field methods, and numerical optimization methods. A brief review of these methods is presented in the next three sections with special emphasis on those that are capable of treating transonic and supersonic flows, where the previously-used analytic and linear methods are inadequate. The review is by no means complete but it indicates the current state of the science.

1.2.1 INVERSE SURFACE METHODS

Inverse surface methods derive their name from the fact that they invert the goal of the flow analysis algorithm. Instead of obtaining the surface distribution of an aerodynamic quantity, such as pressure, for a given shape, they calculate the shape for a given surface distribution of an aerodynamic quantity.

Lighthill solved the inverse surface pressure specification problem for a two dimensional profile in the presence of incompressible inviscid flow by conformal mapping [77].

The air speed over the profile is given by

$$q = \frac{\phi_\theta}{h}, \tag{1.1}$$

where ϕ is the velocity potential for flow past a circle and h is the modulus of the conformal mapping function between the circle and the profile. Since the solution, ϕ, is known for incompressible inviscid flow over a circle, knowing the analytic mapping exactly determines the solution over the profile. Conversely, by letting q_d be the desired surface speed, the value of h can be obtained by setting $q = q_d$ in equation (1.1). Furthermore, the mapping is analytic and is thus uniquely determined by the value of h on the boundary yielding the desired shape. Lighthill's method makes it clear that q may not be chosen entirely arbitrarily, but instead must satisfy constraints as follows:

$$\int_{-\pi}^{\pi} \log q_0 \, d\theta = 0 \tag{1.2}$$

$$\int_{-\pi}^{\pi} \log q_0 \, \cos\theta \, d\theta = 0 \tag{1.3}$$

$$\int_{-\pi}^{\pi} \log q_0 \, \sin\theta \, d\theta = 0, \tag{1.4}$$

where q_0 is the surface velocity distribution for the zero lift condition. The first constraint (1.2) is necessary for q to attain the free stream value q_∞ in the far field. The other two constraints (1.3,1.4) must be satisfied in order to produce a profile without a gap at the trailing edge.

To treat more complicated inverse surface problems, an iteration scheme is usually needed. For transonic potential flows, two basic iterative inverse surface methods exist. The first was pioneered by Tranen [110] who replaced the Neumann surface boundary condition in an existing CFD potential flow analysis code with a Dirichlet boundary condition obtained by integrating a desired target velocity distribution. The shape is then updated iteratively by the calculated normal velocity through the surface (transpiration). The approach can incur difficulties because of the difficulty in constructing a convergent iterative algorithm for the desired shape changes based on the Dirichlet boundary conditions at the surface. In particular, if the target pressure distribution is not realizable the iterations cannot converge. Nevertheless, Tranen's method has become accepted by the aeronautics community, especially with improvements developed by Volpe and Melnik [112, 113, 111] and through its extension to three dimensions by Henne [36] using the FLO22 wing analysis code of Jameson and Caughey [64]. All of these Dirichlet boundary condition methods require between 2-10 inverse design cycles in addition to a final analysis check to determine if the target has truly been achieved.

The second major technique solving the transonic surface quantity specification problem was proposed by McFadden and Garabedian, who essentially extended the method of Lighthill [81]. The iterations in this case are carried out by first solving the flow equation for a given initial mapping h_0. Then an updated mapping is determined by setting $q = q_d$ in equation (1.1). Again the flow equation is solved for this new mapping, h_1, and the process is repeated. In the limiting case of zero Mach number, the method reduces to Lighthill's method. The advantage of the method is that it does not require a modification to a Dirichlet boundary condition at the surface, and therefore retains a valid solution during the entire design process. McFadden gives a proof that the iterations will converge for small Mach numbers. A related method for three-dimensional design of wings was also devised by Garabedian and McFadden [24, 25]. In their scheme, the steady potential flow solution is obtained by solving an artificial time-dependent equation, and the surface is treated as a free boundary. This surface is shifted according to an auxiliary time dependent equation in such a way that the flow evolves toward the specified pressure distribution.

Since the development of these two standard transonic potential flow inverse surface methods, many combinations and variations have been developed which will not be reviewed here. These modified formulations either allow for somewhat greater geometric complexity, admit greater control over the eventual design, or even use different governing equations such as in the work of Fay [21] where the Euler equations were treated. Even though some of these methods have been successfully extended to limited three-dimensional applications, few have made a significant impact on actual aircraft design.

Another design method based on the Euler equations has been developed by Giles, Drela and Thompkins [27]. Their method solves the Euler equations in a stream-function-as-coordinate system. This implies that the coordinates (i.e., the streamlines) represent one of the unknowns in the solution process. The system has been formulated for two-dimensional transonic flow about an airfoil and uses an implicit Newton iteration scheme to achieve rapid convergence. The technique can be used in the analysis mode where the streamline of the surface is fixed. Alternatively, the method can be used in partial inverse mode, where part of the surface is left free and the pressure is prescribed as a boundary condition instead. The method has had dramatic success in the design of airfoils, especially since Drela and Giles have extended it to include an elegant coupling to a two-equation integral boundary layer method [17, 28]. Unfortunately, no clear-cut method exists to extend such a formulation to three dimensions.

Recently through the work of Campbell [13, 14], a simplified inverse method has shown much success. In his approach, the difference between the target and actual pressures is translated into surface changes through the use of the relationship between surface curvature and pressure for subsonic flow, and surface slope and pressure for supersonic flow. The iteration proceeds by coupling these relationships to any CFD algorithm and periodically updating the surface shape. The method is essentially a simplification of Garabedian and McFadden's original method that replaces surface updates developed by a mesh transformation with simple fixed relationships. Campbell has extended the method to treat flows subject to the Navier-Stokes equations in three dimensions with significant success [14]. The method is based on the assumption that the dependence of the local pressure on the shape satisfies simple fixed relationships. For three dimensions, where the surface curvatures and slopes are calculated plane by plane in the free stream direction, any cross-flow character in the solution could disrupt the design process.

1.2.2 INVERSE FIELD METHODS

An alternative means of obtaining desirable aerodynamic shapes is provided by the field-based class of inverse design methods. These methods differ from surface specification methods in that they obtain designs based upon objectives or constraints imposed not only upon the configuration surface but everywhere in the flow field. Most of these methods are based on potential flow techniques with only a few having been extended to three dimensions.

Possibly the most notable example of an inverse field method, which was first introduced by Garabedian and Korn [26], relies upon a hodograph transformation. The method can be regarded as an inverse field method since part of the design process guarantees that no shock will occur in the flow field. However, it also retains the ability to specify the target Mach number distribution on the geometry surface and thus must be considered a hybrid method. The technique has been used with striking success in the development of airfoils displaying shock-free transonic flows [2]. The technique is quite difficult and involves using a method of complex characteristics to solve the equations in the hodograph plane. Its most limiting feature is that hodograph transformations are not applicable to three dimensions.

One true field-based inverse method that also attempts to create airfoils which operate with shock-free transonic flow is the fictitious gas method developed by Sobieczky [105, 106]. This method can be incorporated into any potential flow solver. The basic idea is to solve the mixed elliptic/hyperbolic flow system on an existing geometry by a special procedure. The subsonic points are treated as usual with a standard potential flow method. However, for supersonic points, the isentropic pressure-density relation in potential flow is replaced by an appropriate analytic fictitious density relation. The resulting solutions are correct in the subsonic regions and incorrect in the supersonic regions. Next, the supersonic region (area under the sonic line umbrella) is re-solved in the rheograph plane with a method of characteristics. The boundary conditions for this recalculation are chosen so as to match the values at the sonic line boundary. The solution then determines the streamline locations in the supersonic region and therefore the position of the surface. The method is obviously not a complete geometry design method, but must be thought of as a redesign technique for existing geometries. It only ensures that the supersonic bubble will become stretched and shallow, but this may be enough to ensure an entirely shock-free design. The method has been extended to three dimensions with some degree of success [22, 89], but requires marching the solution of the supersonic zone transverse to the characteristic cone, which can lead to instability.

The characteristic common to all inverse methods is their computational efficiency. *Typically transonic inverse methods require the equivalent of 2-10 complete flow solutions in order to render a complete design. Since obtaining a few solutions for simple two-dimensional and three-dimensional designs can be done in at most a few hours on modern computers systems, the computational cost of most inverse methods is considered to be minimal.* Unfortunately, they suffer from many limitations and difficulties. Their most glaring limitation is that the objective of a target pressure distribution is built directly into the design process and thus cannot be changed to any arbitrary or more appropriate objective function. They cannot directly address other aerodynamic objective functions such as lift, drag, or pitching moment. The user must therefore be highly experienced in order to be able to prescribe surface distributions or choose initial geometries which lead to the desired aerodynamic properties. In addition, the target surface pressure distribution may not correspond to a physically realizable solution. Thus, surface inverse methods, with the exception of Campbell's work, have a tendency to fail because the target surface distribution does not satisfy the necessary constraints to permit the existence of the desired solution. On the other hand, field inverse methods typically only allow for the design of a single shock-free design point and have no means of properly addressing off-design points. Furthermore, it is difficult to formulate inverse methods that can satisfy the desired aerodynamic and geometric constraints. In essence, inverse methods require designers to have an *a priori* knowledge of an optimum pressure distribution that satisfies the geometric and aerodynamic constraints. This limited design capability and difficult implementation to date has restricted the applicability of inverse methods. An alternative approach which has proven to overcome some of the disadvantages of inverse methods at the price of computational expense is provided by the numerical optimization methods.

1.2.3 NUMERICAL OPTIMIZATION METHODS

The final major group of aerodynamic shape design methods is the group employing numerical optimization. The essence of these methods is very simple. A numerical optimization procedure is coupled directly to an existing CFD analysis algorithm. The numerical optimization procedure attempts to extremize a chosen aerodynamic measure of merit which is evaluated by the chosen CFD code. The optimization then proceeds by systematically modifying the configuration through user specified design variables. Design variables must be chosen in such a way as to permit the shape of the configuration to change in a manner that allows the design objective to be improved. Most of these optimization procedures require gradient information in addition to evaluations of the objective function. Here, the gradient refers to changes in the objective function with respect to changes in the design variables. The simplest method of obtaining gradient information is by *finite differences. In this technique, the gradient components are approximated by independently perturbing each design variable by a finite step, calculating the corresponding value of the objective function using CFD analysis, and forming the ratio of the differences.* These estimates of the partial derivatives form the gradient that is then used by the numerical optimization algorithm to calculate a search direction using steepest descent, conjugate gradient, or quasi-Newton techniques. The optimization algorithm then proceeds by estimating the minimum or maximum of the aerodynamic objective function along the search direction using repeated CFD flow analyses. The entire process is repeated until the norm of the gradient approaches zero or further improvement in the aerodynamic objective function appears impossible.

The use of numerical optimization for transonic aerodynamic shape design was pioneered by Hicks, Murman and Vanderplaats [39]. They applied the method to two-dimensional profile design subject to the potential flow equations. The method was quickly extended to wing design by Hicks and Henne [37, 38]. Recently, through the work of Reuther, Cliff, Hicks, and van Dam, the method has proven to be successful for the design of supersonic wing/body transport configurations by its extension to treat three-dimensional flows governed by the Euler equations [91]. In all of these cases, finite differences were used to obtain the required gradient information.

These methods are very versatile, allowing any reasonable aerodynamic quantity to be used as the objective function. They can be used to mimic an inverse method by minimizing the difference between target and actual pressure distributions, or they may be used to maximize other aerodynamic quantities of merit such as C_l/C_d. Geometric constraints can be readily enforced by a proper choice of design variables. Aerodynamic constraints can be treated either by adding weighted terms to the objective function or by the use of a constrained optimization algorithm. Unfortunately, these finite difference numerical optimization methods, unlike the inverse methods, are computationally expensive because of the large number of flow solutions needed to determine the gradient information for a useful number of design variables. For three-dimensional configurations, hundreds or even thousands of design variables may be necessary. This implies that tens of thousands of flow analyses might be required for a complete design.

1.3 Alternative Methods

Clearly, there is a need for alternative methods which have the flexibility and power of current numerical optimization codes but do not require their large demand on computational resources. These new methods must avoid the limitations and difficulties of traditional inverse methods while approaching their inherent computational efficiency.

1.3.1 Genetic Algorithms

An alternative approach which does not require gradient information is the Genetic Algorithm (GA) method [16], where models of evolution are applied to a population of designs. As this population evolves in time, only the "fittest" of the designs will proliferate. This technique also has the advantage of avoiding the pitfalls of gradient-based methods, such as arriving at a local minimum as opposed to a global minimum.

The disadvantage of such an approach is that evolution can still be extremely expensive because many analyses must still be performed before an optimum is found. The total CPU costs of GA methods tend to match or exceed current gradient-based methods.

1.3.2 AUTOMATIC DIFFERENTIATION

Another alternative is the application of automatic differentiation. Here, a preprocessor applies the chain rule on a line-by-line basis to an analysis code, generating new code for calculating analytic derivatives. While such a task would be daunting for a human, programs such as ADIFOR [8] can perform this operation with relative ease. Previous research efforts have applied automatic differentiation to the development of design methods utilizing aerodynamic analysis codes based on linear theory. This work has shown that the ADIFOR code can successfully differentiate analysis codes and that analytic derivative information can be readily incorporated into an optimization procedure. However, in certain circumstances the cost of utilizing ADIFOR may not be significantly less than that of finite-difference gradient calculations. For the results mentioned above, the ADIFOR-calculated derivatives cost approximately 1/3 as much as evaluating the derivatives by finite difference techniques. This significant time savings was not a direct result of the efficiency of ADIFOR but was attributed to multi-point fitting used in the finite difference method. These conclusions regarding the costs are supported by the work of Newman et al. [32, 102], where ADIFOR was applied to a thin layer Navier-Stokes code. The resulting derivatives were found to be very accurate (where finite-differenced values may not have been), but the costs were not significantly lower than those for the conventional approach. In any event, current releases of ADIFOR do not yield the orders of magnitude cost reductions that are needed.

1.4 CONTROL THEORY APPROACH

In this work the proposed solution to reduce the excessive CPU time associated with acquiring gradient information is to use methods based upon *control theory*. The idea draws from the theory of control systems governed by partial differential equations outlined by Lions [78]. It was applied to shape design for elliptic equations by Pironneau [86]. The use of control theory for transonic airfoil and wing design was first proposed by Jameson [54], who also demonstrated a numerical implementation for airfoil design using the transonic potential flow equation [55].

Suppose that the boundary is defined by a function $\mathcal{F}(\mathbf{b})$, where $\mathbf{b}$ is the position vector of the design variables, and the desired objective is measured by a cost function I. This may, for example, measure the deviation from a desired surface pressure distribution, but it can also represent other measures of performance such as lift and drag. Suppose that a variation $\delta\mathcal{F}$ in the control produces a variation δI in the cost. Following control theory, δI can be expressed to first order as an inner product

$$\delta I = (\mathcal{G}, \delta\mathcal{F}),$$

where the gradient $\mathcal{G}$ of the cost function with respect to the control is independent of the particular variation $\delta\mathcal{F}$, and can be determined by solving an adjoint equation. If one makes a shape change

$$\delta\mathcal{F} = -\varsigma\mathcal{G},$$

where ς is sufficiently small and positive, then

$$\delta I = -\varsigma(\mathcal{G}, \mathcal{G}) < 0 \tag{1.5}$$

assuring a reduction in I. The method can be accelerated by choosing $\delta\mathcal{F}$ not simply as a multiple of the gradient (steepest descent) but instead as a more sophisticated search direction provided by numerical optimization.

For flow about an airfoil or wing, the aerodynamic properties which define the cost function are functions of the flow field variables (w), the physical locations of the mesh points within the volume ($\mathcal{X}$), and the physical location of the boundary ($\mathcal{F}$). Then

$$I = I(w, \mathcal{X}, \mathcal{F}),$$

and a change in $\mathcal{F}$ results in a change

$$\delta I = \frac{\partial I^T}{\partial w}\delta w + \frac{\partial I^T}{\partial \mathcal{X}}\delta \mathcal{X} + \frac{\partial I^T}{\partial \mathcal{F}}\delta \mathcal{F} \tag{1.6}$$

in the cost function. As pointed out by Baysal and Eleshaky [5] each term in (1.6), except for δw, can be easily obtained. $\frac{\partial I}{\partial w}$, $\frac{\partial I}{\partial \mathcal{X}}$ and $\frac{\partial I}{\partial \mathcal{F}}$ can be obtained directly without a flow field evaluation because they are partial derivatives. $\delta\mathcal{F}$ is simply the surface modification and $\delta\mathcal{X}$ can be determined by either working out the exact analytical values from a mapping as in Jameson's implementations [54, 55], or by successive grid generation for each design variable, so long as this cost is significantly less than the cost of the flow solution. For solutions requiring a large number of mesh points where grid generation becomes expensive, an alternative method for calculating $\delta\mathcal{X}$ can be formulated using grid perturbation. Finite difference methods evaluate the gradient by making a small change in each design variable separately, then recalculating both the grid and flow field variables. This requires a number of additional flow calculations equal to the number of design variables. Using control theory, the governing equations of the flow field are introduced as a constraint in such a way that the final expression for the gradient does not require multiple flow solutions. In order to achieve this result, δw must be eliminated from (1.6). The residual of the governing equation, R, expresses the dependence of w, $\mathcal{X}$ and $\mathcal{F}$ within the flow field domain D,

$$R(w, \mathcal{X}, \mathcal{F}) = 0.$$

Thus δw is determined from the equation

$$\delta R = \left[\frac{\partial R}{\partial w}\right]\delta w + \left[\frac{\partial R}{\partial \mathcal{X}}\right]\delta\mathcal{X} + \left[\frac{\partial R}{\partial \mathcal{F}}\right]\delta\mathcal{F} = 0. \tag{1.7}$$

Next, introducing a Lagrange multiplier ψ, we have

$$\begin{aligned}\delta I &= \frac{\partial I^T}{\partial w}\delta w + \frac{\partial I^T}{\partial \mathcal{X}}\delta\mathcal{X} + \frac{\partial I^T}{\partial \mathcal{F}}\delta\mathcal{F} - \psi^T\left(\left[\frac{\partial R}{\partial w}\right]\delta w + \left[\frac{\partial R}{\partial \mathcal{X}}\right]\delta\mathcal{X} + \left[\frac{\partial R}{\partial \mathcal{F}}\right]\delta\mathcal{F}\right) \\ &= \left\{\frac{\partial I^T}{\partial w} - \psi^T\left[\frac{\partial R}{\partial w}\right]\right\}\delta w + \left\{\frac{\partial I^T}{\partial \mathcal{X}} - \psi^T\left[\frac{\partial R}{\partial \mathcal{X}}\right]\right\}\delta\mathcal{X} + \left\{\frac{\partial I^T}{\partial \mathcal{F}} - \psi^T\left[\frac{\partial R}{\partial \mathcal{F}}\right]\right\}\delta\mathcal{F}.\end{aligned} \tag{1.8}$$

Choosing ψ to satisfy the adjoint equation,

$$\left[\frac{\partial R}{\partial w}\right]^T\psi = \frac{\partial I}{\partial w}, \tag{1.9}$$

eliminates the first term of (1.8) resulting in

$$\delta I = \left\{ \frac{\partial I^T}{\partial \mathcal{X}} - \psi^T \left[\frac{\partial R}{\partial \mathcal{X}} \right] \right\} \delta \mathcal{X} + \left\{ \frac{\partial I^T}{\partial \mathcal{F}} - \psi^T \left[\frac{\partial R}{\partial \mathcal{F}} \right] \right\} \delta \mathcal{F}. \tag{1.10}$$

This can be written as

$$\delta I = \mathcal{G}^T \delta \mathcal{F} \tag{1.11}$$

where

$$\mathcal{G}^T = \left\{ \frac{\partial I^T}{\partial \mathcal{X}} - \psi^T \left[\frac{\partial R}{\partial \mathcal{X}} \right] \right\} \frac{\delta \mathcal{X}}{\delta \mathcal{F}} + \frac{\partial I^T}{\partial \mathcal{F}} - \psi^T \left[\frac{\partial R}{\partial \mathcal{F}} \right]. \tag{1.12}$$

The advantage is that (1.10) is independent of δw, with the result that the gradient of I with respect to an arbitrary number of design variables can be determined without the need for additional flow field evaluations. The main cost is in solving the adjoint equation (1.9). In general, the adjoint problem is about as complex as a flow solution. If the number of design variables is large, the cost differential between one adjoint equations solution and the large number of flow field evaluations required to determine the gradient by finite differences becomes compelling.

Instead of introducing a Lagrange multiplier, ψ, one can solve for δw directly from equation (1.7)

$$\left[\frac{\partial R}{\partial w} \right] \delta w = - \left\{ \left[\frac{\partial R}{\partial \mathcal{X}} \right] \delta \mathcal{X} + \left[\frac{\partial R}{\partial \mathcal{F}} \right] \delta \mathcal{F} \right\}, \tag{1.13}$$

and insert the result in (1.6). This is the implicit gradient or direct approach which is essentially equivalent to the control theory approach, as has been pointed out by Shubin and Frank [103, 104]. The difference between the direct and adjoint approaches lies in the differences in the right hand sides of (1.9) and (1.13). For the adjoint approach the right hand side depends upon the objective function(s), while in the direct approach the right hand side is a function of the design variables. If the inversion of $\left[\frac{\partial R}{\partial w}\right]$ or $\left[\frac{\partial R}{\partial w}\right]^T$ is calculated directly, the two methods become identical with only a different order of multiplication. However, if equation (1.9) or (1.13) is solved without direct inversion, say by an iterative procedure, then the two methods become quite different since each right hand side requires a separate solution. Thus, the determining factor on which to choose is then settled by examining the design problem. If the number of design variables is greater than the number of independent objective functions and constraints then the adjoint method prevails. The opposite is true if the number of design variables is low compared to the number of objectives and constraints.

Once the gradient is calculated by (1.12), a modification following (1.5) can then be made. After making such a modification, the gradient can be recalculated and the process repeated to follow a path of steepest descent until a minimum is reached. In order to avoid violating geometric constraints, such as a minimum acceptable airfoil thickness, the gradient may be projected into the allowable subspace within which the constraints are satisfied. In this way, procedures can be devised which must necessarily converge at least to a local minimum. The efficiency may be improved by performing line searches to find the minimum in a search direction defined by the negative gradient, and also by the use of more sophisticated descent methods such as conjugate gradient or quasi-Newton algorithms. There is the possibility of more than one local minimum, but in any case the method will lead to an improvement over the original design. Furthermore, unlike the traditional inverse algorithms, any measure of performance can be used as the cost function.

In this research the adjoint approach is used to permit a dramatic reduction in the computational cost of each design solution since the gradient cost will be reduced to the cost of approximately two flow evaluations (provided the adjoint equations are about as computationally expensive as the flow equations) instead of the traditional $\bar{n}+1$ evaluations where $\bar{n}$ is the number of design variables. The key point is that the cost of the optimization method is no longer proportional to the number of design variables, which has been the limiting factor in finite difference-based aerodynamic optimization methods.

Another significant advantage of the adjoint method is its applicability to multi-point design problems. Because of its efficient use of computer resources, two or three design points can be included in the optimization procedure by solving separate adjoint problems for each design point and then defining the total gradient as a weighted combination of the gradients for each of the individual design points. This will allow the performance benefits at various design points to be considered together, yielding a more optimal overall design. If the number of design points becomes larger than the number of design variables, the switch should be made to the direct method of equation (1.13) thus yielding a process requiring a number of solutions equal to the number of design variables. Even in this case a large benefit is still realized when compared with finite difference calculations where the number of solutions required to complete the gradient is $\bar{m}\bar{n}$, where $\bar{m}$ is the number of design points.

1.5 CONTINUOUS VS. DISCRETE SENSITIVITY ANALYSIS

By continuous sensitivities it is implied that control theory is applied directly to the partial differential equations governing the flow solution, thus forming the adjoint equations also as a system of partial differential equations. These adjoint differential equations are then discretized and solved in the same manner as the flow equations to obtain the necessary gradient information. This approach was first used for aerodynamic design in the presence of transonic flows by Jameson [54, 55]. The ideas presented in these early works have been independently verified by Lewis and Agarwal [75, 76, 74]. Jameson's initial formulations were derived in conjunction with analytic grid mappings to obtain, directly at the differential level, the necessary systems of equations defined by equation (1.12). Regularization procedures were introduced to ensure that the adjoint equations remained well posed despite the presence of discontinuities in the flow. The use of analytic mappings in Jameson's initial works restricted the technique in the past to relatively simple geometries such as airfoils and wings.

One may alternatively derive a set of discrete adjoint equations directly from the discrete approximation to the flow equations by following the procedure outlined in equations (1.6 – 1.13). The resulting discrete adjoint equations are one of the possible discretizations of the continuous adjoint equations. This approach (discrete sensitivity analysis) is now adopted in the work of Taylor, Newman, Hou, et al. [83, 70, 43, 68, 102] and also Baysal, Eleshaky and Burgreen [4, 5, 6, 11, 3, 18, 19, 10].

It seems that both alternatives have some advantages. The continuous approach gives the researcher some hope for an intuitive understanding of the adjoint system and its related boundary conditions. The discrete approach, in theory, maintains perfect algebraic consistency at the discrete level. If properly implemented it will give gradients which closely match those obtained through finite differences. The continuous formulation directly approximates the gradient corresponding to the differential equations, and incurs discretization errors which do not necessarily correspond to the discretization errors in the flow solution. Thus, it does not provide the exact gradient corresponding to the discrete flow equations. However, these discrepancies in the discretization errors must vanish in the limit as the mesh width is reduced. The discrete sensitivity approach in fact produces one of the possible discretizations of the continuous equations.

An advantage of the continuous formulation is that the discretization and iteration scheme used to solve the flow field system can be easily adapted for use in solving the adjoint system. Therefore, the robust iteration algorithms and convergence acceleration techniques that have been matured for CFD algorithms can be directly ported for the solution of the adjoint system.

The recycling of the flow solution procedure for the solution of the adjoint equations developed from a discrete sensitivity approach does not appear to be as easy. The level of difficulty in this recycling for the discrete approach is intimately connected to the algorithm used for the flow solution. For methods such as those used by Young et al. [115, 42] where GMRES (Generalized Minimum Residual) is used to solve the very large linear algebra problem, the issue of constructing a discrete adjoint solver involves simply replacing the Jacobian $\left[\frac{\partial R}{\partial w}\right]$ with its transpose. In the case of the explicit solvers used in many of Jameson's codes, a discrete sensitivity based adjoint equation solver may be constructed by working out some tedious algebra to obtain the symbolic form of the adjoint equation residual ($\left[\frac{\partial R}{\partial w}\right]^T \psi$) such that it may replace the flow field residual, R, within the flow solver. For flow solution methods requiring particular decompositions and/or approximations to the Jacobian, such as approximate factorization, the reformulation of the iterative procedure to calculate the adjoint equation solution may be considerably more complex unless the structure of the continuous adjoint is used as a guide. The application of the continuous sensitivity analysis fosters the easy recycling of the flow solution algorithm in any of these cases, since the steps applied to the original governing differential equations can be duplicated for the adjoint differential equations.

Due to this difficulty in recycling of the flow solution process, many present discrete sensitivity methods resort to matrix elimination methods to solve (1.9) or (1.13). While direct techniques to solve these large sparse systems can be robust and reliable they suffer when the number of mesh points becomes large because the operational count grows as $O(\hat{n}\hat{b}^2)$ and the storage goes as $O(\hat{n}\hat{b})$, where $\hat{n}$ is the number of unknowns and $\hat{b}$ is the bandwidth. It is thus impractical to use direct methods in all but the smallest problems. Therefore, in order to solve larger systems, alternatives such as sophisticated matrix decomposition [72] or incremental iterative [69] strategies have been employed. These methods have shown modest improvements over standard Gaussian elimination, but they have not proven to be as efficient as methods developed for CFD. When the number of mesh points becomes large, especially in the case of three-dimensional problems, the $O(\hat{n})$ operational counts and the $O(\hat{n})$ storage of explicit iteration schemes used in many CFD methods can, if applied in an adjoint solution strategy, significantly reduce its time and memory requirements.

In general, competitive discrete sensitivity methods will have to be able to solve the adjoint system with approximately the same computational resources as required for the flow solution algorithm. Present work by Taylor et al. [69] to develop such methods shows promise. The same group has also shown that discrete sensitivity analysis can be approached by using the ADIFOR software discussed above [102]. A challenging aspect of discrete sensitivity analysis is obtaining the exact dependencies of R with respect to w such that the flux Jacobian can be obtained. For complex CFD codes which rarely, if ever, construct $\left[\frac{\partial R}{\partial w}\right]$ explicitly, this can be daunting. It turns out that if ADIFOR is used on selected subroutines, these dependencies can be obtained automatically. Unfortunately, this approach has only been explored for methods that solve the direct problem (1.13) and not the adjoint problem (1.9). Hence the computational cost is still proportional to $\hat{n}$ flow solutions. The best compromise may be to formulate the problem using a continuous approach, which gives a far deeper intuitive understanding of the adjoint system and its boundary conditions, and then to augment it with consistency checks obtained from discrete sensitivity analysis.

1.6 OTHER APPLICATIONS OF CONTROL THEORY IN AERODYNAMICS

A variety of alternative formulations of the design problem, other than the ones discussed thus far, may be treated systematically within the framework of the mathematical theory for control of systems governed by partial differential equations [78]. Other applications of control theory in aerodynamics have been explored by Pironneau for optimum shape design of systems governed by elliptic equations [86]. More recently Pironneau [87] as well as Huan and Modi [40, 41] have studied the use of control theory to solve design problems subject to the incompressible Navier-Stokes equations. Ta'asan, Kuruvila, and Salas have implemented a "one-shot" approach in which the constraint represented by the flow equations is required to be satisfied only at the final converged solution [107, 71]. While the method has currently been explored only for potential flows without shock waves, it remains an intriguing option. Their method is similar to the work presented here in that they use the continuous sensitivity approach. The difference lies in the order in which the design process is evolved. While the work here decouples the design problem into separate parts (flow solver, adjoint solver, optimization algorithm) Ta'asan et al. have attempted to couple the systems strongly through the use of multigrid algorithms. While this process does not lend itself to coupling with more sophisticated optimization algorithms, where errors in the gradient can have adverse effects, it may lead to efficient design procedures by virtue of the concurrent convergence of the three systems. More investigation using the approach will be necessary in order to demonstrate success on applications involving transonic and supersonic problems, where the nonlinear interactions may impede its convergence.

1.7 Goals for this Research

In this research the continuous sensitivity approach employed by Jameson and verified by Lewis and Agarwal will be extended to allow the treatment of more complex geometries. This will be accomplished by foregoing the reliance of the previous method on an analytic mapping, in favor of a general finite volume formulation combined with a grid perturbation method. The research will also introduce the use of both alternative design space parameterizations and an alternative design space search strategy. Two different parameterizations in the form of Hicks-Henne functions and B-spline control points will be tested. The design space search strategy will be an unconstrained quasi-Newton method which has frequently been used for finite difference optimization problems. In addition, various discretization procedures for the adjoint systems that result from the application of continuous sensitivity analysis, including ones that mimic the discretization arrived at if discrete sensitivity analysis were employed, will be presented.

These issues will be explored for both the potential flow and Euler equations. Both equation sets represent inviscid models of the flow physics. The potential flow equation introduces the further approximation that it does not admit the production or convection of vorticity and entropy. While the inviscid approximation will be used throughout this research, it is by no means a restriction for control theory-based design. The purpose of developing the two different design methods is to demonstrate the applicability of using control theory on different problems. Specifically, the difference in character between the two governing systems will have dramatic implications for the creation of the two design methods. For the application to the second-order potential flow equation, the Laplacian-like character of the difference operator, combined with the fact that only a scalar valued variable is involved, results in a straightforward development of the costate boundary conditions. In contrast, the first order hyperbolic nature of the Euler equations, combined with their vector form, results in an initial-valued costate system whose boundary conditions must be addressed with great care. On the other hand, the easier treatment of the boundary conditions for the potential flow costate will be offset by its more difficult development of the related differential equations to define the adjoint equation. This greater difficulty will result primarily from the existence of a jump in the potential flow solution to account for the circulation which is not necessary for the Euler equations.

In order to gain an even greater understanding of various issues concerning the development and implementation of control theory based design, the two design methods will employ different solution techniques. The potential flow equation and its related costate system will be solved by an alternating direction implicit (ADI) scheme, while the Euler equations and their costates will be solved by a Runge-Kutta-like explicit scheme. Regardless of this choice, it will be shown that determining the details of the solution algorithm for the costate systems by recycling those used for the corresponding flow systems is greatly eased by the use of continuous sensitivity analysis.

To illustrate the viability of design via control theory as more than a point of academic interest, the technique will be applied to modern ultra-fast CFD codes that incorporate multigridding. The complementary multigridding of the adjoint systems will also be included such that the adjoint solution will place no higher demand on the computational resources than the flow solution. Several examples will be presented illustrating the capabilities and limitations of the technique in comparison with more traditional methods.

Chapter 2

MATHEMATICAL MODELS FOR FLOW PHYSICS

The development of the two new airfoil design methods in this work is based on numerical techniques for solving the potential flow and Euler equations that approximate flow dynamics. These two models represent different levels of approximation for the flow physics, and thus obtain different solutions. Both are inviscid approximations and are therefore only realistic for the treatment of problems where viscous effects do not play a dominant role in the aerodynamics. This chapter develops both sets of equations and discusses the consequences of their respective approximations. Further, this chapter describes the discretization methodology and solution procedures employed for both systems. The development of two different design methods, depending on two different sets of governing equations, which are solved by different solution strategies, extends the understanding and applicability of aerodynamic design via control theory.

2.1 GOVERNING CONSERVATION EQUATIONS

Mathematical models are used to describe physical phenomena with varying degrees of accuracy. Within the field of fluid dynamics, the mathematical representation of the conservation laws of mass, momentum, and energy define the governing equations. Many formulations of these laws are possible, and can be found in a variety of texts.

For a fixed control volume with respect to an inertial reference frame and a non-reacting fluid with no sources or sinks of mass, momentum or energy, the governing equations may be stated in integral form as (*cf.*, Aris [1], Hansen [35, 34], and Thompson [109]):

Conservation of Mass

$$\frac{d}{dt}\int_{\mathcal{V}} \rho d\mathcal{V} + \int_{\mathcal{S}} \rho(\mathbf{v}\cdot\mathbf{n})\, d\mathcal{S} = 0 \tag{2.1a}$$

Conservation of Momentum

$$\frac{d}{dt}\int_{\mathcal{V}} \rho\mathbf{v} d\mathcal{V} + \int_{\mathcal{S}} \rho\mathbf{v}(\mathbf{v}\cdot\mathbf{n})\, d\mathcal{S} = \int_{\mathcal{S}} \mathbf{t} d\mathcal{S} \tag{2.1b}$$

Conservation of Energy

$$\frac{d}{dt}\int_{\mathcal{V}} \rho E d\mathcal{V} + \int_{\mathcal{S}} \rho E(\mathbf{v}\cdot\mathbf{n})\, d\mathcal{S} = \int_{\mathcal{S}} \mathbf{t}\cdot\mathbf{v} d\mathcal{S} - \int_{\mathcal{S}} \mathbf{q}\cdot\mathbf{n} d\mathcal{S}, \tag{2.1c}$$

where ρ is the density, $\mathbf{v}$ is the velocity vector, $\mathbf{t}$ is the surface traction vector, $\mathbf{q}$ is the heat flux vector, E is the total energy, $\mathbf{n}$ is the surface normal, and $\mathcal{V}$ and $\mathcal{S}$ are the volume and surface respectively. They may also be stated in differential form:

$$\begin{aligned}
\frac{\partial \rho}{\partial t} + \nabla\cdot\rho\mathbf{v} &= 0 \\
\frac{\partial \rho\mathbf{v}}{\partial t} + \nabla\cdot\rho\mathbf{v}\mathbf{v} &= \nabla\cdot\mathbf{T} \\
\frac{\partial \rho E}{\partial t} + \nabla\cdot\rho E\mathbf{v} &= \nabla\cdot(\mathbf{T}\cdot\mathbf{v}) - \nabla\cdot\mathbf{q},
\end{aligned}$$

where $\mathbf{T}$ is the stress tensor including the pressure. These equations are typically referred to as being in conservative or divergence form.

2.1.1 THE THERMODYNAMIC STATE

Solving the above conservation equations requires that the thermodynamic properties of the fluid system be specified. Thus, an equation of state is needed. Additionally, the assumptions of a thermally perfect ($p = \rho\mathbf{R}T$) and calorically perfect ($\gamma \equiv \frac{c_p}{c_v} = $ *constant*) fluid will apply. Consequently, the fluid is a *calorically perfect gas*.

2.2 THE EULER EQUATIONS

The governing equations given thus far have been presented with only limited approximations. For this research the equations are further simplified. The first design method employs the potential flow equation as a basis of formulation while the second design method uses the Euler equations. Here, for the sake of clarity, it is convenient to develop the Euler equations first.

Ignoring viscous and heat transfer effects leads to the development of the Euler equations, which in integral form are:

$$\frac{d}{dt}\int_{\mathcal{V}} \rho d\mathcal{V} + \int_{S} \rho(\mathbf{v}\cdot\mathbf{n})\, dS = 0 \tag{2.2a}$$

$$\frac{d}{dt}\int_{\mathcal{V}} \rho\mathbf{v} d\mathcal{V} + \int_{S} \rho\mathbf{v}(\mathbf{v}\cdot\mathbf{n})\, dS = -\int_{S} p\mathbf{n} dS \tag{2.2b}$$

$$\frac{d}{dt}\int_{\mathcal{V}} \rho E d\mathcal{V} + \int_{S} \rho H(\mathbf{v}\cdot\mathbf{n})\, dS = 0, \tag{2.2c}$$

where H is the total enthalpy. The conservative differential form of the Euler equations is

$$\frac{\partial \rho}{\partial t} + \nabla\cdot\rho\mathbf{v} = 0 \tag{2.3a}$$

$$\frac{\partial \rho\mathbf{v}}{\partial t} + \nabla\cdot(\rho\mathbf{v}\mathbf{v}) + \nabla p = 0 \tag{2.3b}$$

$$\frac{\partial \rho E}{\partial t} + \nabla\cdot\rho H\mathbf{v} = 0. \tag{2.3c}$$

For a perfect gas,

$$p = (\gamma - 1)\rho\left\{E - \frac{1}{2}(\mathbf{v}\cdot\mathbf{v})\right\} \tag{2.4}$$

and

$$\rho H = \rho E + p. \tag{2.5}$$

Equations 2.3, with the perfect gas conditions, form a system of hyperbolic partial differential equations which very accurately models many commonly occurring flow phenomena in aerodynamics. The numerical treatment of this system of equations is addressed in Section 2.5.

To derive the potential flow equation used in the first design method, further simplifications are necessary.

2.3 The Potential Flow Equations

In the absence of shock waves, an initially irrotational flow will remain irrotational, and we can assume that the velocity vector v is the gradient of a potential ϕ. In the presence of weak shock waves this remains a fairly good approximation.

The differential form of the continuity equation (2.3a) may then be rewritten as

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \nabla \phi) = 0, \tag{2.6}$$

the time-dependent potential flow equation. In order to complete the equation, an expression for ρ must be obtained in terms of the potential ϕ. This can be achieved by rearranging the perfect gas equation for pressure (2.4) with the added assumption that the flow is isentropic, giving

$$\rho = \left\{1 + \frac{\gamma - 1}{2} M_\infty^2 \left(1 - 2\phi_t - \left(\phi_x^2 + \phi_y^2\right)\right)\right\}^{\frac{1}{(\gamma-1)}}, \tag{2.7}$$

where

$$p = \frac{\rho^\gamma}{\gamma M_\infty^2}, \quad c^2 = \frac{\gamma p}{\rho}. \tag{2.8}$$

Here M_∞ is the Mach number in the free stream, and the equations have been non-dimensionalized so that ρ and v have the value unity in the far field. The next section presents a methodology to solve this equation.

2.4 Potential Flow Solution Methodology

Separate numerical procedures are used to obtain solutions of the potential flow and Euler equations. In each case the computational domain is constructed as a set of discrete cells which completely tessellates the flow field region of interest. For two-dimensional profiles the computational domains extend from the geometry surface to an outer boundary where deviations from free stream conditions are small. Although

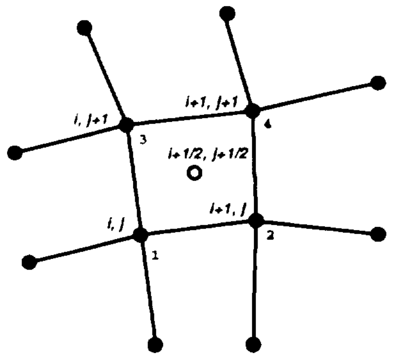


***Figure 2.1:** Mesh used for the Potential Flow Equation in the Physical Plane*

it is possible to tessellate these domains with many mesh types, for this work two-dimensional body-fitted structured meshes are used exclusively. Once such a mesh is created, a solution procedure for a given set of governing equations must be defined.

2.4.1 Finite Volume Formulation

In the first design method, the potential flow equation derived in Section (2.3) is used. The numerical solution procedure for this equation is based on the methodology developed by Jameson [45, 46, 44, 47, 64, 15, 51, 48] and realized in his FLO42 airfoil analysis computer program. The method is a finite volume technique which is applied to a body-fitted grid such that the boundaries are grid lines. The solution ϕ is defined at the mesh points and the variables ρ, u and v are defined at the cell centers. Figure 2.1 illustrates a typical i, j cell with its neighbors. The steady state potential equation can be written in differential form in a Cartesian coordinate system as

$$\frac{\partial}{\partial x}(\rho u) + \frac{\partial}{\partial y}(\rho v) = 0, \tag{2.9}$$

where

$$u = \phi_x, \quad v = \phi_y$$

and ρ is defined by equation (2.7). Equation (2.9) can be transformed into an arbitrary coordinate system (ξ, η) by defining

$$K = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{bmatrix} \text{ and } K^{-1} = \begin{bmatrix} \frac{\partial \xi}{\partial x} & \frac{\partial \xi}{\partial y} \\ \frac{\partial \eta}{\partial x} & \frac{\partial \eta}{\partial y} \end{bmatrix}.$$

Let J be the determinant of K:

$$J = \det|K| = \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi}.$$

Now

$$\begin{Bmatrix} u \\ v \end{Bmatrix} = \begin{Bmatrix} \phi_x \\ \phi_y \end{Bmatrix} = \begin{bmatrix} \frac{\partial \xi}{\partial x} & \frac{\partial \eta}{\partial x} \\ \frac{\partial \xi}{\partial y} & \frac{\partial \eta}{\partial y} \end{bmatrix} \begin{Bmatrix} \phi_\xi \\ \phi_\eta \end{Bmatrix}$$

$$= K^{T^{-1}} \begin{Bmatrix} \phi_\xi \\ \phi_\eta \end{Bmatrix}. \tag{2.10}$$

and also

$$\begin{Bmatrix} \phi_\xi \\ \phi_\eta \end{Bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \begin{Bmatrix} \phi_x \\ \phi_y \end{Bmatrix} = K^T \begin{Bmatrix} \phi_x \\ \phi_y \end{Bmatrix}. \tag{2.11}$$

We can now write the potential flow equation as

$$\frac{\partial}{\partial \xi}(\rho J U) + \frac{\partial}{\partial \eta}(\rho J V) = 0 \quad \text{in } D. \tag{2.12}$$

Here, U and V represent the contravariant velocities

$$\begin{Bmatrix} U \\ V \end{Bmatrix} = \frac{1}{J} \begin{bmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial x}{\partial \eta} \\ -\frac{\partial y}{\partial \xi} & \frac{\partial x}{\partial \xi} \end{bmatrix} \begin{Bmatrix} u \\ v \end{Bmatrix} = K^{-1} \begin{Bmatrix} u \\ v \end{Bmatrix} \tag{2.13}$$

and

$$\begin{Bmatrix} U \\ V \end{Bmatrix} = \mathcal{A} \begin{Bmatrix} \phi_\xi \\ \phi_\eta \end{Bmatrix},$$

with

$$\mathcal{A} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = K^{-1} K^{T^{-1}} = \left(K^T K\right)^{-1}.$$

Thus by noting that $\mathcal{A}$ is symmetric we have

$$U = A_{11}\phi_\xi + A_{12}\phi_\eta \tag{2.14}$$

$$V = A_{12}\phi_\xi + A_{22}\phi_\eta. \tag{2.15}$$

The speed is now determined from (2.11) and (2.13) by the formula

$$q^2 = U\phi_\xi + V\phi_\eta.$$

To formulate a numerical scheme for the potential flow equation, Figure 2.1 illustrates a typical cell with its neighbors which is locally transformed into a Cartesian unit cell. Next the quantities x, y and ϕ within the cell with vertices 1, 2, 3 and 4 are defined by the following bilinear mapping:

$$\begin{aligned}
x &= \frac{1}{4}\{x_1(1-\xi_l)(1-\eta_l) + x_2(1+\xi_l)(1-\eta_l) + x_3(1-\xi_l)(1+\eta_l) + x_4(1-\xi_l)(1-\eta_l)\} \\
y &= \frac{1}{4}\{y_1(1-\xi_l)(1-\eta_l) + y_2(1+\xi_l)(1-\eta_l) + y_3(1-\xi_l)(1+\eta_l) + y_4(1-\xi_l)(1-\eta_l)\} \\
\phi &= \frac{1}{4}\{\phi_1(1-\xi_l)(1-\eta_l) + \phi_2(1+\xi_l)(1-\eta_l) + \phi_3(1-\xi_l)(1+\eta_l) + \phi_4(1-\xi_l)(1-\eta_l)\}
\end{aligned}$$

where ξ_l and η_l are the local coordinates for the cell and vary between -1 and 1. This corresponds to isoparametric finite elements. It results in convenient formulas at the cell center since quantities for x, y and ϕ become simple averages. The mesh metrics and partial derivative of ϕ are then defined at the cell centers as

$$\begin{aligned}
\frac{\partial x}{\partial \xi} &= \frac{1}{2}(x_4 - x_3 + x_2 - x_1) \\
\frac{\partial y}{\partial \xi} &= \frac{1}{2}(y_4 - y_3 + y_2 - y_1) \\
\frac{\partial x}{\partial \eta} &= \frac{1}{2}(x_4 - x_2 + x_3 - x_1)
\end{aligned}$$

$$
\begin{aligned}
\frac{\partial y}{\partial \eta} &= \frac{1}{2}(y_4 - y_2 + y_3 - y_1) \\
\frac{\partial \phi}{\partial \xi} &= \frac{1}{2}(\phi_4 - \phi_3 + \phi_2 - \phi_1) \\
\frac{\partial \phi}{\partial \eta} &= \frac{1}{2}(\phi_4 - \phi_2 + \phi_3 - \phi_1)
\end{aligned}
$$

with the other unknowns such as U, V, q and the terms of $\mathcal{A}$ obtained as multiples of these quantities. The flux balance is then constructed in the computational plane by averaging of the cell-centered quantities along the edges of a secondary finite volume shown in Figure (2.2) to give

$$
\begin{aligned}
&(\rho J U)_{i+\frac{1}{2},j+\frac{1}{2}} + (\rho J U)_{i+\frac{1}{2},j-\frac{1}{2}} - (\rho J U)_{i-\frac{1}{2},j+\frac{1}{2}} - (\rho J U)_{i-\frac{1}{2},j-\frac{1}{2}} \\
&+(\rho J V)_{i+\frac{1}{2},j+\frac{1}{2}} + (\rho J V)_{i-\frac{1}{2},j+\frac{1}{2}} - (\rho J V)_{i+\frac{1}{2},j-\frac{1}{2}} - (\rho J V)_{i-\frac{1}{2},j-\frac{1}{2}} = 0 \qquad (2.16)
\end{aligned}
$$

This scheme, combined with an effective iteration algorithm, suffices for subsonic flow

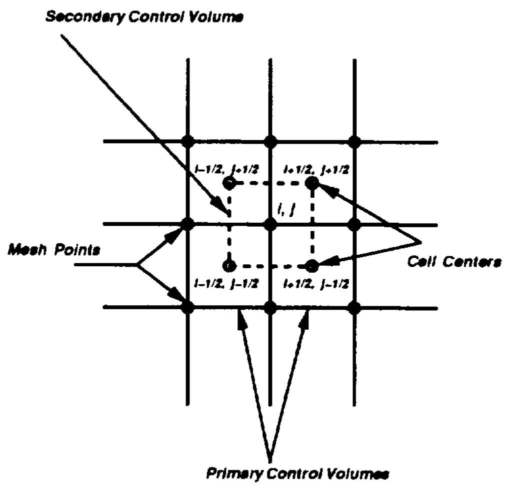


Figure 2.2: *Mesh for the Potential Flow Equation in the Computational Plane*

calculations where the entire flow domain is elliptic. To model transonic flows where embedded regions of supersonic flow create a type switch in the governing equation to hyperbolic, a modification to these equations is necessary.

2.4.2 Mixed type equations and discontinuities

Typically, numerical schemes to solve the discrete form of the governing equations for fluid dynamics require the addition of non-physical terms to ensure proper stability. These artificial terms are required for many reasons. Use of the potential flow equation to model transonic conditions poses a considerable problem because this equation models discontinuities as isentropic jumps. Thus the equation allows both expansion and compression shocks. To eliminate the possibility of non-physical expansion shocks in the solution, artificial viscous terms which destroy the symmetry of the potential flow equation should be added to produce upwind biasing. The Euler equations, developed later in this chapter, do not suffer this type of difficulty since only entropy-producing compression shocks are admitted.

However, regardless of the choice of the governing equation, discontinuities in the flow field present difficulties for the numerical scheme. For the flow of a real fluid, entities such as shock waves or contact discontinuities are regions where large gradients in particular flow properties exist. The length scales of these regions are typically orders-of-magnitude less than the characteristic dimensions of the flow field. Under the approximation of a perfect inviscid fluid, these narrow regions of high gradients should be treated as true discontinuities.

When a domain is to possess discontinuities, care must be taken in the construction of a numerical scheme. The numerical scheme must be capable of reproducing the effects of the discontinuity, while anomalies such as multi-valued points must be avoided. This dilemma is usually handled in one of two ways. The first, shock-capturing, incorporates artificial dissipation to *smear* a discontinuity into a continuous, smooth region. The other, shock-fitting, splits the computational domain at discontinuities and uses *jump conditions* (obtained from conservations laws) to set conditions on the boundaries of the newly formed sub-regions. Shock-fitting is useful in that it can, in theory, precisely represent a discontinuity of proper strength and location. However, it is difficult to implement because locating one or more shock waves and splitting the computational domain can be tricky and tedious, particularly in the case of complex three-dimensional flows.

Shock-capturing yields adequate results if the discrete shock is represented by a few points and the computational domain contains a sufficient number of points in the vicinity of the discontinuity, thereby confining the extent of smearing to a limited region. The magnitude and extent of this error due to smearing are functions of the mesh size in the region of the discontinuity and the type of numerical dissipation used to smear the shock. Nevertheless, it is possible for shock-captured solutions to agree well with those produced by exact jump conditions. Both of the methods presented in this work rely upon shock capturing to model discontinuities in the flow field.

2.4.3 ARTIFICIAL DISSIPATION

Returning to the development of a solution procedure for the potential flow equation, the previous section outlines the logical next step. Upwind biasing must be added to regions where the flow is supersonic to eliminate spurious expansion shocks which might otherwise appear. The symmetry of the potential flow equation which permits both compression and expansion shocks can be removed by the addition of small artificial dissipation terms. The governing equation is rewritten as

$$\frac{\partial}{\partial \xi}(\rho J U + P) + \frac{\partial}{\partial \eta}(\rho J V + Q) = 0,$$

where P and Q are constructed such that

$$\begin{aligned} P \quad &\text{approximates} \quad -\hat{\mu} J \left|U\right| \frac{\partial \rho}{\partial \xi} \\ Q \quad &\text{approximates} \quad -\hat{\mu} J \left|V\right| \frac{\partial \rho}{\partial \eta}. \end{aligned} \tag{2.17}$$

The coefficient $\hat{\mu}$ is then set to the switching operator,

$$\hat{\mu} = \max\left\{0, \left(1 - \frac{M_c^2}{M^2}\right)\right\},$$

where M_c is a critical Mach number set close to 1, such that $\hat{\mu}$ becomes zero in subsonic regions giving $P = Q = 0$. The form of P and Q approximates a shift in the location of where ρ is evaluated in the original equation (2.12). The analogy is exact for positive flow on a Cartesian mesh since $J = 1$, giving

$$\frac{\partial}{\partial \xi}\left(U\left(\rho + \hat{\mu}\frac{\partial \rho}{\partial \xi}\right)\right) + \frac{\partial}{\partial \eta}\left(V\left(\rho + \hat{\mu}\frac{\partial \rho}{\partial \eta}\right)\right) = 0. \tag{2.18}$$

To avoid incorporating unnecessary complications, the upwind biasing is implemented by adding approximations of the terms $\frac{\partial \rho}{\partial \xi}$ and $\frac{\partial \rho}{\partial \eta}$ directly to the difference formula (2.16) without resorting to the finite volume expansion. These approximations for $\frac{\partial \rho}{\partial \xi}$ and $\frac{\partial \rho}{\partial \eta}$ are obtained by realizing that from equation (2.7),

$$\begin{aligned}\frac{\partial \rho}{\partial \xi} &= -\frac{\rho}{2c^2}\frac{\partial}{\partial \xi}\left(q^2\right)\\ \frac{\partial \rho}{\partial \eta} &= -\frac{\rho}{2c^2}\frac{\partial}{\partial \eta}\left(q^2\right).\end{aligned} \tag{2.19}$$

By substituting (2.17) into (2.19) and using the symmetry of $\mathcal{A}$ we can develop the approximations,

$$\begin{aligned}\hat{P} &= \hat{\mu}\frac{\rho J}{c^2}\left(U^2\phi_{\xi\xi} + UV\phi_{\xi\eta}\right)\\ \hat{Q} &= \hat{\mu}\frac{\rho J}{c^2}\left(UV\phi_{\xi\eta} + V^2\phi_{\eta\eta}\right).\end{aligned}$$

P and Q can be calculated at the cell edges from

$$P_{i+\frac{1}{2},j} = \begin{cases} \hat{P}_{i,j} & \text{if } U > 0\\ -\hat{P}_{i+1,j} & \text{if } U < 0\end{cases}$$

$$Q_{i,j+\frac{1}{2}} = \begin{cases} \hat{Q}_{i,j} & \text{if } V > 0\\ -\hat{Q}_{i,j+1} & \text{if } V < 0\end{cases} \tag{2.20}$$

The new numerical system can treat both subsonic and transonic conditions and does not admit isentropic expansion discontinuities.

A close examination of this scheme, which reduces to (2.16) in subsonic regions, shows that one mode of instability still persists. If the finite volume scheme is expanded for a Cartesian mesh with incompressible flow, it reduces to a rotated five point Laplacian operator given by

$$\phi_{i+1,j+1} + \phi_{i+1,j-1} + \phi_{i-1,j+1} + \phi_{i-1,j-1} - 4\phi_{i,j} = 0. \tag{2.21}$$

The difficulty arises because this scheme allows two superimposed solutions to exist that are offset to create a checkerboard pattern of odd-even decoupling. To prevent the solution from decoupling in this manner, the scheme is adjusted with terms which rotate the stencil back. By introducing the averaging and differencing operators

$$\begin{aligned}
\bar{\nu}_\xi\left(f_{i,j}\right) &= \frac{1}{2}\left(f_{i+\frac{1}{2},j} + f_{i-\frac{1}{2},j}\right)\\
\bar{\nu}_{\xi\xi}\left(f_{i,j}\right) &= \frac{1}{4}\left(f_{i+1,j} + 2f_{i,j} + f_{i-1,j}\right) = \bar{\nu}_\xi\bar{\nu}_\xi\left(f_{i,j}\right)\\
\bar{\delta}_\xi\left(f_{i,j}\right) &= \left(f_{i+\frac{1}{2},j} - f_{i-\frac{1}{2},j}\right)\\
\bar{\delta}_{\xi\xi}\left(f_{i,j}\right) &= \left(f_{i+1,j} - 2f_{i,j} + f_{i-1,j}\right) = \bar{\delta}_\xi\bar{\delta}_\xi\left(f_{i,j}\right),
\end{aligned}$$

where f is an arbitrary function and similar formulas are defined for the η direction, we can define the difference operator (2.21) for incompressible flow on a Cartesian mesh as

$$(\bar{\nu}_{\eta\eta}\bar{\delta}_{\xi\xi} + \bar{\nu}_{\xi\xi}\bar{\delta}_{\eta\eta})\phi = 0. \tag{2.22}$$

Note that multiple operators commute, whether they are multiple subscripts as in $\bar{\delta}_{\xi\xi}$ or multiple operators such as $\bar{\nu}_{\eta\eta}\bar{\delta}_{\xi\xi}$. Henceforth in this text, $\bar{\nu}$ will imply average operators and $\bar{\delta}$ will imply difference operators. Now by augmenting the operator (2.22) with the term $\sigma\bar{\delta}_{\xi\eta\xi\eta}\phi$ and setting $\sigma = \frac{1}{2}$, the standard five point stencil can be recovered, which eliminates the decoupling. The actual difference operator for general flows and meshes can be similarly augmented with the appropriate terms. These may be written as

$$\bar{\delta}_{\xi\eta}\left[(\bar{A}^{(\xi)} + \bar{A}^{(\eta)})\bar{\delta}_{\xi\eta}\phi\right]$$

where

$$\begin{aligned}
\bar{A}^{(\xi)} &= \rho J(A_{11} - \frac{U^2}{c^2})\\
\bar{A}^{(\eta)} &= \rho J(A_{22} - \frac{V^2}{c^2}).
\end{aligned}$$

The difference formulation is now complete and can be fully written out as

$$\mathcal{L}\phi = \bar{\delta}_\xi\left(\bar{\nu}_\eta(\rho JU) + P\right) + \bar{\delta}_\eta\left(\bar{\nu}_\xi(\rho JV) + Q\right) - \sigma\bar{\delta}_{\xi\eta}\left[\left(\bar{A}^{(\xi)} + \bar{A}^{(\eta)}\right)\bar{\delta}_{\xi\eta}\phi\right] = 0.$$

(2.23)

Details of the numerical scheme are given by Jameson [44, 45, 46, 64, 15]. The next step is to construct an iteration scheme to solve (2.23).

2.4.4 ITERATION SCHEME

A generalized alternating direction implicit (ADI) procedure is used to converge the discrete system of equations given by (2.23). The difference operator $\mathcal{Z}$ is now introduced as

$$\mathcal{Z} = \bar{\alpha}_0 + \bar{\alpha}_1 \bar{\delta}_\xi^- + \bar{\alpha}_2 \bar{\delta}_\eta^-$$

where $\bar{\delta}_\xi^-$ and $\bar{\delta}_\eta^-$ are defined as one-sided differences biased in the direction of the ADI sweep. Now by linearizing equation (2.23) about a perturbation potential $\delta\phi$, and dropping higher order terms the ADI scheme can be formulated as

$$\left(\mathcal{Z} - \bar{\delta}_\xi\left(\bar{\nu}\left(\bar{A}^{(\xi)}\right)\bar{\delta}_\xi\right) - \bar{\delta}_\xi\left(\bar{P}\bar{\delta}_\xi^2\right)\right)\left(\mathcal{Z} - \bar{\delta}_\eta\left(\bar{\nu}\left(\bar{A}^{(\eta)}\right)\bar{\delta}_\eta\right) - \bar{\delta}_\eta\left(\bar{Q}\bar{\delta}_\eta^2\right)\right)\hat{\delta}\phi^{n+1} = -\omega\mathcal{Z}\mathcal{L}\phi^n \tag{2.24}$$

where ω is a relaxation factor, $\mathcal{L}$ is the operator defined by (2.23), $\hat{\delta}$ is the correction operator, $\bar{\delta}_\xi$ and $\bar{\delta}_\eta$ are the shifted operators defined so as to be consistent with (2.20), and

$$\begin{aligned} \bar{P} &= \frac{\hat{\mu}\rho J U^2}{c^2} \\ \bar{Q} &= \frac{\hat{\mu}\rho J V^2}{c^2}. \end{aligned}$$

This scheme can be considered a discrete approximation to the time-dependent equation

$$\beta_0\phi_t + \beta_1\phi_{\xi t} + \beta_2\phi_{\eta t} = \mathcal{L}\phi \tag{2.25}$$

where the coefficients β_0, β_1 and β_2 are related to the parameters $\bar{\alpha}_0$, α_1 and α_2. The advantage of using $\mathcal{Z}$ in equation (2.24) to drive the ADI cycle, as opposed to a traditional constant, is that the corresponding time-dependent equation (2.25) remains hyperbolic irrespective of the signs of $\bar{A}^{(\xi)}$ and $\bar{A}^{(\eta)}$. The relaxation scheme (2.24) is performed in two stages. The first stage involves solving

$$\left(\mathcal{Z} - \bar{\delta}_{\eta}\left(\bar{\nu}\left(\bar{A}^{(\eta)}\right)\bar{\delta}_{\eta}\right) - \bar{\delta}_{\bar{\eta}}\left(\bar{Q}\bar{\delta}_{\eta}^{2}\right)\right)\Upsilon = -\omega\mathcal{Z}\mathcal{L}\phi^{n}$$

for Υ. Recalling that $\mathcal{Z}$ has one-sided differences, the system is thus solved in a line by line process where information from the previous line has already been updated and each line is solved implicitly. For this first step the domain is split into two parts about the leading edge of the airfoil. The solution is then swept downstream around the airfoil separately but symmetrically for the upper and lower halves. Each line of the solution requires that a scalar penta-diagonal solver be used for Υ. Once a single pass of this split sweep is completed in the ξ direction, the second part of the solution follows by solving

$$\left(\mathcal{Z} - \bar{\delta}_{\xi}\left(\bar{\nu}\left(\bar{A}^{(\xi)}\right)\bar{\delta}_{\xi}\right) - \bar{\delta}_{\bar{\xi}}\left(\bar{P}\bar{\delta}_{\xi}^{2}\right)\right)\hat{\delta}\phi^{n+1} = \Upsilon$$

for the desired correction $\hat{\delta}\phi^{n+1}$. This system is also solved in a line by line scheme, but with the sweep direction running in the η direction from the outer boundary inward. While this choice is somewhat arbitrary when compared with the obvious sweep direction chosen for the ξ direction, the technique still improves convergence by capturing and transmitting inward the updated value of the circulation specified at the outer boundary. More details of the scheme are presented in [46, 48, 44, 47].

2.4.5 CONVERGENCE ACCELERATION

Convergence acceleration is especially important in design applications since more than one complete flow solution will be required. Thus even with the efficient ADI scheme presented in the last section, convergence may still be too slow.

MULTIGRID ACCELERATION

One method of enhancing convergence performance of many numerical schemes is by the use of a multigrid method. The idea, presented by Jameson [51] for transonic potential flows, is to add corrections to the solution based on calculations obtained by using different mesh densities. The corrections from different meshes are advantageous for two reasons. First, the calculations of the updates on coarser meshes are much cheaper since they involve significantly fewer grid points. Second, a coarser mesh converges faster since, if the scheme is not fully implicit, information propagates at speeds proportional to the physical mesh widths. If a sequence of meshes are defined with different degrees of fidelity, the coarser meshes will propagate lower frequency modes in the solution, while the finer meshes permit the resolution of the higher frequencies. In other words, a system of meshes superimposed on the same domain with varying resolutions allows for the construction of solution procedures that avoid the need for the fine mesh solution process to transfer the entire band of solution frequencies.

To implement a proper multigrid algorithm, a sequence of meshes must first be defined with varying degrees of fidelity. For structured meshes, as are used in this research, the obvious choice is to construct coarse meshes from fine meshes by eliminating every other mesh point. By repeatedly applying this concept starting from an initial fine mesh, a sequence of successively coarser meshes can be constructed. The second important issue with any multigrid procedure is to ensure that convergence is achieved and accelerated with respect to the finest mesh. Thus, the developing solution on the fine mesh should be used to drive the solution on all coarser meshes. To illustrate this procedure, consider the linear problem

$$\bar{\mathcal{L}}_f \phi_f = 0$$

where $\bar{\mathcal{L}}_f$ is a linear difference operator defined on the fine mesh, and ϕ_f is the fine mesh solution. In delta form this can be written

$$\bar{\mathcal{L}}_f \delta\phi_f^{n+1} = -\bar{\mathcal{L}}_f \phi_f^n \tag{2.26}$$

where the subscripts denote whether the mesh is fine or coarse and the superscripts determine the iteration count. Now instead of solving this equation successively on the fine mesh alone, we can define on the next coarser mesh in the sequence:

$$\mathcal{L}_c \delta\phi_c^{n+1} = -\bar{\mathcal{L}}_c \phi_c^n + \mathcal{P}_p \tag{2.27}$$

with

$$\mathcal{P}_p = \hat{\mathcal{L}}_c \phi_c^n - \mathcal{I}_f^c \left(\hat{\mathcal{L}}_f \phi_f^n \right),$$

where $\mathcal{I}_f^c$ is a collection transfer operator that moves a quantity from the fine grid to the coarse grid. Note that the residual on the fine mesh replaces the coarse mesh residual and becomes the forcing function to drive the solution. After solving for $\delta\phi_c^{n+1}$, the solution on the fine mesh can be updated with the interpolation transfer operator,

$$\phi_f^{n+1} = \phi_f^n + \mathcal{I}_c^f \delta\phi_c^{n+1}.$$

The entire solution process can be defined by first calculating the correction on all meshes, starting from the finest mesh according to (2.26), and proceeding on all coarser meshes by (2.27), and then interpolating all these corrections back to the finest mesh.

The procedure defined by equations (2.26) and (2.27) is employed to solve the nonlinear system defined by (2.24). For a more complete treatment of the process and an understanding of its connection to the ADI method, see references [47, 48, 51]. Another example of the implementation of the multigrid algorithm is given in Section 2.5.4 where it is applied to the Euler equations.

2.4.6 Boundary conditions

Solutions of the governing equations require that appropriate conditions be specified on the bounding surfaces of the computational domain. It is crucial that these conditions be modeled properly to ensure a correct solution. For the domains used for the potential flow equation three classes of boundary conditions are necessary: solid walls, far fields and the Kutta condition.

Solid surface boundaries

Solid boundaries must be modeled by a no-penetration condition that still permits slip. Such a boundary condition is satisfied by setting $\frac{\partial\phi}{\partial n} = 0$. This Neumann boundary condition is easily satisfied in our present formulation because meshes are constructed such that the faces of the computational cells are coincident with the surface. The construction of the computation cell depicted in Figure (2.2) at the surface is shown in

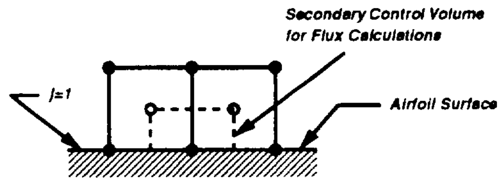


Figure 2.3: *Mesh for the Potential Flow Equation at Airfoil Surface in the Computational Plane*

Figure (2.3), and implies that the Neumann condition is satisfied by setting $V = 0$ for the faces on the boundary of the airfoil (η = constant boundary).

THE KUTTA CONDITION

In calculations with lift, the circulation has to be adjusted to satisfy the Kutta condition. In two-dimensional flows this involves only a minor complication. Along the *cut-line* (dividing line of the structured mesh leaving the trailing edge) there should be a constant jump Γ in the circulation. If we use an O-mesh this can easily be accommodated by subtracting a term $\frac{\Gamma\theta}{2\pi}$ from the potential, where θ is measured along the mesh at the outer domain and remains constant along the η lines. This does not map in the physical plane to a pure circulation component except at the outer domain. However, any necessary adjustments will be incorporated into the perturbation potential so that the governing equation is still satisfied. The perturbation potential also has the free stream component subtracted with the result that it is a single-valued function that can be solved with a periodic boundary condition along the cut-line. The value of the circulation constant Γ must be updated as part of the solution procedure. If the trailing edge is assumed sharp, it forms a singularity in the potential flow solution unless

$$\phi_\xi = \phi_{G_\xi} - \frac{\Gamma}{2\pi} + \phi_{U_{\infty_\xi}} = 0, \tag{2.28}$$

with the convention that ϕ is the total potential, ϕ_G is the perturbation potential, and ϕ_{U_∞} is the free stream potential. The term related to the circulation potential has been simplified with the assumption that an O-mesh is being employed with θ as a linear function of ξ. This condition forces the tangential velocity at the trailing edge to be zero. By enforcing (2.28) as part of the solution procedure, Γ may be updated iteratively such that the Kutta condition is satisfied.

Far field boundaries

The outer boundary is treated as a Dirichlet boundary condition in which the value of the perturbation potential is prescribed by subtracting out the free stream conditions as well as the circulation potential. The accuracy of this approximation improves as the outer boundary is extended further from the surface geometry.

2.5 Euler Solution Methodology

As in the case of the potential flow equation, a numerical discretization and solution procedure is necessary for the Euler governing equations.

2.5.1 Finite volume formulation

Again a finite volume methodology developed by Jameson, [66, 49, 63, 50, 61, 57, 51, 53, 62, 52, 60] and realized in his FLO82 computer code is used as a basis. Unfortunately, the solution procedure for the potential flow equation cannot be directly used for the Euler equations, since in the subsonic zone, the former is second order and elliptic in character, while the latter are first order and hyperbolic. This difference in equation type requires the use of an alternative solution method.

In compact form, the Euler equations (*cf.* equation 2.2) may be written as

$$\frac{d}{dt}\int_V \mathrm{w}\, dV + \int_S \mathbf{F}\cdot \mathrm{n}\, dS = 0. \tag{2.29}$$

Here, w is a vector of dependent variables, **F** is a flux matrix, and n is a surface unit normal oriented outward from the volume. These are defined in two dimensions as

$$\mathrm{w} = \left\{ \begin{array}{c} \rho \\ \rho u \\ \rho v \\ \rho E \end{array} \right\}, \quad \mathbf{F} = \begin{bmatrix} \rho u & \rho v \\ \rho u u + p & \rho u v \\ \rho v u & \rho v v + p \\ \rho H u & \rho H v \end{bmatrix}, \quad \mathrm{n} = \left\{ \begin{array}{c} \bar{n}_x \\ \bar{n}_y \end{array} \right\}.$$

First, the volume integral in equation 2.29 may be expanded for the computational

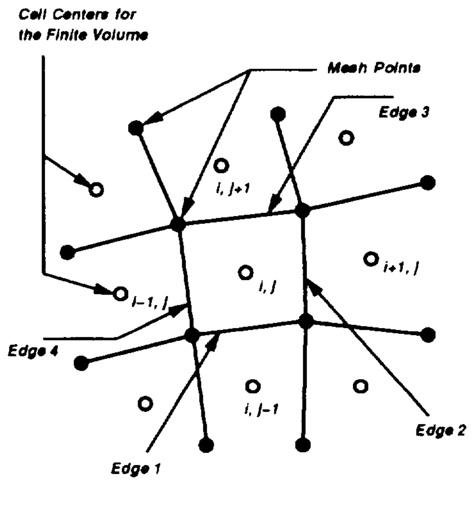


Figure 2.4: *Mesh for the Euler Flow Equations in the Physical Plane*

control volume $\mathrm{V}_{i,j}$ of cell i, j as

$$\frac{d}{dt}\int_V \mathrm{w}\, dV = \frac{d\mathrm{w}_{i,j}\mathrm{V}_{i,j}}{dt}$$

where $\mathrm{w}_{i,j}$ represents the average value of w in the cell, which may be approximated by its value at a sample point (*cf.*, Jameson and Baker [62]).

Second, the surface integral over each computational control volume is decomposed into constituent surface elements. For the case of a two-dimensional structured mesh consisting of quadrilaterals there are four such constituent (k) edges. This gives

$$\int_S \mathbf{F}\cdot\mathbf{n}\,dS = \sum_k \mathbf{F}_k\mathbf{S}_k$$

where $\mathbf{S}_k$ are the surface-projected normals and the discrete fluxes $\mathbf{F}_k$ at the cell faces are approximated by the average of the cell-centered values taken from the corresponding two adjoining cells. For example, the flux at the $k = 2$ face shown in Figure 2.4 is given by

$$\mathbf{F}_2 = \frac{1}{2}\mathbf{F}_{i+1,j} + \frac{1}{2}\mathbf{F}_{i,j}.$$

Incorporating the above approximations, equation 2.29 is transformed into a first-order ordinary differential equation for cell i,j with volume $\mathbf{V}_{i,j}$:

$$\frac{d\mathbf{w}_{i,j}}{dt} = -\frac{1}{\mathbf{V}_{i,j}}\left(\sum_k \mathbf{F}_k\mathbf{S}_k\right). \tag{2.30}$$

To determine the explicit form of the summation $\sum_k \mathbf{F}_k\mathbf{S}_k$ it is beneficial to apply the same general coordinate transformation as was applied to the potential equation. The Cartesian flux components are now written as **f** and **g** with K and J defined as in Section (2.4.1). Then by again introducing the contravariant velocity components,

$$\left\{\begin{array}{c} U \\ V \end{array}\right\} = \frac{1}{J}\begin{bmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial x}{\partial \eta} \\ -\frac{\partial y}{\partial \xi} & \frac{\partial x}{\partial \xi} \end{bmatrix}\left\{\begin{array}{c} u \\ v \end{array}\right\} = K^{-1}\left\{\begin{array}{c} u \\ v \end{array}\right\},$$

it is possible to rewrite equation (2.29) as

$$\frac{d}{dt}\int_{\bar{V}} W\,d\bar{V} + \int_{\bar{S}} F\bar{n}_\xi + G\bar{n}_\eta\,d\bar{S} = 0.$$

$\bar{V}$ and $\bar{S}$ are unit volumes and areas respectively and W, F, and G are given by

$$W = J\begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{bmatrix}, \quad F = J\begin{bmatrix} \rho U \\ \rho U u + \frac{\partial \xi}{\partial x}p \\ \rho U v + \frac{\partial \xi}{\partial y}p \\ \rho U H \end{bmatrix}, \quad G = J\begin{bmatrix} \rho V \\ \rho V u + \frac{\partial \eta}{\partial x}p \\ \rho V v + \frac{\partial \eta}{\partial y}p \\ \rho V H \end{bmatrix}.$$

It is now possible to rewrite (2.30) as

$$\begin{aligned}\frac{dW_{i,j}}{dt} = & \left\{\left(\frac{1}{2}F^{+}_{i+1,j} + \frac{1}{2}F^{+}_{i,j}\right) - \left(\frac{1}{2}F^{-}_{i,j} + \frac{1}{2}F^{-}_{i-1,j}\right)\right. \\ + & \left.\left(\frac{1}{2}G^{+}_{i,j+1} + \frac{1}{2}G^{+}_{i,j}\right) - \left(\frac{1}{2}G^{-}_{i,j} + \frac{1}{2}G^{-}_{i,j-1}\right)\right\}, \end{aligned} \tag{2.31}$$

where the $^{(+)}$ and $^{(-)}$ indicate that the evaluation of the mesh metric terms in F and G occur at the appropriate cell faces. For example, F is a function of the metrics $\frac{\partial x}{\partial \eta}$ and $\frac{\partial y}{\partial \eta}$, which for $F^{+}_{i+1,j}$ and $F^{+}_{i,j}$ are evaluated at $i+\frac{1}{2}, j$ and for $F^{-}_{i,j}$ and $F^{-}_{i-1,j}$ are evaluated at $i-\frac{1}{2}, j$.

2.5.2 TIME-STEPPING

Equation (2.31) may be written as

$$\frac{d\mathbf{w}_{i,j}}{dt} = -\mathcal{Q}(\mathbf{w}_{i,j}),$$

where $\mathcal{Q}(\mathbf{w}_{i,j})$ represents the collected residual of Euler fluxes that is balanced by the rate of change of w within the cell. Thus when equation (2.30) is applied to each cell in the computational domain, a system of first order ordinary differential equations is obtained that must be integrated to steady state. In order to suppress odd-even decoupling of the solution and possible overshoots around shock waves, equation (2.30) must be augmented with artificial dissipation, giving

$$\frac{d\mathbf{w}_{i,j}}{dt} = -\mathcal{Q}(\mathbf{w}_{i,j}) + \mathcal{D}(\mathbf{w}_{i,j}). \tag{2.32}$$

The form of $\mathcal{D}(\mathbf{w}_{i,j})$ will be discussed later in section (2.5.3). By setting

$$\mathcal{R}(\mathbf{w}_{i,j}) = +\mathcal{Q}(\mathbf{w}_{i,j}) - \mathcal{D}(\mathbf{w}_{i,j}),$$

we have a system of equations of the form

$$\frac{d\mathbf{w}_{i,j}}{dt} + \mathcal{R}(\mathbf{w}_{i,j}) = 0. \tag{2.33}$$

Equation 2.33 is in a semi-discrete form, and it must be discretized in time for numerical computations. Various formulas for discretizing equations of type $\frac{d\mathbf{w}_{i,j}}{dt} = -\mathcal{R}\,(\mathbf{w}_{i,j})$

are possible. Generating these schemes can be simplified if the solution of the differential equation is viewed as an integration. That is,

$$\mathbf{w}_{i,j}^{n+1} = \mathbf{w}_{i,j}^{n} - \int_{t_n}^{t_{n+1}} \mathcal{R}\left(\mathbf{w}_{i,j}\right) dt.$$

Because the form presented by equation (2.33) fits into a class of well investigated equations, o.d.e's., a solution procedure may be patterned off existing schemes. One such well investigated scheme, the Runge-Kutta multi-stage scheme, is chosen for this research. A complete discussion of the various aspects and choices of Runge-Kutta schemes applied to the time integration of the Euler equations is given by Jameson [51, 66, 49, 50, 57].

In this work, a modified five-stage Runge-Kutta scheme was utilized, in which the convective and dissipative parts are treated separately and in such a way as to increase the stability region of the scheme beyond that obtained with a standard Runge-Kutta scheme. It can be summarized as

$$\begin{array}{lrcl}
 & \mathbf{w}_{i,j}^{\tau_0} & = & \mathbf{w}_{i,j}^{n} \\
\tau_1, & \mathbf{w}_{i,j}^{\tau_1} & = & \mathbf{w}_{i,j}^{\tau_0} - \frac{1}{4}\Delta t_{i,j}\left\{\mathcal{Q}(\mathbf{w}_{i,j}^{\tau_0}) - \mathcal{D}(\mathbf{w}_{i,j}^{\tau_0})\right\} \\
\tau_2, & \mathbf{w}_{i,j}^{\tau_2} & = & \mathbf{w}_{i,j}^{\tau_0} - \frac{1}{6}\Delta t_{i,j}\left\{\mathcal{Q}(\mathbf{w}_{i,j}^{\tau_1}) - \mathcal{D}(\mathbf{w}_{i,j}^{\tau_0})\right\} \\
\tau_3, & \mathbf{w}_{i,j}^{\tau_3} & = & \mathbf{w}_{i,j}^{\tau_0} - \frac{3}{8}\Delta t_{i,j}\left\{\mathcal{Q}(\mathbf{w}_{i,j}^{\tau_2}) - \left[.44\mathcal{D}(\mathbf{w}_{i,j}^{\tau_0}) + .56\mathcal{D}(\mathbf{w}_{i,j}^{\tau_2})\right]\right\} \\
\tau_4, & \mathbf{w}_{i,j}^{\tau_4} & = & \mathbf{w}_{i,j}^{\tau_0} - \frac{1}{2}\Delta t_{i,j}\left\{\mathcal{Q}(\mathbf{w}_{i,j}^{\tau_3}) - \left[.44\mathcal{D}(\mathbf{w}_{i,j}^{\tau_0}) + .56\mathcal{D}(\mathbf{w}_{i,j}^{\tau_2})\right]\right\} \\
\tau_5, & \mathbf{w}_{i,j}^{\tau_5} & = & \mathbf{w}_{i,j}^{\tau_0} - \Delta t_{i,j}\left\{\mathcal{Q}(\mathbf{w}_{i,j}^{\tau_4}) - \left(.56\left[.44\mathcal{D}(\mathbf{w}_{i,j}^{\tau_0}) + .56\mathcal{D}(\mathbf{w}_{i,j}^{\tau_2})\right] + .44\mathcal{D}(\mathbf{w}_{i,j}^{\tau_4})\right)\right\} \\
 & \mathbf{w}_{i,j}^{n+1} & = & \mathbf{w}_{i,j}^{\tau_5},
\end{array} \tag{2.34}$$

where the various weighting coefficients have been optimized through numerical investigations by Martinelli [80]. Pseudo-time-like stages in the scheme are denoted τ_0 - τ_5. Note that the Euler fluxes are evaluated at each stage while the dissipation fluxes are only evaluated at every other stage. Since the work incurred to evaluate the dissipation fluxes is roughly equal to that of evaluating the Euler fluxes, there is a considerable savings by evaluating the dissipation fluxes at only every other stage provided that this does not hinder convergence.[1] Another advantage of this scheme is that due to its damping of high frequency modes it is effective for use in conjunction with multigrid techniques.[2]

In order to determine the proper time step limit it is helpful to estimate the speed and direction of wave propagation through each cell in the computational domain. Consider the Euler equations, in equation 2.29, rewritten as

$$\frac{\partial \mathbf{w}}{\partial t} + \frac{\partial \mathbf{f}(\mathbf{w})}{\partial x} + \frac{\partial \mathbf{g}(\mathbf{w})}{\partial y} = \mathbf{0}, \tag{2.35}$$

where

$$\mathbf{w} = \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{Bmatrix}, \quad \mathbf{f} = \begin{Bmatrix} \rho u \\ \rho u u + p \\ \rho v u \\ \rho H u \end{Bmatrix}, \quad \mathbf{g} = \begin{Bmatrix} \rho v \\ \rho u v \\ \rho v v + p \\ \rho H v \end{Bmatrix}.$$

Expanding the spatial derivatives yields a non-conservative system of equations, which are nonetheless useful for assessing properties of the previously introduced integral and conservative differential formulations (*cf.* equations (2.2) and (2.3)):

$$\frac{\partial \mathbf{w}}{\partial t} + \mathbf{A}_1(\mathbf{w})\frac{\partial \mathbf{w}}{\partial x} + \mathbf{A}_2(\mathbf{w})\frac{\partial \mathbf{w}}{\partial y} = 0. \tag{2.36}$$

[1]See section 2.5.3.
[2]See section 2.5.4.

The matrices $\mathbf{A_1}$ and $\mathbf{A_2}$ are the flux Jacobians, and can be explicitly written as

$$\mathbf{A_1}(\mathbf{w}) = \frac{\partial \mathbf{f}(\mathbf{w})}{\partial \mathbf{w}} =$$

$$\begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{w_2^2}{w_1^2} + \frac{\gamma-1}{2}\frac{w_2^2+w_3^2}{w_1^2} & (3-\gamma)\frac{w_2}{w_1} & (1-\gamma)\frac{w_3}{w_1} & (\gamma-1) \\ -\frac{w_2 w_3}{w_1^2} & \frac{w_3}{w_1} & \frac{w_2}{w_1} & 0 \\ -\frac{w_2}{w_1^2}(w_4+p) + \frac{w_2}{w_1}\frac{\gamma-1}{2}\frac{w_2^2+w_3^2}{w_1^2} & \frac{1}{w_1}(w_4+p) + (1-\gamma)\frac{w_2^2}{w_1^2} & (1-\gamma)\frac{w_2 w_3}{w_1^2} & \gamma\frac{w_2}{w_1} \end{bmatrix},$$

$$\mathbf{A_2}(\mathbf{w}) = \frac{\partial \mathbf{g}(\mathbf{w})}{\partial \mathbf{w}} =$$

$$\begin{bmatrix} 0 & 0 & 1 & 0 \\ -\frac{w_2 w_3}{w_1^2} & \frac{w_3}{w_1} & \frac{w_2}{w_1} & 0 \\ -\frac{w_3^2}{w_1^2} + \frac{\gamma-1}{2}\frac{w_2^2+w_3^2}{w_1^2} & (1-\gamma)\frac{w_2}{w_1} & (3-\gamma)\frac{w_3}{w_1} & (\gamma-1) \\ -\frac{w_3}{w_1^2}(w_4+p) + \frac{w_3}{w_1}\frac{\gamma-1}{2}\frac{w_2^2+w_3^2}{w_1^2} & (1-\gamma)\frac{w_2 w_3}{w_1^2} & \frac{1}{w_1}(w_4+p) + (1-\gamma)\frac{w_3^2}{w_1^2} & \gamma\frac{w_3}{w_1} \end{bmatrix}.$$

For a perfect gas, equation (2.36) is hyperbolic. Thus, wave-like solutions exist and propagate in some direction defined by κ, say. Although the exact propagation direction of various waves may be difficult to determine, by making a simple approximation for κ, a reasonable time step scale can be derived. In this research κ is dimensional and assumed to be related to the area of a computational face such that

$$\begin{aligned} \kappa_x &= \mathrm{S}_x \\ \kappa_y &= \mathrm{S}_y \end{aligned}$$

where S_x and S_y are components of the directed surface areas. With this approximation it is then possible to derive a time step limit by noting that properties of the wave-like solutions are governed by a linear combination of the flux Jacobians such that

$$\tilde{\mathbf{A}} = \kappa_x \mathbf{A_1} + \kappa_y \mathbf{A_2},$$

where

$$\boldsymbol{\kappa} = \kappa_x \hat{\imath} + \kappa_y \hat{\jmath}.$$

With this, the eigenvalues of the matrix $\tilde{\mathrm{A}}$ are

$$\begin{aligned} \tilde{\lambda}_1 = \tilde{\lambda}_2 &= \mathbf{v}\cdot\boldsymbol{\kappa}, \\ \tilde{\lambda}_3 &= \mathbf{v}\cdot\boldsymbol{\kappa} + c\,||\boldsymbol{\kappa}||, \\ \tilde{\lambda}_4 &= \mathbf{v}\cdot\boldsymbol{\kappa} - c\,||\boldsymbol{\kappa}||. \end{aligned}$$

The spectral radius is defined as the largest eigenvalue of a given matrix. An estimate of the spectral radius for discrete point i,j, given by $\varrho_{i,j}$ is utilized as a *characteristic scaling quantity* for both the artificial dissipation (*cf.*, section 2.5.3) and the time step. $\varrho_{i,j}$ is formed as the sum

$$|\mathbf{v}\cdot\boldsymbol{\kappa}| + c\,||\boldsymbol{\kappa}||$$

for the different coordinate directions. Finally, a conservative time scale can be set as

$$\Delta t_{i,j} = \frac{\mathbf{V}_{i,j}}{\varrho_{i,j}}.$$

The admissible time step limit is defined as $\tilde{\Delta} t_{i,j} = \lambda \frac{\mathbf{V}_{i,j}}{\varrho_{i,j}}$, where λ is the maximum Courant-Friedrich-Lewey (CFL) condition number allowed by the time stepping scheme. The time step scale is used both to scale the dissipation—see section (2.5.3)—and to set the time step limit used to integrate the solution. The CFL number does not scale the dissipation since this would mean the final solution would be dependent upon the chosen time step. Note that $\bar{\Delta t}$ is a local time step limit and is scaled to the local flow conditions as well as the cell size, thus rendering all transient solutions non-physical.

2.5.3 Artificial Dissipation

For the finite volume discretization of the Euler equations presented in section (2.5.1), another mode of instability is also possible. When the time dependent term is dropped and the scheme is expanded for a uniform Cartesian mesh, it reduces to simple central differencing with a four point stencil (the center term is zero). This implies, at least for this example, that two separate superimposed solutions can arise resembling a checkerboard. This odd-even decoupling can be present irrespective of the smoothness of the flow field. To obviate this type of difficulty, a weak dissipation term must be added everywhere in the solution that effectively couples all the points. The best manner in which artificial dissipation should be constructed for the Euler equations remains an open issue. In this research, the dissipative residual $\mathcal{D}_{i,j}$ for cell i,j is formed from a blend of two separate dissipations (*cf.* Jameson et al. [66, 51]), $\mathcal{D}^2_{i,j}$ and $\mathcal{D}^4_{i,j}$:

$$\mathcal{D}_{i,j} = \mathcal{D}^2_{i,j} - \mathcal{D}^4_{i,j}.$$

They are defined as[3]

$$\begin{aligned}\mathcal{D}^2_{i,j} &= \mathrm{d}^2_{i+\frac{1}{2},j} - \mathrm{d}^2_{i-\frac{1}{2},j} + \mathrm{d}^2_{i,j+\frac{1}{2}} - \mathrm{d}^2_{i,j-\frac{1}{2}}, \\ \mathcal{D}^4_{i,j} &= \mathrm{d}^4_{i+\frac{1}{2},j} - \mathrm{d}^4_{i-\frac{1}{2},j} + \mathrm{d}^4_{i,j+\frac{1}{2}} - \mathrm{d}^4_{i,j-\frac{1}{2}}.\end{aligned}$$

The terms on the right all have a similar form. For example,

$$\begin{aligned}\mathrm{d}^2_{i+\frac{1}{2},j} &= \varepsilon^2_{i+\frac{1}{2},j}\left(\mathbf{w}_{i+1,j} - \mathbf{w}_{i,j}\right), \\ \mathrm{d}^4_{i+\frac{1}{2},j} &= \varepsilon^4_{i+\frac{1}{2},j}\left(\mathbf{w}_{i+2,j} - 3\mathbf{w}_{i+1,j} + 3\mathbf{w}_{i,j} - \mathbf{w}_{i-1,j}\right).\end{aligned}$$

[3]The dissipation for the energy (the fourth entry in the $\mathbf{w}$ vector) is computed for ρH rather than ρE since ρH is the convected quantity. This also ensures that for homenergic flow ($\frac{\partial H}{\partial t}$ and ∇H both zero), H is constant everywhere in the computed steady-state solution.

The two different dissipation flux terms serve different roles in the solution stabilization procedure. The first derivative dissipation flux, d^2, is used to damp oscillation around large gradient regions such as shock waves, and it is suitably scaled by the normalized second difference of the pressure. The third derivative dissipation flux, d^4, is used to eliminate possible odd-even decoupling in smooth flow regions and can be appropriately scaled by a scalar coefficient outside any large gradient regions. However, since $\mathcal{D}^4_{i,j}$ can induce oscillations near shock waves , and in smooth regions $\mathcal{D}^2_{i,j}$ degrades solution accuracy, the two dissipation terms are blended to obtain optimum results. The dissipation coefficients are therefore adapted to the flow and given for the $i, j+\frac{1}{2}$ face by

$$
\begin{aligned}
\varepsilon^2_{i,j+\frac{1}{2}} &= \frac{\mu^2 \mathrm{V}_{i,j+\frac{1}{2}}}{\Delta t_{i,j}} \max\left(\nu^j_{i+2,j}, \nu^j_{i+1,j}, \nu^j_{i,j}, \nu^j_{i-1,j}\right), \\
\varepsilon^4_{i,j+\frac{1}{2}} &= \max\left(0, \left[\frac{\mu^4 \mathrm{V}_{i,j+\frac{1}{2}}}{\Delta t_{i,j}} - \varepsilon^2_{i,j+\frac{1}{2}}\right]\right),
\end{aligned}
$$

with the coefficients for the other faces constructed similarly. The $\nu_{i,j}$ term acts as a limiter for $\mathcal{D}^2_{i,j}$ and is defined independently for each coordinate direction as

$$
\begin{aligned}
\nu^i_{i,j} &= \left|\frac{p_{i+1,j} - 2p_{i,j} + p_{i-1,j}}{p_{i+1,j} + 2p_{i,j} + p_{i-1,j}}\right| \quad \text{in the } i \text{ direction} \\
\nu^j_{i,j} &= \left|\frac{p_{i,j+1} - 2p_{i,j} + p_{i,j-1}}{p_{i,j+1} + 2p_{i,j} + p_{i,j-1}}\right| \quad \text{in the } j \text{ direction.}
\end{aligned}
$$

On a regular Cartesian mesh with ε^2 and ε^4 equal to unity, $\mathcal{D}^2_{i,j}$ and $\mathcal{D}^4_{i,j}$ represent the undivided Laplacian and bi-harmonic of w. The factors ε^2 and ε^4 are introduced to scale the artificial dissipation properly as well as limit its effects. Note that V is the volume of the computational cell, and $\frac{1}{\Delta t_{i,j}} = \frac{\varrho_{i,j}}{\mathrm{V}_{i,j}}$ is the quotient of the characteristic scaling quantity (*i.e.*, spectral radius) to this volume. Terms μ^2 and μ^4 represent dimensionless coefficients which may be selected by the user.

Dimensional analysis of $\mathcal{D}^2_{i,j}$ and $\mathcal{D}^4_{i,j}$ offers insight into how these quantities scale in comparison with the Euler residual $\mathcal{Q}_{i,j}$ as the mesh spacing is decreased. Denoting L as length and T as time, $\frac{\varrho_{i,j}}{\mathrm{V}_{i,j}}$ has dimensions of $\left(\frac{L}{T}\right)\frac{1}{L}$ where the term in parentheses represents a characteristic speed. The dimensions of the representative *undivided*

Laplacian and bi-harmonic difference operators may be written as $\bar{L}^2\left(\frac{1}{L^2}\right)$ and $\bar{L}^4\left(\frac{1}{L^4}\right)$ respectively. It follows that the dimensions of the dissipation coefficients with respect to *divided* difference operators would be the product of a speed with a length or a length cubed. Defining $\hat{h}$ to be a characteristic length of a computational control volume, ultimately $\mathcal{D}^2_{i,j}$ will scale as $O(\hat{h})$ and $\mathcal{D}^4_{i,j}$ will scale as $O(\hat{h}^3)$. However the coefficient $\varepsilon^2_{i,j+\frac{1}{2}}$ will also scale as $O(\hat{h}^2)$ in smooth regions of the flow. Thus the entire artificial diffusion will scale as $O(\hat{h}^3)$ in smooth regions of the flow but reverts to $O(\hat{h})$ around discontinuities.

2.5.4 Convergence acceleration

Like the basic scheme for the potential flow equation (see Sections 2.4.1–2.4.4), the scheme discussed thus far for the Euler equations in Sections 2.5.1–2.5.3 converges at a less than desirable rate. The scheme can be accelerated by using specific techniques suited to this task. Three such techniques are implemented for the Euler formulation that significantly increase the convergence rate of the numerical procedure.

Enthalpy damping

One device for accelerating the convergence of the Euler formulation is the augmentation of the governing equations with a term that is proportional to the difference between the total enthalpy, H, and its free stream value H_∞. For steady state, inviscid and adiabatic flows with constant uniform free stream, $H = H_\infty$ throughout the domain. Thus a forcing term proportional to $H - H_\infty$ will not alter the steady state solution so long as the discretization of the Euler fluxes and implementation of the artificial dissipation preserves conservation of total enthalpy. A check of the finite volume formulation in section (2.5.1) and the dissipation scheme in section (2.5.3) shows that conservation of H is indeed maintained in the present method. The augmented governing equations are thus written in differential form as

$$\frac{\partial \rho}{\partial t} + \nabla\cdot\rho\mathbf{v} + \tilde{\alpha}\rho(H - H_\infty) = 0 \qquad (2.37a)$$

$$\frac{\partial \rho \mathbf{v}}{\partial t} + \nabla\cdot(\rho \mathbf{v}\mathbf{v}) + \nabla p + \tilde{\alpha}\rho \mathbf{v}(H - H_\infty) = 0 \qquad (2.37b)$$

$$\frac{\partial \rho E}{\partial t} + \nabla\cdot\rho H \mathbf{v} + \tilde{\alpha}\rho E(H - H_\infty) = 0, \qquad (2.37c)$$

where $\tilde{\alpha}$ is a user-provided input to control the magnitude of the enthalpy damping. A rigorous presentation and justification of adding these terms to accelerate the convergence has been given by Jameson et al. [49, 66]. Here, it is sufficient to state that the additional terms act to damp transients and have proven to be efficient in accelerating convergence.

IMPLICIT RESIDUAL SMOOTHING

Implicit residual smoothing can significantly increase the CFL number at which the scheme remains stable by efficiently increasing the stencil of support for the scheme. In the potential flow formulation, the equivalent of residual smoothing was built into the iteration scheme through the alternating direction algorithm. For the explicit multistage time-stepping scheme used for the Euler equations, residual smoothing is applied in product form to reduce computational cost and can be written

$$(1 - \epsilon_i \delta_x^2)(1 - \epsilon_j \delta_y^2)\bar{\mathcal{R}}_{i,j} = \mathcal{R}_{i,j},$$

where ϵ_i and ϵ_j control the level of smoothing, and $\bar{\mathcal{R}}_{i,j}$ is the updated value of the residual that is obtained by solving the equation implicitly in each coordinate direction in series. Each scalar component of w can be solved for independently by the use of a computationally inexpensive tridiagonal solver. This smoothing can work concurrently and efficiently with the multistage scheme by application at every other stage. A rigorous discussion of the stability character and overall benefit of this acceleration device is provided by Jameson and Baker in [52, 60].

MULTIGRID ACCELERATION

The third technique of accelerating the convergence of the Euler formulation used in this research is the multigrid method. The advantages of using a multigrid method were discussed in Section 2.4.5 where it was applied to the potential flow equation.

Here it is applied to the Euler equations using a similar technique. As for the potential flow equation, a coarser mesh is introduced by eliminating every other point in a fine mesh. The full multistage time stepping scheme is used to drive the solution on each mesh.

Once a full time step is calculated on the finest mesh in the sequence, the multigrid cycle starts by transferring both the residuals and the flow variables to the next coarser mesh. The values of the flow variables are transferred conservatively with

$$\mathbf{w}_c^n = \frac{\sum(\mathbf{V}_f \mathbf{w}_f^{n+1})}{\mathbf{V}_c}, \tag{2.38}$$

where it is noted that the coarse mesh flow variables $\mathbf{w}_c^n$ have yet to be updated and thus are still at state n. For two dimensions the sums extend over the four fine mesh cells that are agglomerated in order to create the single coarse cell. The corresponding residuals are transferred with the simple summation

$$\mathcal{R}_c(\mathbf{w}_f) = \sum \mathcal{R}_f(\mathbf{w}_f), \tag{2.39}$$

where $\mathcal{R}_f(\mathbf{w}_f)$ stands for the fine mesh residual calculated on the fine mesh and $\mathcal{R}_c(\mathbf{w}_f)$ is the collected fine mesh residual on the coarse mesh. Now a forcing function is defined as

$$\mathcal{P}_c = \mathcal{R}_c\left(\mathbf{w}_f^{n+1}\right) - \mathcal{R}_c(\mathbf{w}_c^n), \tag{2.40}$$

where $\mathcal{R}_c(\mathbf{w}_c^n)$ is the residual calculated on the coarse mesh with the collected w from the fine mesh. To update the solution on a coarser mesh the multistage time stepping scheme is reformulated as

$$\begin{array}{lrcl}
 & \mathbf{w}_c^{\tau 0} & = & \mathbf{w}_c^{n} \\
\nu = 1 & \mathbf{w}_c^{\tau 0} & = & \mathbf{w}_c^{\tau 0} - \frac{1}{4}\Delta t\left\{\mathcal{R}_c\left(\mathbf{w}_c^{\tau 0}\right) + \mathcal{P}_e\right\} \\
\nu = 2 & \mathbf{w}_c^{\tau 1} & = & \mathbf{w}_c^{\tau 0} - \frac{1}{6}\Delta t\left\{\mathcal{R}_c\left(\mathbf{w}_c^{\tau 1}\right) + \mathcal{P}_e\right\} \\
\nu = 3 & \mathbf{w}_c^{\tau 3} & = & \mathbf{w}_c^{\tau 0} - \frac{3}{8}\Delta t\left\{\mathcal{R}_c\left(\mathbf{w}_c^{\tau 2}\right) + \mathcal{P}_e\right\} \\
\nu = 4 & \mathbf{w}_c^{\tau 4} & = & \mathbf{w}_c^{\tau 0} - \frac{1}{2}\Delta t\left\{\mathcal{R}_c\left(\mathbf{w}_c^{\tau 3}\right) + \mathcal{P}_e\right\} \\
\nu = 5 & \mathbf{w}_c^{\tau 5} & = & \mathbf{w}_c^{\tau 0} - \Delta t\left\{\mathcal{R}_c\left(\mathbf{w}_c^{\tau 4}\right) + \mathcal{P}_e\right\} \\
 & \mathbf{w}_c^{n+1} & = & \mathbf{w}_c^{\tau 5},
\end{array} \tag{2.41}$$

where $\mathcal{R}_c\left(\mathbf{w}_c^{\tau 0 - \tau 4}\right)$ are calculated in the same manner as that for the five stage scheme in equation (2.34). In the first stage on the coarse mesh, the residual calculated on the coarse mesh is replaced by the residual collected from the fine mesh, with the result that the evolution of the solution on the coarse mesh is driven by the residual on the fine mesh. The multigrid cycle continues by repeating the collection and the time stepping procedures of equations (2.38–2.41) on each successive coarser mesh. Once a time step has been completed on the coarsest mesh in the sequence, the correction calculated on each mesh is passed back to the next finer mesh in succession by bilinear interpolation. Because the solution on each coarser mesh is driven by the residuals on the finest mesh, the final solution on the finest mesh is independent of the solution details on any of the coarser meshes. A more complete treatment of the implementation and convergence of multigrid acceleration for multistage time stepping of the Euler equations is given in [50, 61, 51, 52].

2.5.5 Boundary conditions

Like the potential flow solution method, appropriate conditions must be specified on the boundaries of the computational domain. For the computational domains used for the solution of the Euler equations, two classes of boundary conditions are necessary: solid walls, and far fields.

Solid surface boundaries

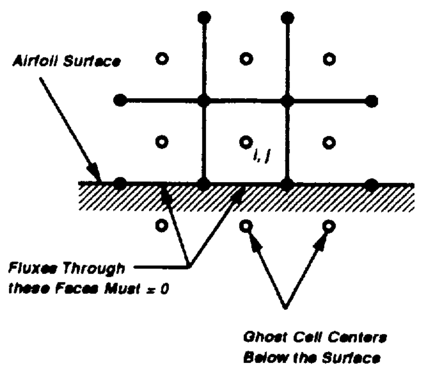

Airfoil Surface
i, j
Fluxes Through these Faces Must = 0
Ghost Cell Centers Below the Surface

Figure 2.5: *Mesh for the Euler Equation at the Airfoil Surface in the Computational Plane*

Solid boundaries must be modeled by a no-penetration condition that still permits slip. For arbitrary meshes, this is most conveniently done by first transforming the differential form of the governing equations to be coincident with the body-fitted mesh coordinates ξ and η. Then if a constant $\eta = 0$ fits the surface, the Euler fluxes in (2.35) can be transformed to give

$$F = y_\eta \mathbf{f} - x_\eta \mathbf{g} \text{ and } G = x_\xi \mathbf{g} - y_\xi \mathbf{f},$$

where F and G are the fluxes tangential and normal to the surface respectively. Figure 2.5 shows that at the surface all convective terms must be dropped from the definition of G, giving

$$G = x_\xi p - y_\xi p.$$

This formulation has the advantage that most of the flow field quantities never have to be prescribed at the boundary. The only information needed is an estimate for the pressure at the wall. The value of p_η for the cell centers immediately off the surface can be estimated from the interior points by calculating the pressure gradient due to curvature of the streamlines as proposed by Rizzi [98]:

$$(x_\xi^2 + y_\xi^2)p_\eta = (x_\xi x_\eta + y_\xi y_\eta)p_\xi + \rho(y_\eta u - x_\eta v)(v x_{\xi\xi} - u y_{\xi\xi}). \tag{2.42}$$

With this estimate it is then possible to extrapolate the value of p at the wall.

FAR FIELD BOUNDARIES

For the far field, a characteristics-based boundary condition is imposed such that outgoing waves are extrapolated from the interior while incoming waves are specified in a manner consistent with conditions imposed by the free stream. In order to accomplish this, Riemann invariants are introduced which correspond to the one-dimensional flow normal to the boundary. Fixed and extrapolated Riemann invariants may then be defined for the incoming and outgoing characteristics respectively as,

$$\bar{R}_\infty = \mathbf{v}_\infty \cdot \mathbf{n} - \frac{2c_\infty}{\gamma - 1}, \qquad \bar{R}_e = \mathbf{v}_e \cdot \mathbf{n} + \frac{2c_e}{\gamma - 1}$$

where ∞ refers to free stream values and e refers to those extrapolated from the interior of the computational domain. These incoming and outgoing Riemann invariants may then be added and subtracted to give,

$$\mathbf{v}_a \cdot \mathbf{n} = \frac{1}{2}\left(\bar{R}_e + \bar{R}_\infty\right), \qquad c_a = \frac{\gamma - 1}{4}\left(\bar{R}_e - \bar{R}_\infty\right)$$

where a represents the actual quantity at the boundary. For outflow boundaries the tangential component of the velocity is taken as the extrapolated values giving

$$\mathbf{v}_a = \mathbf{v}_e \cdot \mathbf{n}_t + \mathbf{v}_a \cdot \mathbf{n},$$

whereas for inflow boundaries, it is taken from the free stream values giving

$$\mathbf{v}_a = \mathbf{v}_\infty \cdot \mathbf{n}_t + \mathbf{v}_a \cdot \mathbf{n}.$$

Here $\mathbf{n}_t$ is the unit normal in the tangential direction to the outer boundary. The scheme is completed by extrapolating the entropy at an outflow boundary or setting it equal to the free stream value at an inflow boundary. This choice and the definition of c_a together fixes the values of density, energy and pressure at the boundary. For a complete treatment, see [60].

Chapter 3

NUMERICAL OPTIMIZATION METHODS

The use of numerical optimization in aerodynamic shape design is not unique to this research as has been discussed in *Chapter 1*. However, the choice of which numerical optimization algorithm to employ is of considerable importance. The optimization algorithm should be both robust and efficient. The choice here is limited to those based on gradient information since the primary advantage of using control theory is to provide inexpensive gradient information. Even so, this restriction should not be viewed as excessive, since non-gradient based methods, such as genetic algorithms [16], are typically expensive alternatives because they require a very large number of objective function evaluations. Gradient-based numerical optimization methods are based on the assumption that the cost function depends smoothly on the design variables. They can be classified into two distinct categories: constrained and unconstrained methods. Both of these methods require the gradient of the objective function with respect to the design variables. In addition, constrained methods typically also require the gradient of the constraint functions which must be satisfied during the design process. In theory, it is possible to use control theory to provide this additional constraint gradient information. If the adjoint variable formulation is used, and assuming that the constraints refer to quantities that involve flow variables, such as pitching moment, C_m, this method amounts to obtaining the solution to a separate adjoint equation for each constraint equation that must be satisfied. If the constraints are independent of flow field variables, such as geometric constraints, no additional calculations may be needed since the constraint gradients can be obtained by direct analysis.

For the scope of this research, it is assumed that all problems will contain a single, possibly composite, objective function, with no additional flow variable dependent constraints. Thus, if an aerodynamic constraint such as C_m is necessary, it must be posed as a penalty constraint which is added directly to the objective function. While this approach may have limitations in the practical environment, it is still more than adequate to demonstrate the strength of control theory based optimization. By presupposing that the problem of interest can be cast in a single objective function, with at most penalty constraints, it is possible to use an unconstrained numerical optimization algorithm. It is accepted that the more high order information provided to a gradient based numerical optimization algorithm, without a significant increase in computational cost, the greater the likelihood of success.[1] Here, a summary of existing methods is given with special focus on the method employed in this research.

Numerical optimization terminology used in this thesis is as follows: The *objective function*, I, is the scalar figure of merit used to judge whether or not one design is better than another. It is also equivalent to the *cost function* defined from the controls point of view. The *design variables*, u, form a vector set of independent parameters which affect the design. The *design space* is that space spanned by the value of the objective function with respect to the variation of the design variables. For example, where the objective function to be minimized has only two independent design variables (u_1 and u_2), the design space is then the surface defined by the value of the objective function, $I(u_1, u_2)$, for all allowable combinations of u_1 and u_2.

The reliability of all gradient methods depends on the presumption that the design space is *smooth*. Smoothness implies that the gradient and possibly the higher-order derivatives of the objective function with respect to the design variables are reasonably definable and continuous. The objective function should be carefully defined to ensure that this is the case. For example, in the presence of shock waves integrated quantities such as the drag depend continuously on the shape because the shock location and strength vary continuously, although the local pressure in the vicinity of the shock is subject to a sudden change as the shock moves. If a quadratic design procedure is

[1]For a detailed description of the various optimization algorithms available and their application to aerodynamic design, see Gill, Murray and Wright [31] and Reuther [90] respectively.

employed, an additional condition that the design space is *unimodal* should also be satisfied. Unimodal functions are those that experience one extremum in the design space and not several local extrema. Generally, unimodal functions are quadratic functions, or are at least strongly approximated by quadratic functions near their extremum. If a problem displays discontinuous (i.e. *non-smooth*), highly nonlinear, or extensively multi-modal properties in the design space, then a non-gradient method may be the best choice to find the global extremum. If, on the other hand, only a local extremum is sought, and so long as the design space is smooth, then even a finite-difference based gradient method should be more efficient than a genetic algorithm.

3.1 General Unconstrained Gradient-Based Algorithm

Gradient-based methods are constructed from a simple Taylor series expansion of the objective function about a point subject to the design variables:

$$I(\mathbf{u} + \bar{\lambda}\mathbf{p}) = I(\mathbf{u}) + \bar{\lambda}\mathcal{G}(\mathbf{u})^T\mathbf{p} + \frac{1}{2}\bar{\lambda}^2\mathbf{p}^T\mathcal{H}(\mathbf{u})\mathbf{p} + O(\bar{\lambda}^3) + \ldots \tag{3.1}$$

where $\mathcal{G}(\mathbf{u})$ is the gradient vector, $\mathcal{H}(\mathbf{u})$ represents the matrix of partial second derivatives (the Hessian matrix), $\mathbf{p}$ represents the search direction, and $\bar{\lambda}$ represents the stepsize. These gradient-based algorithms have in common not only a Taylor series foundation, but also a basic algorithm structure:

General Unconstrained Gradient-Based Numerical Optimization

1. Calculate the objective function and the gradient at the current point in the design space.
2. Calculate the search direction based in part on the objective function and gradient information.
3. Determine the optimum or near optimum step length along the search direction.
4. Check for convergence and if necessary return to step (1).

Although one variation of this approach effectively reverses the order and combines step (2) and (3) (thrust region methods), the basic iterative nature of the algorithm remains. The presentation of this chapter follows the steps of this general algorithm. It is presumed that the flow solutions and adjoint solutions that are required for step (1) are available. Sections (3.2 - 3.5) treat steps (2) and section (3.6) addresses step (3).

3.2 Steepest Descent

The simplest means of exploiting gradient information in numerical optimization is the steepest descent algorithm. The essence of the technique is to use the direction which has the steepest negative gradient as an estimate for the direction in which the minimum lies. It is assumed that the optimization problem is posed such that a minimum rather than a maximum is sought. This assumption creates no loss of generality since any maximization problem is easily re-posed as a minimization problem. Provided that the initial objective function value and the gradient are available, the minimum is then assumed to lie in the search direction $\mathbf{p} = -I'(\mathbf{u}) = -\mathcal{G}$. One virtue of the steepest descent method is the trivial cost of calculating the search direction. Thus even as the degree of u (the number of design variables) increases, the cost of calculating p remains insignificant; this is not necessarily true for the high order methods discussed later. The next step involves searching in this direction for a minimum. For problems where the design space is perfectly quadratic, that is where $\mathcal{H}$ is constant and positive definite and the higher order terms can be neglected, the stepsize to the minimum along the steepest descent direction can be directly calculated. For this case, the goal is to minimize, with respect to $\bar{\lambda}$,

$$I(\mathbf{u}+\bar{\lambda}\mathbf{p}) = I(\mathbf{u}) + \bar{\lambda}\mathcal{G}(\mathbf{u})^T\mathbf{p}. \tag{3.2}$$

One choice of p which ensures a reduction in I provided $\bar{\lambda}$ is sufficiently small and positive, is $\mathbf{p} = -\mathcal{G}$. This direction guarantees the greatest reduction in I for a small constant $\bar{\lambda}$. With the steepest descent direction chosen as the search direction, a one-parameter search in $\bar{\lambda}$ may be performed in order to find a minimum along p. In many instances, a separate algorithm is used to determine an appropriate $\bar{\lambda}$. A discussion of such an algorithm is provided in Section (3.6). However for specific functional forms of I, $\bar{\lambda}$ can be obtained by alternative means, as illustrated next.

Suppose that the objective function is of the quadratic form

$$I(\mathbf{u}) = \frac{1}{2}\mathbf{u}^T A\mathbf{u} - b^T\mathbf{u} + \mathbf{c}, \tag{3.3}$$

where A is a symmetric positive definite matrix. It can readily be shown that minimizing this function is equivalent to solving the linear algebra problem

$$A\mathbf{u} = b. \tag{3.4}$$

An important property of the quadratic form (3.3) is that it forms a paraboloid that is concave up in every direction due to the positive definiteness of A. Thus the function is ensured to be unimodal. For this particular form of the objective function, much insight into the steepest descent procedure can be gained. The search direction is given by

$$\mathbf{p}_i = -\mathcal{G}_i = b - A\mathbf{u}_i.$$

It is desired to find the minimum along $\mathbf{p}$ with respect to λ. This is given by solving

$$\begin{aligned} 0 = \frac{d}{d\lambda}\left(I(\mathbf{u})_{i+1}\right) &= \frac{dI_{i+1}}{d\mathbf{u}_{i+1}}\frac{d\mathbf{u}_{i+1}}{d\lambda} \\ &= \mathcal{G}_{i+1}^T\frac{d\mathbf{u}_{i+1}}{d\lambda} \\ &= -\mathcal{G}_{i+1}^T\mathcal{G}_i \\ &= \left(b - A\mathbf{u}_{i+1}\right)^T\mathcal{G}_i \\ &= \left(b - A\left(\mathbf{u}_i - \lambda_i\mathcal{G}_i\right)\right)^T\mathcal{G}_i \\ &= \left(b - A\mathbf{u}_i\right)^T\mathcal{G}_i + \lambda_i\left(A\mathcal{G}_i\right)^T\mathcal{G}_i \\ \lambda_i &= \frac{\mathcal{G}_i^T\mathcal{G}_i}{\mathcal{G}_i^T A\mathcal{G}_i}, \end{aligned}$$

giving an explicit formula for λ in terms of the current gradient.

This derivation shows that minimizing the function along $\mathbf{p}_i$ is equivalent to finding the point along $\mathbf{p}_i$ such that the new gradient is orthogonal to the last, since $\mathcal{G}_{i+1}^T\mathcal{G}_i = 0$. If it happens that the paraboloid defined by $I(\mathbf{u})$ is stretched along an axis, the convergence of the steepest descent algorithm will become poor; see Figure (3.1).

The poor behavior results from the fact that each new search direction is orthogonal to the last, and hence a fine-toothed zig-zag path must be followed to traverse the base of any "canyon" present in the design space. While progress toward a global minimum continues, the rate of convergence can become extremely dilatory due to the infinitesimal steps taken along each successive $\mathbf{p}_i$. This degenerate behavior of the steepest descent algorithm is unfortunately not confined to quadratic objective functions, but instead typifies its behavior on more general functions. Clearly, a more robust iterative algorithm should be feasible from knowledge of the gradient and the objective function.

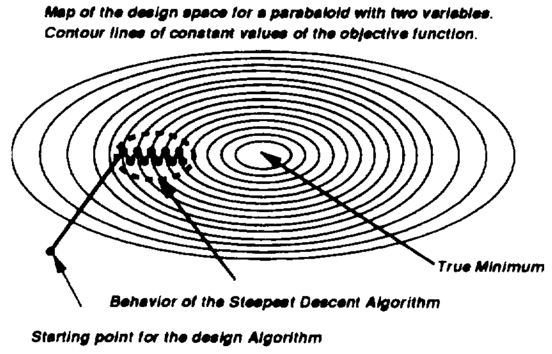

Map of the design space for a paraboloid with two variables.
Contour lines of constant values of the objective function.
True Minimum
Behavior of the Steepest Descent Algorithm
Starting point for the design Algorithm

Figure 3.1: *Performance of the Steepest Descent Method on a Paraboloid*

3.3 CONJUGATE GRADIENT

The slow convergence behavior of the steepest descent algorithm can be mitigated to some extent by the use of the conjugate gradient algorithm. Following the development of steepest descent, it is again initially assumed that the objective function is of the form of equation (3.3). Motivated by the fact that the steepest descent algorithm tends to repeat previous search directions, a set of $\bar{n}$ search directions are defined that span the design space[2]. Thus, $\mathbf{p}_i$ is no longer defined by the negative gradient but is chosen to be linearly independent of all previous searches. Thus each iteration should eliminate one component from the design space and after $\bar{n}$ iterations the minimum should be obtained. For this approach to work, $\mathbf{p}_i$ must satisfy the property that

$$\mathbf{p}_j^T A \mathbf{p}_i = 0 \text{ for } j \neq i, \tag{3.5}$$

which is equivalent to saying that they are conjugate with respect to A. Given that a set of conjugate search directions exist, it is necessary to find the minimum along each such direction just as in the steepest descent algorithm. The directional derivative along $\mathbf{p}_i$ is set to zero, giving

$$\begin{aligned}
\frac{d}{d\lambda}\left(I(\mathbf{u})_{i+1}\right) &= \mathcal{G}_{i+1}^T \frac{d\mathbf{u}_{i+1}}{d\lambda} = 0 \\
&= \mathcal{G}_{i+1}^T \mathbf{p}_i \\
&= -\left(b - A\mathbf{u}_{i+1}\right)^T \mathbf{p}_i \\
&= -\left(b - A\left(\mathbf{u}_i + \lambda_i \mathbf{p}_i\right)\right)^T \mathbf{p}_i \\
&= -\left(b - A\mathbf{u}_i\right)^T \mathbf{p}_i + \lambda_i \left(A\mathbf{p}_i\right)^T \mathbf{p}_i \\
\lambda_i &= -\frac{\mathcal{G}_i^T \mathbf{p}_i}{\mathbf{p}_i^T A \mathbf{p}_i}.
\end{aligned} \tag{3.6}$$

Thus it is possible to construct an iterative search algorithm that systematically uses elements of the basis vector $\mathbf{p}_i$ which spans the design space using the choice of λ_i prescribed by (3.6).

To see that the procedure will indeed converge to the minimum, it is convenient to introduce the error term

$$\mathbf{e}_i = \mathbf{u}_i - \mathbf{u}_s$$

where $\mathbf{u}_s$ is the location of the desired minimum. Now since $\mathbf{p}_i$ is defined to span the

[2]Note that $\bar{n}$ here is still the number of design variables

space $\mathbf{u}_i$ we can express the initial error $\mathbf{e}_0$ as

$$\mathbf{e}_0 = \sum_{i=0}^{\bar{n}-1} \delta_i \mathbf{p}_i, \tag{3.7}$$

with δ_i being the proper stepsize to eliminate the error component $\mathbf{e}_i$. Multiplying (3.7) by $\mathbf{p}_k^T A$, where k is between 0 and $\bar{n} - 1$, and using the conjugacy condition (3.5) gives

$$\begin{aligned}
\mathbf{p}_k^T A \mathbf{e}_0 &= \sum_{i=0}^{\bar{n}-1} \delta_i \mathbf{p}_k^T A \mathbf{p}_i \\
&= \delta_k \mathbf{p}_k^T A \mathbf{p}_k \\
\delta_k &= \frac{\mathbf{p}_k^T A \mathbf{e}_0}{\mathbf{p}_k^T A \mathbf{p}_k} \\
&= \frac{\mathbf{p}_k^T A \left(\mathbf{e}_0 + \sum_{j=0}^{k-1} \bar{\lambda}_j \mathbf{p}_j\right)}{\mathbf{p}_k^T A \mathbf{p}_k}.
\end{aligned} \tag{3.8}$$

Also,

$$\begin{aligned}
\mathbf{e}_0 &= \mathbf{u}_0 - \mathbf{u}_s \\
\mathbf{e}_1 &= \mathbf{u}_1 - \mathbf{u}_s \\
&= \mathbf{e}_0 + \mathbf{u}_1 - \mathbf{u}_0 \\
\mathbf{e}_{i+1} &= \mathbf{e}_i + \bar{\lambda}_i \mathbf{p}_i \\
\mathbf{e}_{i+1} &= \mathbf{e}_0 + \sum_{j=0}^{i} \bar{\lambda}_j \mathbf{p}_j.
\end{aligned} \tag{3.9}$$

Substituting (3.9) into (3.8) gives

$$\delta_k = \frac{\mathbf{p}_k^T A \mathbf{e}_k}{\mathbf{p}_k^T A \mathbf{p}_k},$$

and since

$$\begin{aligned}
\mathcal{G}_i &= A\mathbf{u}_i - b \\
&= A\mathbf{u}_i - b - (A\mathbf{u}_s - b) \\
&= A(\mathbf{u}_i - \mathbf{u}_s) \\
&= A\mathbf{e}_i
\end{aligned}$$

δ_k becomes

$$\delta_k = \frac{\mathbf{p}_k^T \mathcal{G}_k}{\mathbf{p}_k^T A \mathbf{p}_k}.$$

Thus $\delta_i = -\bar{\lambda}_i$, which means that the stepsizes $\bar{\lambda}$ defined by (3.6) are precisely the right choices such that in $\bar{n}$ steps $\mathbf{e}_i$ is eliminated. Mathematically,

$$\begin{aligned}
\mathbf{e}_i &= \mathbf{e}_0 + \sum_{j=0}^{i-1} \bar{\lambda}_j \mathbf{p}_j \\
&= \sum_{j=0}^{\bar{n}-1} \delta_j \mathbf{p}_j - \sum_{j=0}^{i-1} \delta_j \mathbf{p}_j \\
&= \sum_{j=i}^{\bar{n}-1} \delta_j \mathbf{p}_j \qquad (3.10)
\end{aligned}$$

giving $\mathbf{e}_{\bar{n}} = 0$. Thus if a set of conjugate vectors which span the design space are defined, it is possible to converge on the optimum in at most $\bar{n}$ steps.

One method of obtaining an A-orthogonal set of search directions is by the conjugate Gram-Schmidt process. Start with any non-A-orthogonal set of vectors which spans the design space, say $\mathbf{u}_i$. Form an A-orthogonal basis $\mathbf{p}_i$ by first setting $\mathbf{p}_0 = \mathbf{u}_0$. The remaining terms for $i > 0$ are formed by subtracting out components from the starting $\mathbf{u}_i$ that are not A-orthogonal to previous $\mathbf{p}_i$ components; that is,

$$\mathbf{p}_i = \mathbf{u}_i + \sum_{k=0}^{i-1} \beta_{i,k} \mathbf{p}_k \qquad (3.11)$$

where $\beta_{i,k}$ for $i > k$ are determined by multiplying by $\mathbf{p}_j^T A$, giving

$$\begin{aligned}
\mathbf{p}_j^T A \mathbf{p}_i &= \mathbf{p}_j^T A \mathbf{u}_i + \sum_{k=0}^{i-1} \beta_{i,k} \mathbf{p}_j^T A \mathbf{p}_k \\
0 &= \mathbf{p}_j^T A \mathbf{u}_i + \beta_{i,j} \mathbf{p}_j^T A \mathbf{p}_j \quad \text{for } i > j \\
\beta_{i,j} &= -\frac{\mathbf{p}_j^T A \mathbf{u}_i}{\mathbf{p}_j^T A \mathbf{p}_j}. \qquad (3.12)
\end{aligned}$$

Since the cost of forming $\mathbf{p}_i$ directly from equations (3.11) and (3.12) is proportional to $O(\bar{n}^3)$, a more efficient method becomes necessary. Returning to (3.10) we write,

$$\mathbf{e}_i = \sum_{j=i}^{\bar{n}-1} \delta_j \mathbf{p}_j$$

$$\begin{aligned}\mathbf{p}_k^T A\mathbf{e}_i &= \sum_{j=i}^{\tilde{n}-1} \delta_j \mathbf{p}_k^T A\mathbf{p}_j \\ \mathbf{p}_k^T \mathcal{G}_i &= 0 \quad \text{for } i > k.\end{aligned}$$

This means that, by definition, $\mathcal{G}_i$ is orthogonal to all previous conjugate search directions. Now by choosing our original basis as the set of gradients, $\mathbf{u}_i = -\mathcal{G}_i$, we get

$$\beta_{i,k} = +\frac{\mathbf{p}_k^T A\mathcal{G}_i}{\mathbf{p}_k^T A\mathbf{p}_k},$$

and

$$\begin{aligned}\mathcal{G}_{i+1} &= A\mathbf{e}_{i+1} \\ &= A\left(\mathbf{e}_i + \lambda_i \mathbf{p}_i\right) \\ &= \mathcal{G}_i + \lambda_i A\mathbf{p}_i \\ \mathcal{G}_j^T \mathcal{G}_{i+1} &= \mathcal{G}_j^T \mathcal{G}_i + \lambda_i \mathcal{G}_j^T A\mathbf{p}_i \\ -\mathcal{G}_j^T A\mathbf{p}_i &= -\frac{1}{\lambda_i}\mathcal{G}_j^T \mathcal{G}_{i+1} + \frac{1}{\lambda_i}\mathcal{G}_j^T \mathcal{G}_i \\ -\mathbf{p}_k^T A\mathcal{G}_i &= -\frac{1}{\lambda_k}\mathcal{G}_{k+1}^T \mathcal{G}_i + \frac{1}{\lambda_k}\mathcal{G}_k^T \mathcal{G}_i.\end{aligned}$$

Thus

$$\beta_{i,k} = \frac{1}{\lambda_k}\left(\frac{\mathcal{G}_{k+1}^T \mathcal{G}_i}{\mathbf{p}_k^T A\mathbf{p}_k} - \frac{\mathcal{G}_k^T \mathcal{G}_i}{\mathbf{p}_k^T A\mathbf{p}_k}\right).$$

and since $i > k$, β is defined as

$$\beta_{i,k} = \left\{\begin{array}{ll} \frac{1}{\lambda_{i-1}}\frac{\mathcal{G}_i^T \mathcal{G}_i}{\mathbf{p}_{i-1}^T A\mathbf{p}_{i-1}} & \text{for } k = i-1 \\ 0 & \text{for } k < i-1 \end{array}\right\}. \tag{3.13}$$

Thus, the k index may be dropped and (3.11) becomes

$$\mathbf{p}_i = \mathbf{u}_i + \beta_i \mathbf{p}_{i-1}.$$

β_i may be simplified further, using (3.6) to give

$$\beta_i = -\frac{\mathcal{G}_i^T \mathcal{G}_i}{\mathbf{p}_{i-1}^T \mathcal{G}_{i-1}}.$$

Finally, because

$$\begin{aligned}
\mathbf{p}_i &= \mathbf{u}_i + \sum_{k=0}^{i-1} \beta_{i,k} \mathbf{p}_k \\
\mathcal{G}_i^T \mathbf{p}_i &= \mathcal{G}_i^T \mathbf{u}_i + \sum_{k=0}^{i-1} \beta_{i,k} \mathcal{G}_i^T \mathbf{p}_k \\
&= \mathcal{G}_i^T \mathbf{u}_i \\
&= -\mathcal{G}_i^T \mathcal{G}_i .
\end{aligned}$$

Therefore,

$$\beta_i = \frac{\mathcal{G}_i^T \mathcal{G}_i}{\mathcal{G}_{i-1}^T \mathcal{G}_{i-1}}. \tag{3.14}$$

Instead of requiring $O(n^3)$ operations to generate a conjugate set of search directions, by using the gradients as the search directions it is possible to construct them as the algorithm proceeds at the cost of a few vector products. The entire conjugate gradient algorithm can be written as follows:

$$\begin{array}{ccc}
\text{set } \mathbf{p}_0 & = & -\mathcal{G}_0 \\
\text{then for} & i = 0, n-1 & \text{repeat:} \\
\lambda_i & = & \dfrac{\mathcal{G}_i^T \mathcal{G}_i}{\mathbf{p}_i^T A \mathbf{p}_i} \\
\mathbf{u}_{i+1} & = & \mathbf{u}_i + \lambda_i \mathbf{p}_i \\
\mathcal{G}_{i+1} & = & \mathcal{G}_i + \lambda_i A \mathbf{p}_i \\
\beta_{i+1} & = & \dfrac{\mathcal{G}_{i+1}^T \mathcal{G}_{i+1}}{\mathcal{G}_i^T \mathcal{G}_i} \\
\mathbf{p}_{i+1} & = & \mathcal{G}_{i+1} + \beta_{i+1} \mathbf{p}_i .
\end{array}$$

By using the conjugate gradient algorithm it is possible to minimize the quadratic function of equation (3.3), which is equivalent to solving (3.4). It was for this application that the algorithm was first developed. However, it can also be applied to more general functions. Used as a general optimization algorithm, where gradient and function information are available but the Hessian is not, the conjugate gradient algorithm can be written as:

$$
\begin{array}{rcll}
\text{set } \mathbf{p}_0 & = & -\mathcal{G}_0 & \\
\text{then for } & i = 0, \bar{n}-1 & \text{repeat:} & (3.15) \\
\text{find} & \bar{\lambda}_i & \text{that minimizes } I\left(\mathbf{u}_i + \lambda_i \mathbf{p}_i\right) & \\
\mathbf{u}_{i+1} & = & \mathbf{u}_i + \bar{\lambda}_i \mathbf{p}_i & \\
\text{calculate } \mathcal{G}_{i+1} & & & \\
\beta_{i+1} & = & \dfrac{\mathcal{G}_{i+1}^T \mathcal{G}_{i+1}}{\mathcal{G}_i^T \mathcal{G}_i} & \\
\mathbf{p}_{i+1} & = & \mathcal{G}_{i+1} + \beta_{i+1} \mathbf{p}_i . & (3.16)
\end{array}
$$

In this case, the cost of evaluating subsequent search directions, $\mathbf{p}$, has risen from $\bar{n}$ operations to $3\bar{n}$ operations. Even from the stand-point of treating a very large number of design variables, this remains an insignificant cost. However, this point must be stressed since the focus of this research is to allow essentially an unlimited number of design variables. Just as in the steepest descent algorithm, the calculation of $\bar{\lambda}_i$ becomes a one-dimensional search problem whose solution is discussed in Section (3.6). When the algorithm is used for functions that cannot be characterized by equation (3.3), the proof of convergence in $\bar{n}$ iterations no longer holds. As a result, care must be taken when employing (3.16), especially since it has a tendency to become stuck in regions of the design space where a well defined positive gradient still exists. This behavior can be understood by recalling that throughout the derivation of the algorithm two important assumptions were made. First, the problem was characterized by having a constant symmetric positive definite Hessian; and secondly, the current search direction was required to be conjugate to all preceding search directions. For quadratic problems, the second assumption results in never having to repeat a line search along an old direction. However, for a general problem it is often necessary to repeat search directions. Therefore, the tendency of (3.16) to eliminate previous search directions can actually penalize convergence. Furthermore, the determination of β, and hence the next search direction, hinged on the first assumption, and while the Hessian may be taken to be symmetric even in a general problem, it cannot be assumed to be constant or have positive eigenvalues. A rather crude fix, often used to keep the method from stalling, is to restart the algorithm after a given number of iterations $< \bar{n}$ or when convergence becomes excessively slow. In practice the conjugate gradient method employed on general problems displays greater efficiency than steepest descent but is far from ideal.[3]

3.4 NEWTON'S METHOD

By extending the Taylor series expansion to include the second order term,

$$I(\mathbf{u}_i + \mathbf{s}_i) = I(\mathbf{u}_i) + \mathcal{G}(\mathbf{u}_i)^T \mathbf{s}_i + \frac{1}{2}\mathbf{s}_i^T \mathcal{H}(\mathbf{u}_i)\mathbf{s}_i, \tag{3.17}$$

it is possible to use Newton's method:

$$\frac{dI}{d\mathbf{s}_i} = 0 = \mathcal{G}(\mathbf{u}_i) + \mathcal{H}(\mathbf{u}_i)\mathbf{s}_i.$$

Thus

$$\mathbf{s}_i = -\mathcal{H}_i^{-1}\mathcal{G}_i, \tag{3.18}$$

where $\mathbf{s}_i = \lambda_i \mathbf{p}_i$ from previous definitions. Therefore, if the Hessian and the gradient are available throughout the design space, $\mathbf{s}_i$ not only gives the direction but also the step length towards the minimum. For the quadratic problem of equation (3.3), we obtain

$$\begin{aligned} \mathcal{G}_i &= A\mathbf{u}_i - b \\ \mathcal{H}_i &= A \\ \mathbf{s}_i &= -A^{-1}(A\mathbf{u}_i - b) \\ &= \mathbf{u}_s - \mathbf{u}_i. \end{aligned}$$

[3]As an auxiliary note, the conjugate gradient method of solving symmetric positive definite matrices has been extended by many authors to treat more general non-symmetric systems, notably [99]. And while the method has been studied here in the context of the design problem, such methods have also been routinely applied to the flow analysis [7, 12].

In *one step* the minimum is found for the quadratic. For more general problems, the process can be repeated in order to converge to the minimum. The advantage of switching to Newton's method for these more general problems (provided that $\mathcal{G}_i$ and $\mathcal{H}_i$ are available and inexpensive) is that it should in theory accelerate convergence and hence reduce computational costs. If Newton's method is applied successfully, the convergence for general design problems would be quadratic as opposed to linear for the steepest descent method and superlinear for the conjugate gradient method. Further, since the solution of the quadratic problem (3.18) prescribes the step length as well as the search direction, a univariate line search with its additional function evaluations is not required. Unfortunately, and unlike the steepest descent method, the direction and step length prescribed by $\mathbf{s}_i$ for Newton's method do not guarantee a reduction in the objective function. A helpful improvement for highly nonlinear objective functions is to use a safeguarded Newton's method whereby $\mathbf{s}_i$ is used simply as a search direction $\mathbf{p}_i$ with a scalar multiplier $\bar{\lambda}_i$ being determined by a one-dimensional search (see Section 3.6). This modification which is often necessary for general problems has the the disadvantage of both dropping the possible convergence rate from quadratic to superlinear and adding the computational cost of function evaluations in univariate searches. Even with this adjustment, Newton's method proves impractical in many cases.

The most significant difficulty of using numerical optimization for aerodynamic shape design is that it can be very expensive. For the steepest descent and conjugate gradient algorithms explored earlier, both the objective function evaluation and the gradient of the design space are needed. As *Chapter 1* explained, if finite differences are used to calculate the gradient information, $O(n)$ flow field evaluations are required per design iteration [90]. Because of the prohibitive cost of these evaluations, this research seeks to find a means of reducing the cost of gradient information. For Newton's method, in addition to the objective function and the gradient, the Hessian is also required. If finite differences were used to calculate the Hessian, order $O(n^2/2)$ flow solutions would be required per design iteration. Of course control theory could be used to reduce the cost of the Hessian evaluation in a similar manner to the way it is used here to reduce the cost of the gradient evaluation. Unfortunately, even with the use of control theory the cost of using the full Newton's method is still prohibitive. While control theory can reduce the cost of the gradient from $O(\bar{n})$ to $O(2)$ flow field evaluations, the cost of the Hessian can only be reduced from $O(\bar{n}^2/2)$ to $O(\bar{n}+2)$, which implies that $O(n)$ flow field evaluations would still be needed per design iteration. Furthermore, it is questionable whether the use of full Newton information should be used, even if the computational cost of obtaining the Hessian were not proportional to the number of design variables. This results from the fact that calculating $\mathbf{p}$ requires $O(\bar{n}^3)$ operations per design iteration, as is evident in equation (3.18). Thus as $\bar{n}$ becomes truly large, the operational count to obtain the search direction will eventually surpass that of obtaining both the gradient and the Hessian. *Appendix A1* explains the methodology for using control theory to obtain cheap Hessian information. For this research all calculations will explore the reduction in the computational costs of the gradient only. Without an attempt to reduce the cost of the Hessian, Newton's method is impractical for aerodynamic optimization using CFD.

3.5 Quasi-Newton Method

The advantage of using Newton's method does not become entirely nullified by the impracticality of obtaining direct Hessian information. It is possible to obtain approximations to the Hessian matrix by using the available gradient information. Let the gradient be expanded about a point with a Taylor series, giving

$$\mathcal{G}(\mathbf{u}_i + \lambda_i \mathbf{p}_i) = \mathcal{G}(\mathbf{u}_i) + \lambda_i \mathcal{H}(\mathbf{u}_i)\mathbf{p}_i + \cdots . \tag{3.19}$$

Curvature information along the search direction $\mathbf{p}_i$ can thus be obtained.

The idea of the quasi-Newton method is to build up better approximations to the Hessian matrix systematically as the iterations proceed. It follows that an approximate Hessian can be defined as

$$\tilde{\mathcal{H}}_{i+1} = \tilde{\mathcal{H}}_i + \mathcal{U}_i$$

where $\mathcal{U}_i$ is an update matrix. The various forms of quasi-Newton methods in existence differ in the specification of this update matrix. The first condition that all of these techniques share in common is the quasi-Newton update condition, which can be stated as

$$\mathcal{G}(\mathbf{u}_{i+1}) = \mathcal{G}(\mathbf{u}_i) + \lambda_i \tilde{\mathcal{H}}_i \mathbf{p}_i,$$

or

$$\tilde{\mathcal{H}}_{i+1}\mathbf{s}_i = \mathbf{y}_i$$

where $\mathbf{s}_i = \lambda_i \mathbf{p}_i$ and $\mathbf{y}_i = \mathcal{G}(\mathbf{u}_{i+1}) - \mathcal{G}(\mathbf{u}_i)$ are known. It is noted that the quasi-Newton condition is a necessary approximation to the Taylor series expansion (3.19) arising from the need to lag information. The condition is simply a first order finite difference approximation of the change in the gradient along the search direction. Satisfying this condition alone does not create a unique update matrix. For example, if we choose an update of the form

$$\mathcal{U}_i = \mathbf{r}_i \mathbf{z}_i^T$$

where $\mathbf{r}_i$ and $\mathbf{z}_i$ are some vectors, the quasi-Newton direction becomes

$$\begin{aligned}
\tilde{\mathcal{H}}_{i+1}\mathbf{s}_i &= \left(\tilde{\mathcal{H}}_i + \mathbf{r}_i\mathbf{z}_i^T\right)\mathbf{s}_i = \mathbf{y}_i \\
\mathbf{r}_i\left(\mathbf{z}_i^T\mathbf{s}_i\right) &= \mathbf{y}_i - \tilde{\mathcal{H}}_i\mathbf{s}_i \\
\mathbf{r}_i &= \frac{1}{\mathbf{z}_i^T\mathbf{s}_i}\left(\mathbf{y}_i - \tilde{\mathcal{H}}_i\mathbf{s}_i\right) \\
\mathcal{U}_i &= \frac{1}{\mathbf{z}_i^T\mathbf{s}_i}\left(\mathbf{y}_i - \tilde{\mathcal{H}}_i\mathbf{s}_i\right)\mathbf{z}_i^T \qquad (3.20)
\end{aligned}$$

with $\mathbf{z}_i$ still remaining arbitrary. A point that will become useful later is the fact that, without violating the quasi-Newton condition, the expression above may be augmented with an additional term $\tilde{\mathbf{r}}_i\tilde{\mathbf{z}}_i^T$, where $\tilde{\mathbf{r}}_i$ is arbitrary and $\tilde{\mathbf{z}}_i$ is orthogonal to $\mathbf{s}_i$. Thus, $\mathcal{U}_i$ can be set to,

$$\mathcal{U}_i = \mathbf{r}_i\mathbf{z}_i^T + \tilde{\mathbf{r}}_i\tilde{\mathbf{z}}_i^T,$$

where it is obvious that the last term is annihilated by orthogonality to satisfy the quasi-Newton condition.

Fortunately, there is another condition that should be met by the approximate Hessian, namely that it, like the true Hessian, should be symmetric. Returning to equation (3.20), the easiest way to impose symmetry is by choosing $\mathbf{r}_i$ to be a multiple of $\mathbf{z}_i$ giving the unique rank one update:

$$\mathcal{U}_i = \frac{1}{\left(\mathbf{y}_i - \tilde{\mathcal{H}}_i \mathbf{s}_i\right)^T \mathbf{s}_i} \left(\mathbf{y}_i - \tilde{\mathcal{H}}_i \mathbf{s}_i\right)\left(\mathbf{y}_i - \tilde{\mathcal{H}}_i \mathbf{s}_i\right)^T .$$

However, this is not the only way to construct a symmetric update to the approximate Hessian. If we choose the update rule to follow

$$\begin{aligned} \tilde{\mathcal{H}}_{i+1} &= \frac{1}{2}\left(\tilde{\mathcal{H}}_i + \mathbf{r}_i \mathbf{z}_i^T\right) + \frac{1}{2}\left(\tilde{\mathcal{H}}_i + \mathbf{r}_i \mathbf{z}_i^T\right)^T \\ &= \tilde{\mathcal{H}}_i + \frac{1}{2}\left(\mathbf{r}_i \mathbf{z}_i^T\right) + \frac{1}{2}\left(\mathbf{r}_i \mathbf{z}_i^T\right)^T , \end{aligned}$$

symmetry of $\tilde{\mathcal{H}}_{i+1}$ is ensured. This rule results in the rank two update formula:

$$\mathcal{U}_i = \frac{1}{\mathbf{z}_i^T \mathbf{s}_i}\left[\left(\mathbf{y}_i - \tilde{\mathcal{H}}_i \mathbf{s}_i\right)\mathbf{z}_i^T + \mathbf{z}_i\left(\mathbf{y}_i - \tilde{\mathcal{H}}_i \mathbf{s}_i\right)^T\right] - \frac{\left(\mathbf{y}_i - \tilde{\mathcal{H}}_i \mathbf{s}_i\right)^T \mathbf{s}_i}{(\mathbf{z}_i^T \mathbf{s}_i)^2} \mathbf{z}_i \mathbf{z}_i^T .$$

By choosing various $\mathbf{z}_i$ not orthogonal to $\mathbf{s}_i$ it is possible to calculate a wide variety of possible updates that satisfy both symmetry and the quasi-Newton condition. Important examples are: the Powell-Symmetric-Broyden (PSB) update in which $\mathbf{z}_i = \mathbf{s}_i$, and the Davidon-Fletcher-Powell (DFP) update with $\mathbf{z}_i = \mathbf{y}_i$, which results in

$$\mathcal{U}_i = \frac{1}{\mathbf{y}_i^T \mathbf{s}_i}\left(\mathbf{y}_i \mathbf{y}_i^T\right) - \frac{1}{\mathbf{y}_i^T \mathbf{s}_i}\left(\tilde{\mathcal{H}}_i \mathbf{s}_i \mathbf{y}_i^T\right) - \frac{1}{\mathbf{y}_i^T \mathbf{s}_i}\left(\mathbf{y}_i \mathbf{s}_i^T \tilde{\mathcal{H}}_i\right) + \frac{\mathbf{s}_i^T \tilde{\mathcal{H}}_i \mathbf{s}_i}{(\mathbf{y}_i^T \mathbf{s}_i)^2}\left(\mathbf{y}_i \mathbf{y}_i^T\right) .$$

This may be rewritten as

$$\mathcal{U}_i = \frac{1}{\mathbf{y}_i^T \mathbf{s}_i}\left(\mathbf{y}_i \mathbf{y}_i^T\right) + \mathbf{s}_i^T \tilde{\mathcal{H}}_i \mathbf{s}_i \left(\tilde{\mathbf{z}}_i \tilde{\mathbf{z}}_i^T\right) - \frac{1}{\mathbf{s}_i^T \tilde{\mathcal{H}}_i \mathbf{s}_i}\left(\tilde{\mathcal{H}}_i \mathbf{s}_i \mathbf{s}_i^T \tilde{\mathcal{H}}_i\right) ,$$

where

$$\tilde{\mathbf{z}}_i = \frac{1}{\mathbf{y}_i^T \mathbf{s}_i}\mathbf{y}_i - \frac{1}{\mathbf{s}_i^T \tilde{\mathcal{H}}_i \mathbf{s}_i}\left(\tilde{\mathcal{H}}_i \mathbf{s}_i\right)$$

and it can be verified that $\tilde{\mathbf{z}}_i$ is orthogonal to $\mathbf{s}_i$. Thus, a family of related rank two methods which all satisfy the quasi-Newton condition can be defined by multiplying the second term by a constant. The DFP method happens to be the one in which this constant is chosen as unity. It is generally accepted that the best choice for this constant is actually zero, with the resulting update known as the Broyden-Fletcher-Goldfarb-Shanno (BFGS) update and given by

$$\mathcal{U}_i = \frac{-1}{\mathbf{s}_i^T \tilde{\mathcal{H}}_i \mathbf{s}_i}\tilde{\mathcal{H}}_i \mathbf{s}_i \mathbf{s}_i^T \tilde{\mathcal{H}}_i + \frac{1}{\mathbf{y}_i^T \mathbf{s}_i}\mathbf{y}_i \mathbf{y}_i^T .$$

For this research, the BFGS quasi-Newton algorithm will be used exclusively. While many other choices are available, the BFGS quasi-Newton method remains the standard by which other gradient-based methods are measured. There is no intent here to select the best numerical optimization procedure since the research focuses upon obtaining inexpensive gradient information.

An important implementational issue in using quasi-Newton procedures is where the updates should be applied. As presented thus far, the Hessian itself can be directly updated. However, this is not the only choice; some schemes update the inverse of the approximate Hessian while others update the Cholesky factors of the Hessian. Updating the Cholesky factors is beneficial for two reasons. First, calculating the search direction according to (3.18) is considerably cheaper if the Cholesky factors are used as opposed to the Hessian itself. Calculating the updates to the Cholesky factors costs no more than updating the Hessian itself. Secondly, it is easy to correct the updates to the Cholesky factors such that they result in positive definiteness of $\tilde{\mathcal{H}}$. Note that the BFGS update may be written as

$$\mathcal{U}_i = \bar{\gamma}_1 \bar{\mathbf{z}}_i \bar{\mathbf{z}}_i^T + \bar{\gamma}_2 \mathbf{y}_i \mathbf{y}_i^T$$

where $\bar{\gamma}_1$ and γ_2 are scalars and $\bar{\mathbf{z}}_i = \tilde{\mathcal{H}}_i \mathbf{s}_i$. The point is that $\mathcal{U}_i$ is the sum of two rank one updates.

Developing this further, assume that instead of storing $\tilde{\mathcal{H}}_i$ its Cholesky factors are stored; that is

$$\tilde{\mathcal{H}}_i = \tilde{\mathcal{L}}_i \tilde{\mathcal{D}}_i \tilde{\mathcal{L}}_i^T,$$

where $\tilde{\mathcal{L}}_i$ is a lower unit triangular matrix and $\tilde{\mathcal{D}}_i$ is a diagonal matrix. So long as $\tilde{\mathcal{H}}_i$ is symmetric and positive definite, this factorization is possible and yields positive elements in $\tilde{\mathcal{D}}_i$. As mentioned earlier, by storing $\tilde{\mathcal{L}}_i$ and $\tilde{\mathcal{D}}_i$ the cost of calculating the quasi-Newton search direction by solving

$$\begin{aligned} \tilde{\mathcal{H}}_i \mathbf{s}_i &= -\mathcal{G}_i \\ \tilde{\mathcal{L}}_i \tilde{\mathcal{D}}_i \tilde{\mathcal{L}}_i^T \mathbf{s}_i &= -\mathcal{G}_i \end{aligned}$$

for $\mathbf{s}_i$ is reduced since only back substitution with $O(\bar{n}^2)$ operations is required. Now, since the previous $\tilde{\mathcal{L}}$ and $\tilde{\mathcal{D}}$ exist at the new iteration, it remains to obtain the low rank updates of these factors:

$$\begin{aligned}
\tilde{\mathcal{H}}_{i+1} &= \tilde{\mathcal{H}}_i + \mathcal{U}_i \\
&= \tilde{\mathcal{H}}_i + \bar{\gamma}_1 \bar{\mathbf{z}}_i \bar{\mathbf{z}}_i^T + \bar{\gamma}_2 \mathbf{y}_i \mathbf{y}_i^T \\
\tilde{\mathcal{L}}_{i+1} \tilde{\mathcal{D}}_{i+1} \tilde{\mathcal{L}}_{i+1}^T &= \tilde{\mathcal{L}}_i \tilde{\mathcal{D}}_i \tilde{\mathcal{L}}_i^T + \bar{\gamma}_1 \bar{\mathbf{z}}_i \bar{\mathbf{z}}_i^T + \bar{\gamma}_2 \mathbf{y}_i \mathbf{y}_i^T \\
&= \tilde{\mathcal{L}}_i \left(\tilde{\mathcal{D}}_i + \bar{\gamma}_1 \hat{\mathbf{z}}_i \hat{\mathbf{z}}_i^T + \bar{\gamma}_2 \hat{\mathbf{y}}_i \hat{\mathbf{y}}_i^T \right) \tilde{\mathcal{L}}_i^T
\end{aligned}$$

where $\hat{\mathbf{z}}_i$ and $\hat{\mathbf{y}}_i$ are the solutions of the triangular systems,

$$\begin{aligned}
\tilde{\mathcal{L}}_i \hat{\mathbf{z}}_i &= \bar{\mathbf{z}}_i \\
\tilde{\mathcal{L}}_i \hat{\mathbf{y}}_i &= \mathbf{y}_i .
\end{aligned}$$

Next it follows by construction that

$$\tilde{\mathcal{D}}_i + \bar{\gamma}_1 \hat{\mathbf{z}}_i \hat{\mathbf{z}}_i^T + \bar{\gamma}_2 \hat{\mathbf{y}}_i \hat{\mathbf{y}}_i^T = \hat{\mathcal{L}}_i \hat{\mathcal{D}}_i \hat{\mathcal{L}}_i^T$$

without any work due to the special form of the rank one updates. Further, it turns out that $\hat{\mathcal{L}}_i$ is very sparse and has a fixed structure. The result is that

$$\tilde{\mathcal{L}}_{i+1} \tilde{\mathcal{D}}_{i+1} \tilde{\mathcal{L}}_{i+1}^T = \tilde{\mathcal{L}}_i \hat{\mathcal{L}}_i \hat{\mathcal{D}}_i \hat{\mathcal{L}}_i^T \tilde{\mathcal{L}}_i^T$$

where $\tilde{\mathcal{L}}_{i+1} = \tilde{\mathcal{L}}_i \hat{\mathcal{L}}_i$ and $\tilde{\mathcal{D}}_{i+1} = \hat{\mathcal{D}}_i$. As a last point, since retaining all positive elements in $\tilde{\mathcal{D}}$ ensures that the approximate Hessian is positive definite, and hence the solution of the quasi-Newton search direction remains non-singular, a way to safeguard the design procedure can be incorporated by forcing all the terms of the new $\tilde{\mathcal{D}}_{i+1}$ to remain positive.

The entire quasi-Newton procedure with BFGS updates of the Cholesky factors can now be written as

$$\begin{aligned}
\text{set } \tilde{\mathcal{L}}_1 &= 0 \text{ and } \tilde{\mathcal{D}}_1 = \mathcal{I} \\
\text{calculate } & \quad \mathcal{G}_1
\end{aligned}$$

$$
\begin{array}{rcl}
\mathbf{p}_1 & = & -\mathcal{G}_1 \\
\text{then for} & i > 0 & \text{repeat:} \\
\text{find} & \bar{\lambda}_i & \text{that minimizes } I\,(\mathbf{u}_i + \lambda_i \mathbf{p}_i) \\
\mathbf{u}_{i+1} & = & \mathbf{u}_i + \bar{\lambda}_i \mathbf{p}_i \\
\text{calculate} & & \mathcal{G}_{i+1} \\
\mathbf{s}_i & = & \bar{\lambda}_i \mathbf{p}_i \\
\mathbf{y}_i & = & \mathcal{G}_{i+1} - \mathcal{G}_i \\
\bar{\mathbf{z}}_i & = & \tilde{\mathcal{L}}_i \tilde{\mathcal{D}}_i \tilde{\mathcal{L}}_i^T \mathbf{s}_i \\
\bar{\gamma}_1 & = & -\dfrac{1}{\mathbf{s}_i^T \bar{\mathbf{z}}_i} \text{ and } \quad \bar{\gamma}_2 = \dfrac{1}{\mathbf{y}_i^T \mathbf{s}_i} \\
\text{calculate} & & \hat{\mathbf{z}}_i \text{ and } \hat{\mathbf{y}}_i \text{ from} \\
\tilde{\mathcal{L}}_i \hat{\mathbf{z}}_i & = & \bar{\mathbf{z}}_i \\
\tilde{\mathcal{L}}_i \hat{\mathbf{y}}_i & = & \mathbf{y}_i \\
\text{assign} & & \hat{\mathcal{L}}_i \text{ and } \hat{\mathcal{D}}_i \text{ from} \\
\tilde{\mathcal{D}}_i + \bar{\gamma}_1 \hat{\mathbf{z}}_i \hat{\mathbf{z}}_i^T + \bar{\gamma}_2 \hat{\mathbf{y}}_i \hat{\mathbf{y}}_i^T & = & \hat{\mathcal{L}}_i \hat{\mathcal{D}}_i \hat{\mathcal{L}}_i^T \\
\tilde{\mathcal{L}}_{i+1} & = & \tilde{\mathcal{L}}_i \hat{\mathcal{L}}_i \text{ and } \quad \tilde{\mathcal{D}}_{i+1} = \hat{\mathcal{D}}_i \\
\text{solve for} & & \mathbf{p}_{i+1} \text{ by back substitution of} \\
\tilde{\mathcal{L}}_{i+1} \tilde{\mathcal{D}}_{i+1} \tilde{\mathcal{L}}_{i+1}^T \mathbf{p}_{i+1} & = & -\mathcal{G}_{i+1}.
\end{array}
\tag{3.21}
$$

Note that even though the last expression in (3.21) should actually solve for $\mathbf{s}_{i+1}$ instead of $\mathbf{p}_{i+1}$ the latter is used so that the univariate line search can be used to safeguard the iteration process. However, even with its improvements in efficiency, this form of the quasi-Newton method must be used cautiously within the framework of optimization in the presence of inexpensive gradients. By applying updates to the Cholesky factors, the operational count of calculating the search direction drops from $O(\bar{n}^3)$ to $O(\bar{n}^2)$. But even this operational count can become unmanageably expensive from the standpoint of storage and extra CPU time for very large $\bar{n}$.

It is reasonable to conjecture that for a three-dimensional problem with a mesh of $\hat{n}$ points an estimate of the number of surface points is given by $O(\hat{n}^{\frac{2}{3}})$. Further, the design must be limited to having at most a similar number of design variables. Thus the cost of calculating the search direction from the Cholesky factors for the maximum number of design variables can be estimated as $O(\hat{n}^{\frac{4}{3}})$ operations. Similarly, the storage of the factored Hessian, for this worst case, can be estimated as $O(\hat{n}^{\frac{4}{3}})$. The result is that, both in terms of time and storage, simply calculating the search direction can become more expensive than the gradient evaluation itself—a relatively unhappy prospect. The point at which this trade-off occurs remains as an area of future study.

A more complete treatment of the quasi-Newton and other optimization strategies is given by Gill, Murray and Wright [31]. In this work the BFGS quasi-Newton method applied to the Cholesky factored Hessian matrix will be used exclusively. The specific implementation of the algorithm (3.21) that is used, QNMDIF, was written by Gill, Murray and Pitfield [30] and enhanced by Kennelly [67].

3.6 Line Search Algorithm

Once the search direction is obtained from the quasi-Newton or alternative method, step (2) of the general algorithm presented in Section (3.1) is complete. For highly non-linear design spaces, an explicit method to determine the optimal step length does not exist. Thus an iterative univariate minimization method may be employed to find the optimum stepsize. Any of the available line search techniques, such as the Fibonacci search, the golden section search, or successive quadratic interpolation, can be used in order to estimate the minimum along the search direction. *Here, successive quadratic interpolation is used exclusively.* Successive quadratic interpolation calculates the objective function at three points in the search direction, fits a parabola through the three points, and then obtains the minimum of that parabola. The objective function is then calculated at the location of the projected minimum, replacing one of the original three points. A new quadratic is thus defined and the process is repeated until a solution within a specified tolerance of the true minimum along $\mathbf{p}_i$ is obtained. A consideration that should not be neglected is the computational cost of these univariate searches. If an iterative line search method is used in conjunction with an adjoint based method to obtain gradients, the computational cost of the function evaluations within these searches is likely to constitute the majority of the total computational costs. It is for this among other reasons that Jameson chose not to attempt iterative univariate searches in his initial work. However, it must be recalled that for both conjugate gradient and quasi-Newton methods which employ rank two updates, there is an assumption that minimums are attained along their previous search directions.

Once the univariate minimum along the search direction is found, step (3) of the general algorithm in Section (3.1) is complete and the entire process can be repeated. In practice the unconstrained quasi-Newton algorithm combined with successive quadratic interpolations for the line searches is extremely efficient. It has been applied successfully by the author to many aerodynamic optimization problems where finite-difference gradients were used [90, 97, 91]. Reference [91] shows that even for three-dimensional calculations, where the large cost of the function evaluations often debilitates practical design, the quasi-Newton/successive quadratic interpolation approach can in fact obtain designs with reasonable computational resources. It was through this finite difference research that it became apparent that most of the computational effort spent in the optimization process occurred in determining gradients. The idea was therefore to replace the finite difference gradient by a much cheaper and yet still accurate technique, while leaving the rest of the design process intact.

3.7 Alternative Strategies

Although this research will exclusively use a quasi-Newton descent method, such a method may not necessarily be the best alternative when an efficient adjoint solver is employed. In fact the development of a method to obtain cheap gradients is forcing a reexamination of what design algorithm is appropriate.

3.7.1 Finite difference based gradients

When finite difference methods are used to approximate the gradients, it is always necessary to work with highly converged solutions. This is illustrated by taking a simple example involving drag minimization.

Let us say that the objective function is $I = \varphi C_d$, where φ is such that the product scales to $O(1)$. Now, suppose that during the calculation of the flow, the error in the evaluation of the cost function scales linearly with the residual error of the flow solution.[4] The finite difference procedure may be written as

$$\begin{aligned}\frac{dI}{d\bar{h}} &= \frac{\varphi\left(C_d + \Delta C_d \pm \tilde{\lambda}\rho_1\right) - \varphi\left(C_d \pm \tilde{\lambda}\rho_1\right)}{\Delta \bar{h}} \\ &= \varphi\left(\frac{\Delta C_d}{\Delta \bar{h}} \pm \frac{2\tilde{\lambda}\rho_1}{\Delta \bar{h}}\right) \\ &= \varphi\frac{\Delta C_d}{\Delta \bar{h}}\left(1 \pm \frac{2\tilde{\lambda}\rho_1}{\Delta C_d}\right)\end{aligned}$$

where ρ_1 is the residual of the discrete flow equations, $\tilde{\lambda}$ is the ratio of the error in the cost function to the residual of the flow solution, and $\Delta\bar{h}$ is the change in the design variables. If it is required to have at least 2 significant digits of accuracy in the gradient, then

$$\frac{2\tilde{\lambda}\rho_1}{\Delta C_d} < .01.$$

This implies that for a given $\tilde{\lambda}$, say $\tilde{\lambda} = 1$, the flow solver must be converged to two orders below the level of ΔC_d. The order of ΔC_d is determined by the magnitude of $\Delta\bar{h}$. However, since the magnitude of $\Delta\bar{h}$ must be small in order for the discretization

[4]This relationship seems to hold for many aerodynamic design problems involving nonlinear inviscid governing equations.

error of the finite difference method to be small, $|\rho_1|$ must be forced to small values in order for the method to remain accurate. Assuming $\Delta C_d = .0001$ (one drag count), this gives $|\rho_1| = .00001$ meaning a convergence tolerance of 10^{-6} for the flow solver.

Thus even for cases where accuracy in the gradient can be relaxed, a high order of convergence is required for flow solutions. It was this fact that led initial researchers in the field, such as Hicks [39], to lean towards more sophisticated descent algorithms. The reasoning was that since significant computational effort was expended to achieve reasonable gradients for even a few design variables, the maximum possible information should be obtained from these expensive gradients.

3.7.2 ADJOINT-BASED GRADIENTS

Replacing the finite difference gradient by one obtained through the solution of the adjoint equations changes the picture dramatically. Not only is it possible to determine the gradient with respect to many design variables at low cost, but also the error analysis of the gradient changes for the better:

$$\begin{aligned}\frac{dI}{dh} &= \varphi\left(\frac{dC_d}{dh} \pm \frac{dC_d}{dh}\left(\tilde{\lambda}_1\rho_1 + \tilde{\lambda}_2\rho_2\right)\right)\\ &= \varphi\frac{dC_d}{dh}\left(1 \pm \left(\tilde{\lambda}_1\rho_1) + \tilde{\lambda}_2\rho_2\right)\right)\end{aligned}$$

where ρ_2 is the order of convergence of the adjoint solver, and λ_1 and λ_2 represent the ratios of the error in $\frac{dC_d}{dh}$ to the residuals of the flow and adjoint solutions respectively. The error term is scaled by $\frac{dC_d}{dh}$ since the solution of the adjoint is the solution of only the Δ portion of the flow solution. Thus, as the gradient goes to zero, the error must also go to zero so long as the adjoint solver consistently represents the variation of the flow equations. Note that since $\frac{dC_d}{dh}$ is directly calculated the accuracy of the gradient does not depend on a choice of Δh. Again, if $\tilde{\lambda}_1 = 1$ and $\tilde{\lambda}_2 = 1$ and two places are required for the gradient, $|\rho_1| + |\rho_2| < .01$. This implies that the flow solver and the adjoint solver must be converged to just less than 10^{-2} instead of 10^{-6}. The key point is that even after only a few flow solver and adjoint solver iterations are complete, valuable gradient information is already available. In contrast, hundreds of flow solver iterations are usually required to obtain useful levels of accuracy in the gradient information for the finite difference method. The concluding implication is that if a sophisticated descent procedure is used which requires that the minimum be obtained along the calculated search directions, accurate flow solutions obtained for these single parameter searches may be far more expensive than the cheap solutions required for the gradient calculations.

This realization leads to reassessment of the whole design procedure. As cited earlier, Ta'asan chose the alternative of tightly coupling the flow solver, the adjoint solver, and the design solution via a multigrid procedure. This approach cannot be used in conjunction with a sophisticated descent procedure because such procedures are intolerant of the noise created in both the cost function and gradient by the initial stages of the strongly coupled approach. In theory, this alternate approach would converge all three systems with a small coupling penalty for each of the systems; thus the total cost of the entire design would be reduced to approximately that of two flow solutions. It was this attractive feature which initially led Jameson to use tight coupling and the steepest descent algorithm in his preliminary works [54, 55, 59, 58]. Ta'asan's approach is an extension of Jameson's initial method in which multigridding is used to further accentuate the coupling among the three systems. Although it is not explored here, future efforts that employ an adjoint formulation should study the trade-offs between the level of coupling and the use of sophisticated design methods.

3.7.3 Limited Memory Quasi-Newton Methods

The future of using adjoint methods may reside in their application in conjunction with limited memory quasi-Newton methods (LMQN). This wide class of methods tends to fit between the conjugate gradient and (full) quasi-Newton methods that have been described earlier in this chapter. They attempt to combine the low storage requirements of the conjugate gradient methods with the computational efficiency of the quasi-Newton methods. Some of these methods, like the conjugate gradient method, employ a restart procedure if it appears convergence toward a minimum has stalled. In the case of the conjugate gradient method, recall that these restarts were necessary for problems in which a particular search direction can not be properly eliminated from the design space by one pass. For aerodynamic shape optimization, where exact line searches are not available and quadratic design spaces are not necessarily present, this restart option becomes necessary. Some LMQN methods attempt to improve the available design space information of the basic conjugate gradient method by recasting it as a quasi-Newton method [9, 29, 101]. These methods retain the conjugate gradient methodology of eliminating prior search directions but attempt to take greater advantage of the available curvature information along each successive search direction. They are characterized by very low memory requirements making them ideal for large scale problems. However they also retain the necessity to restart that is inherent in conjugate gradient methods. Other LMQN methods[79] resemble true quasi-Newton methods more closely. These methods achieve their low memory by storing the necessary vectors that are used to calculate update matrices instead of the approximate Hessian. Then instead of retaining all of the previous update information, only those developed from the last several iterates are retained. The search direction for the next iterate can be calculated directly from the update vectors without directly determining the approximate Hessian. This strategy has the advantage over the conjugate gradient-like methods in that no restart option is necessary. Old search direction information is progressively dropped as the method proceeds. As for true quasi-Newton methods, the update vectors may be specified by a host of different low rank approaches. Again large scale problems are possible since the storage and computational costs of these methods are proportional to $\mathcal{K}n$ where $\mathcal{K}$ is the number of previous updates that are retained. Such alternatives should be attractive for use in conjunction with adjoint methods since they are more appropriate for large scale problems. More importantly, if rank one update versions of these methods are used they should be more robust in the presence of both inexact line searches (since the rank one update does not place any conditions upon obtaining a minimum along the search direction for accuracy to be achieved) and inexact gradients (since old possibly inaccurate updates are quickly cycled out of the Hessian approximation). Thus it may be possible to construct an approach, such as the one used by Jameson and Ta'asan, that attempts to converge the state, costate, and design systems simultaneously without resorting to steepest descent.

Chapter 4

Control Theory Applied to the Potential Flow Equation

Chapter 1 presented the conceptual framework for applying control theory to a general cost function and an arbitrary set of constraint governing equations. This chapter explores the application of control theory to various cost functions and flows governed by the potential flow equation. In keeping with Jameson's previous work [54, 55], this research explores the application of control theory to the continuous differential equations (often termed the continuous sensitivity approach). The development follows closely from that presented in reference [55] of Jameson's earlier work, with the significant difference here being that the potential equations are treated in a general transformed coordinates instead of polar coordinates and a conformal mapping.

4.1 Potential Flow Equations

In this chapter equations (1.6 - 1.13) are applied to the differential form of the potential flow equation. Consider the case of two-dimensional compressible inviscid flow. The steady state potential flow equation can be written as in equation (2.9):

$$\frac{\partial}{\partial x}(\rho u) + \frac{\partial}{\partial y}(\rho v) = 0, \tag{4.1}$$

where u and v represent the Cartesian velocity components. The coordinate transformations may be defined as in equation (2.10):

$$\left\{ \begin{array}{c} u \\ v \end{array} \right\} = \left\{ \begin{array}{c} \phi_x \\ \phi_y \end{array} \right\} = \left[\begin{array}{cc} \frac{\partial \xi}{\partial x} & \frac{\partial \eta}{\partial x} \\ \frac{\partial \xi}{\partial y} & \frac{\partial \eta}{\partial y} \end{array} \right] \left\{ \begin{array}{c} \phi_\xi \\ \phi_\eta \end{array} \right\}$$

$$= K^{T^{-1}} \left\{ \begin{array}{c} \phi_\xi \\ \phi_\eta \end{array} \right\}, \tag{4.2}$$

where x and y represent the physical plane, and ξ and η represent the computational plane. Following the development in Section (2.4.1) we can write

$$\frac{\partial}{\partial \xi}(\rho J U) + \frac{\partial}{\partial \eta}(\rho J V) = 0 \quad \text{in } D, \tag{4.3}$$

where

$$U = A_{11}\phi_\xi + A_{12}\phi_\eta \tag{4.4}$$

$$V = A_{12}\phi_\xi + A_{22}\phi_\eta, \tag{4.5}$$

and

$$\mathcal{A} = K^{-1}K^{T^{-1}} = \left[\begin{array}{cc} A_{11} & A_{12} \\ A_{12} & A_{22} \end{array} \right].$$

4.2 Cost Function for the Inverse Problem

In order to carry out the continuous sensitivity approach, it is first necessary to define the cost function. While various cost functions will be developed in this research, it is illustrative to develop fully one particular choice of the cost function and generalize later. The simplest practical design problem is to force the airfoil toward a target surface speed distribution. One concern that confronts this, as well as any choice in the cost function, is whether it can lead to a design problem having non-unique solutions. Since this will be a concern for the design problem using the Euler equations as well, the reader is referred to the general discussion to be presented in Section (5.2.1). Returning to the problem at hand, for isentropic potential flow, the specification of a surface speed distribution uniquely determines pressure, so the problem is equivalent to specifying a target pressure distribution. In effect, we are using control theory combined with numerical optimization to solve the inverse problem. In fact, the development follows very closely Lighthill's original development for incompressible potential flow. The major differences here are that the target need not necessarily be achievable for the method to produce useful results, and that the method is generalized to treat compressible flows. Consider the cost function defined such that a target speed distribution will be achieved:

$$I = \frac{1}{2}\oint_C (q - q_d)^2\, ds = \frac{1}{2}\oint_C (q - q_d)^2 \left(\frac{ds}{d\xi}\right) d\xi, \tag{4.6}$$

where q_d is the desired speed distribution, C is the airfoil surface, and s is the arc length along the surface.

The design problem is now treated as a control problem where the control function is the airfoil shape, which is to be chosen to minimize I subject to the constraints defined by the flow equations (4.1–4.5). Following the linear algebra developed in the introduction (1.6–1.13) we take the first variation of this cost function giving

$$\begin{aligned}\delta I &= \oint_C (q - q_d)\,\delta q \left(\frac{ds}{d\xi}\right) d\xi + \frac{1}{2}\oint_C (q - q_d)^2\, \delta\left(\frac{ds}{d\xi}\right) d\xi \\ &= \oint_C (q - q_d)\frac{\partial(\delta\phi)}{\partial\xi} d\xi \\ &\quad + \frac{1}{2}\oint_C (q - q_d)^2\, \delta\left(\frac{ds}{d\xi}\right) d\xi + \oint_C (q - q_d)\frac{\partial\phi}{\partial\xi}\,\delta\left(\frac{d\xi}{ds}\right)\frac{ds}{d\xi} d\xi,\end{aligned} \tag{4.7}$$

since on the airfoil surface

$$q_C = \frac{\partial\phi}{\partial s} = \frac{\partial\phi}{\partial\xi}\frac{\partial\xi}{\partial s}.$$

By using the fact that

$$\delta\left(\frac{d\xi}{ds}\right) = \delta\left(\frac{1}{\frac{ds}{d\xi}}\right) = -\frac{1}{\left(\frac{ds}{d\xi}\right)^2}\,\delta\left(\frac{ds}{d\xi}\right)$$

we may rewrite equation (4.7) as

$$\delta I = \oint_C (q - q_d)\frac{\partial(\delta\phi)}{\partial\xi} d\xi + \frac{1}{2}\oint_C \left(q_d^2 - q^2\right)\delta\left(\frac{ds}{d\xi}\right) d\xi. \tag{4.8}$$

4.3 VARIATIONS OF THE GOVERNING EQUATION

In general, the next step is to find how a modification to the airfoil geometry causes a variation $\delta\phi$, as well as a variation in the grid parameters $\delta A_{11}, \delta A_{12}, \delta A_{22}$, and δJ. The variations in U, V and ρ are

$$\begin{aligned}
\delta U &= \delta(A_{11})\phi_\xi + A_{11}\delta\phi_\xi + \delta(A_{12})\phi_\eta + A_{12}\delta\phi_\eta \\
\delta V &= \delta(A_{12})\phi_\xi + A_{12}\delta\phi_\xi + \delta(A_{22})\phi_\eta + A_{22}\delta\phi_\eta \\
\delta\rho &= -\frac{\rho}{c^2}\left[U\frac{\partial}{\partial\xi} + V\frac{\partial}{\partial\eta}\right]\delta\phi - \frac{\rho}{2c^2}\left[\delta A_{11}\phi_\xi^2 + 2\delta A_{12}\phi_\eta\phi_\xi + \delta A_{22}\phi_\eta^2\right].
\end{aligned}$$

By inserting these relations into equation (4.3) it follows that $\delta\phi$ satisfies

$$L\delta\phi = -\frac{\partial}{\partial\xi}\tilde{Q}(\delta J, \delta A_{11}, \delta A_{12}, \delta A_{22}) - \frac{\partial}{\partial\eta}\tilde{P}(\delta J, \delta A_{11}, \delta A_{12}, \delta A_{22}), \tag{4.9}$$

where

$$\begin{aligned}
L \equiv \quad & \frac{\partial}{\partial\xi}\left[\rho J\left(A_{11} - \frac{U^2}{c^2}\right)\frac{\partial}{\partial\xi} + \rho J\left(A_{12} - \frac{UV}{c^2}\right)\frac{\partial}{\partial\eta}\right] \\
+ \quad & \frac{\partial}{\partial\eta}\left[\rho J\left(A_{12} - \frac{UV}{c^2}\right)\frac{\partial}{\partial\xi} + \rho J\left(A_{22} - \frac{V^2}{c^2}\right)\frac{\partial}{\partial\eta}\right]
\end{aligned} \tag{4.10}$$

and

$$\begin{aligned}
\tilde{Q} &= \rho U\delta J + \rho J\phi_\xi\left(1 - \frac{U\phi_\xi}{2c^2}\right)\delta A_{11} + \rho J\phi_\eta\left(1 - \frac{U\phi_\xi}{c^2}\right)\delta A_{12} + \rho J\phi_\eta\left(-\frac{U\phi_\eta}{2c^2}\right)\delta A_{22} \\
\tilde{P} &= \rho V\delta J + \rho J\phi_\eta\left(1 - \frac{V\phi_\eta}{2c^2}\right)\delta A_{22} + \rho J\phi_\xi\left(1 - \frac{V\phi_\eta}{c^2}\right)\delta A_{12} + \rho J\phi_\xi\left(-\frac{V\phi_\xi}{2c^2}\right)\delta A_{11}.
\end{aligned}$$

The result is that (4.9) is equivalent to step (1.7) in the introduction. Now following the developments between equations (1.7) and (1.12), if ψ is any periodic function vanishing in the far field, equation (4.9) can be multiplied by ψ and integrated over the domain giving

$$\int_D \psi L\delta\phi \, d\xi\, d\eta + \int_D \left(\psi\frac{\partial\tilde{Q}}{\partial\xi} + \psi\frac{\partial\tilde{P}}{\partial\eta}\right) d\xi\, d\eta = 0. \tag{4.11}$$

Here the integration over the domain, just as in the case of the integration on the surface to obtain I, is the continuous equivalent to the matrix-vector and vector-vector products seen in the introduction. In the case of the definition of I (4.6), the integration results in a single well-defined scalar cost function. Likewise, in the case of equation (4.11) the integration over the domain results in a scalar which can then be subtracted from the definition of δI.

4.4 DECOMPOSITION OF THE VARIATIONAL POTENTIAL

Now as in the potential flow solution methodology (see Section 2.4.1), the variational of the potential is split here into three parts:

$$\delta\phi = \delta\phi_G + \delta\phi_E + \delta\phi_{U_\infty}$$

where $\delta\phi_G$ is the variation of the perturbation potential, $\delta\phi_E$ is the variation of the circulation potential, and $\delta\phi_{U_\infty}$ is the variation of the free stream potential. Thus

$$\int_D \left(\psi L\delta\phi_G + \psi L\delta\phi_E + \psi L\delta\phi_{U_\infty}\right) d\xi\, d\eta = -\int_D \left(\psi\frac{\partial\tilde{Q}}{\partial\xi} + \psi\frac{\partial\tilde{P}}{\partial\eta}\right) d\xi\, d\eta. \tag{4.12}$$

Similarly, the definition of the variation in the cost function, equation (4.8), can also be split, giving after integration by parts,

$$\begin{aligned}\delta I \;=\; & -\oint_C \frac{\partial(q-q_d)}{\partial\xi}\delta\phi_G d\xi - \oint_C \frac{\partial(q-q_d)}{\partial\xi}\delta\phi_E d\xi - \oint_C \frac{\partial(q-q_d)}{\partial\xi}\delta\phi_{U_\infty} d\xi \\ & +\frac{1}{2}\oint_C \left(q_d^2 - q^2\right)\delta\left(\frac{ds}{d\xi}\right) d\xi.\end{aligned} \tag{4.13}$$

By subtracting (4.12) from (4.13) we mimic step (1.8) to obtain:

$$\begin{aligned}\delta I \;=\; & -\oint_C \frac{\partial(q-q_d)}{\partial\xi}\delta\phi_G d\xi - \oint_C \frac{\partial(q-q_d)}{\partial\xi}\delta\phi_E d\xi - \oint_C \frac{\partial(q-q_d)}{\partial\xi}\delta\phi_{U_\infty} d\xi \\ & +\frac{1}{2}\oint_C \left(q_d^2 - q^2\right)\delta\left(\frac{ds}{d\xi}\right) d\xi \\ & -\int_D \left(\psi L\delta\phi_G + \psi L\delta\phi_E + \psi L\delta\phi_{U_\infty}\right) d\xi\, d\eta - \int_D \left(\psi\frac{\partial\tilde{Q}}{\partial\xi} + \psi\frac{\partial\tilde{P}}{\partial\eta}\right) d\xi\, d\eta.\end{aligned} \tag{4.14}$$

To keep this development simple it is assumed that the solutions are all calculated on an O-mesh.

4.5 Variation of the Perturbation Potential

The next step is to form the adjoint equation. In the case of the linear algebra, equation (1.9) was obtained from (1.7) simply by taking the transpose of the term $\psi^T \left[\frac{\partial R}{\partial w}\right] \delta w$ to obtain $\delta w^T \left[\frac{\partial R}{\partial w}\right]^T \psi$. In the case of the continuous system presented here, this transpose is equivalent to switching the function that L operates on from $\delta\phi$ to ψ. Thus the next logical step is to integrate portions of equation (4.14) by parts in such a way as to obtain the terms $\delta\phi_G L\psi$, $\delta\phi_E L\psi$, and $\delta\phi_{U_\infty} L\psi$, where $L\psi$ will become the left hand side of the continuous adjoint equation. Now since L is a second order operator it is convenient to use the second form of Green's theorem (integrating by parts twice) to switch the order of operation.[1]. Thus, for the term in equation (4.14) that involves $\delta\phi_G$ we can write, after subtracting the term $\int_D \delta\phi_G L\psi d\xi\, d\eta$,

$$\int_D (\psi L\delta\phi_G - \delta\phi_G L\psi)\, d\xi\, d\eta =$$

$$\oint_C \left\{-\psi\rho J\left(A_{12}\delta\phi_{G_\xi} + A_{22}\delta\phi_{G_\eta}\right) + \delta\phi_G \rho J\left(A_{12}\psi_\xi + A_{22}\psi_\eta\right)\right\} d\xi, \qquad (4.15)$$

where the other boundary integrals along the cut line and the outer boundary vanish. The integrals along the cut cancel since both ψ and ϕ_G are assumed to be periodic. $\psi = \phi_G = 0$ at the far field by definition. Therefore, by choosing ψ to satisfy the adjoint equation,

$$L\psi = 0 \qquad (4.16)$$

in the domain with additional constraints such that

$$\begin{aligned} \psi &= 0 \text{ at the far field,} \\ \psi & \quad \text{is periodic along the cut,} \\ \rho J\left(A_{12}\psi_\xi + A_{22}\psi_\eta\right) &= -\frac{\partial(q-q_d)}{\partial\xi} \text{ on the airfoil surface,} \end{aligned} \qquad (4.17)$$

it is possible to rewrite equation (4.14) after cancelling boundary terms as

$$\delta I = -\oint_C \frac{\partial(q-q_d)}{\partial\xi}\delta\phi_E d\xi - \oint_C \frac{\partial(q-q_d)}{\partial\xi}\delta\phi_{U_\infty} d\xi$$

[1]The general form of Green's theorem used here is presented in *Appendix C*

$$+\frac{1}{2}\oint_C \left(q_d^2 - q^2\right)\delta\left(\frac{ds}{d\xi}\right) d\xi$$

$$-\int_D \left(\psi L\delta\phi_E + \psi L\delta\phi_{U_\infty}\right) d\xi\, d\eta - \int_D \left(\psi\frac{\partial \tilde{Q}}{\partial \xi} + \psi\frac{\partial \tilde{P}}{\partial \eta}\right) d\xi\, d\eta$$

$$+\oint_C \left\{\psi\rho J\left(A_{12}\delta\phi_{G_\xi} + A_{22}\delta\phi_{G_\eta}\right)\right\} d\xi. \qquad (4.18)$$

In equation (4.18) the variation in the perturbation potential, $\delta\phi_G$, has been eliminated from all terms except for the last one. In order to complete the process, terms dependent upon $\delta\phi_E$ and $\delta\phi_{U_\infty}$ as well as the last term must be eliminated.

4.6 Variation of the Circulation Potential

The term involving variation of the circulation potential can be written following (4.15) as

$$\int_D \left(\psi L\delta\phi_E - \delta\phi_E L\psi\right) d\xi\, d\eta =$$

$$-\oint_C \psi\rho J\left(A_{12}\delta\phi_{E_\xi} + A_{22}\delta\phi_{E_\eta}\right) d\xi + \oint_C \delta\phi_E\rho J\left(A_{12}\psi_\xi + A_{22}\psi_\eta\right) d\xi$$

$$-\oint_B \left(\delta\phi_E\rho J\left(A_{22} - \frac{V^2}{c^2}\right)\psi_\eta\right) d\xi$$

$$-\oint_{Cut_U} \delta\phi_E\rho J\left(\left(A_{11} - \frac{U^2}{c^2}\right)\psi_\xi + \left(A_{12} - \frac{UV}{c^2}\right)\psi_\eta\right) d\eta$$

$$+\oint_{Cut_L} \delta\phi_E\rho J\left(\left(A_{11} - \frac{U^2}{c^2}\right)\psi_\xi + \left(A_{12} - \frac{UV}{c^2}\right)\psi_\eta\right) d\eta,$$

where B represents the outer boundary and Cut_U and Cut_L represent boundaries along the cut line on the upper and lower halves. The various boundary terms in (4.19) have been derived as follows: First, along the profile, the portion of the boundary integral related to the flux through the surface have been dropped. Meanwhile, at the outer boundary, most of the boundary integral vanish since $\psi = 0$, $\psi_\xi = 0$, and $\delta\phi_{E_\eta} = 0$ by definition throughout the domain. Finally, along the cut line, boundary integrals not present in (4.19) cancel since both ψ and $\delta\phi_{E_\xi}$ are continuous across the cut and $\delta\phi_{E_\eta} = 0$ everywhere. Remembering that ψ must satisfy $L\psi = 0$ with the boundary condition specified by (4.17), the variation in the cost function, after cancelling terms, becomes

$$\begin{aligned}
\delta I \;=\; & -\oint_C \frac{\partial(q-q_d)}{\partial\xi}\delta\phi_{U_\infty} d\xi \\
& +\frac{1}{2}\oint_C \left(q_d^2 - q^2\right)\delta\left(\frac{ds}{d\xi}\right) d\xi \\
& -\int_D \left(\psi L\delta\phi_{U_\infty}\right) d\xi\, d\eta - \int_D \left(\psi\frac{\partial\tilde{Q}}{\partial\xi} + \psi\frac{\partial\tilde{P}}{\partial\eta}\right) d\xi\, d\eta \\
& +\oint_C \left\{\psi\rho J\left(A_{12}\delta\left(\phi_{G_\xi}+\phi_{E_\xi}\right) + A_{22}\delta\left(\phi_{G_\eta}+\phi_{E_\eta}\right)\right)\right\} d\xi \\
& +\oint_B \left(\delta\phi_E \rho J\left(A_{22}-\frac{V^2}{c^2}\right)\psi_\eta\right) d\xi \\
& +\oint_{Cut_U} \delta\phi_E\rho J\left(\left(A_{11}-\frac{U^2}{c^2}\right)\psi_\xi + \left(A_{12}-\frac{UV}{c^2}\right)\psi_\eta\right) d\eta \\
& -\oint_{Cut_L} \delta\phi_E\rho J\left(\left(A_{11}-\frac{U^2}{c^2}\right)\psi_\xi + \left(A_{12}-\frac{UV}{c^2}\right)\psi_\eta\right) d\eta.
\end{aligned}$$

4.7 Variation of the Uniform Flow Potential

Further simplification occurs by expanding the term related to the variation of the uniform flow portion of the potential:

$$\begin{aligned}
& \int_D \left(\psi L\delta\phi_{U_\infty} - \delta\phi_{U_\infty} L\psi\right) d\xi\, d\eta = \\
- \quad & \oint_C \psi\rho J\left(A_{12}\delta\phi_{U_{\infty_\xi}} + A_{22}\delta\phi_{U_{\infty_\eta}}\right) d\xi + \oint_C \delta\phi_{U_\infty}\rho J\left(A_{12}\psi_\xi + A_{22}\psi_\eta\right) d\xi.
\end{aligned}$$

All additional boundary terms at the far field vanish since $\psi = \psi_\xi = 0$, and we will restrict ourselves to design cases in which the flight conditions except for α will remain fixed ($\delta\phi_{U_{\infty_\eta}} = 0$). We could have just as easily applied the $\delta\phi_{U_{\infty_\eta}} = 0$ condition from the beginning of these derivations and avoided some of the details in this section. However, by including these details, we gain an understanding of how a change to the uniform flow such as Mach number can get into the act. Along the cut it is again assumed that ψ is periodic. The variation in the cost function can now be written as

$$\delta I \;=\; -\oint_C \frac{\partial(q-q_d)}{\partial\xi}\delta\phi_{U_\infty} d\xi$$

$$
\begin{aligned}
&+\frac{1}{2}\oint_C \left(q_d^2 - q^2\right)\delta\left(\frac{ds}{d\xi}\right) d\xi \\
&+\int_D \left(\tilde{Q}\frac{\partial\psi}{\partial\xi} + \tilde{P}\frac{\partial\psi}{\partial\eta}\right) d\xi\, d\eta \\
&+\oint_C \tilde{P}\psi d\xi \\
&+\oint_C \left\{\psi\rho J\left(A_{12}\delta\left(\phi_{G_\xi} + \phi_{E_\xi} + \phi_{U_{\infty_\xi}}\right) + A_{22}\delta\left(\phi_{G_\eta} + \phi_{E_\eta} + \phi_{U_{\infty_\eta}}\right)\right)\right\} d\xi \\
&+\oint_B \left(\delta\phi_E\rho J\left(A_{22} - \frac{V^2}{c^2}\right)\psi_\eta\right) d\xi \\
&+\oint_{Cut_U} \delta\phi_E\rho J\left(\left(A_{11} - \frac{U^2}{c^2}\right)\psi_\xi + \left(A_{12} - \frac{UV}{c^2}\right)\psi_\eta\right) d\eta \\
&-\oint_{Cut_L} \delta\phi_E\rho J\left(\left(A_{11} - \frac{U^2}{c^2}\right)\psi_\xi + \left(A_{12} - \frac{UV}{c^2}\right)\psi_\eta\right) d\eta,
\end{aligned}
$$

where the terms involving the grid variations $\tilde{Q}$ and $\tilde{P}$ have been integrated by parts and the resulting boundary integrals except at the profile surface vanish either due to periodicity at the cut or because the mesh metrics are frozen at the far field. Now it is possible to substitute the definition for $\tilde{P}$ in the above equation and use the fact that both $V = 0$ and $\delta V = 0$ such that the fourth and fifth terms cancel to give:

$$
\begin{aligned}
\delta I = &+\frac{1}{2}\oint_C \left(q_d^2 - q^2\right)\delta\left(\frac{ds}{d\xi}\right) d\xi \\
&+\int_D \left(\tilde{Q}\frac{\partial\psi}{\partial\xi} + \tilde{P}\frac{\partial\psi}{\partial\eta}\right) d\xi\, d\eta \\
&+\oint_B \left(\delta\phi_E\rho J\left(A_{22} - \frac{V^2}{c^2}\right)\psi_\eta\right) d\xi \\
&+\oint_{Cut_U} \delta\phi_E\rho J\left(\left(A_{11} - \frac{U^2}{c^2}\right)\psi_\xi + \left(A_{12} - \frac{UV}{c^2}\right)\psi_\eta\right) d\eta \\
&-\oint_{Cut_L} \delta\phi_E\rho J\left(\left(A_{11} - \frac{U^2}{c^2}\right)\psi_\xi + \left(A_{12} - \frac{UV}{c^2}\right)\psi_\eta\right) d\eta.
\end{aligned}
$$

4.8 THE KUTTA CONDITION

The only remaining term involving a variation in the potential relates to a variation in the circulation potential $\delta\phi_E$. This term may be eliminated by employing the equation for the Kutta condition:

$$\begin{aligned}\phi_E + \Gamma\frac{\theta}{2\pi} &= 0\\ \delta\phi_E + \Gamma\delta\left(\frac{\theta}{2\pi}\right) + \delta\Gamma\frac{\theta}{2\pi} &= 0.\end{aligned}$$

In the case where lift is fixed ($\delta\Gamma = 0$), *a condition that will be forced for all design examples using the potential flow formulation*, the jump in the variation across the cut line may be written as

$$\delta\phi_E \,|_l^u = -\Gamma\delta\left(\frac{\theta}{2\pi}\right)|_l^u\,.$$

However, since

$$\delta\left(\frac{\theta}{2\pi}\right)|_l^u = 0,$$

it follows that the boundary integrals involving the cut lines cancel each other out of the expression for δI. Furthermore, the implication of fixed circulation also forces $\delta\phi_E = 0$ to be satisfied along the outer boundary since it is a region where the mesh is assumed to be fixed. The final expression for the variation in the cost function may therefore be written as

$$\begin{aligned}\delta I &= \frac{1}{2}\oint_C \left(q_d^2 - q^2\right)\delta\left(\frac{ds}{d\xi}\right)d\xi\\ &\quad + \int_D \left(\tilde{Q}\frac{\partial\psi}{\partial\xi} + \tilde{P}\frac{\partial\psi}{\partial\eta}\right)d\xi\, d\eta. \qquad (4.19)\end{aligned}$$

It must be noted that since a fixed circulation formulation has been chosen, which mandates that angle of attack α is allowed to vary, in order to be consistent the term $\frac{\partial I}{\partial\alpha}\delta\alpha$ must be added to our definition of the variation in the cost function. The remaining variation $\delta\alpha$ would then create a supplementary boundary condition on the adjoint equation which would be determined by using the definition of fixed circulation. However in practical tests, when the cost function is chosen as the inverse problem, the addition of this term showed no discernible difference. In the case of other cost functions, such as drag, it turns out that this additional term can not be neglected, and thus the derivations in Section (4.13.1) include the term. In the recent work of Ta'asan et al. [71] it has been shown that a Dirac delta function is necessary as a boundary forcing term for the case of $\delta\Gamma \neq 0$, in order to eliminate $\delta\phi_E$ properly for, say, a fixed α case.

4.9 Algorithm Outline

Considering the cumbersome integration by parts, it is gratifying that the final form of equation (4.19) remains relatively simple. In order to solve for δI, the following steps are necessary:

1. Solve the flow equation at the initial point according to Section (2.4).
2. Solve the adjoint equation
$$L\psi = 0 \text{ in } D$$
subject to
$$\rho J\left[A_{12}\psi_\xi + A_{22}\psi_\eta\right] = -\frac{\partial(q-q_d)}{\partial\xi} \quad \text{on } C,$$
ψ is periodic along the cut line,
$\psi = 0$ at the outer boundary.
3. Solve for the mesh metric variations, $\tilde{Q}$ and $\tilde{P}$.
4. Calculate the gradient $\mathcal{G}$ or a specific variation δI using equation (4.19).

The method of solving the flow variables has already been discussed in Section (2.4). The method of obtaining the mesh metric variations as well as a more detailed presentation of the design algorithm and discussion of the design variables is given in *Chapter 6*. The details of the discretization and solution algorithm for the adjoint equation will be discussed in *Chapter 7*.

4.10 Discontinuities

If the flow is subsonic, this procedure should converge toward the desired speed distribution since the solution will remain smooth and no unbounded derivatives will appear. If the flow is transonic, however, one must allow for the appearance of shock waves in the trial solutions even if q_d is smooth. In such instances, $q - q_d$ is not differentiable and the boundary condition for the adjoint equation contains a jump in the forcing term. If the target speed distribution also has shock wave(s), the boundary condition may have multiple step discontinuities. The solution to the adjoint problem for these cases can become difficult because they are analogous to solving flow solution problems with slope discontinuities in their airfoil surfaces. Furthermore, the validity of using the presented continuous sensitivity analysis is strictly tied to the continuity of at least first derivatives in both the flow and adjoint solutions due to the multiple use of integration by parts.

In the presence of shock waves the analysis should preoceed from the weak form of the governing differential equation. It can be argued, however, that the employment of artificial dissipation in both the state and co-state fields, which smooths any unbounded derivatives in the solutions, implies that the integration by parts developed in Sections (4.5 - 4.7) remains consistent. As in the case of the flow solution problem, the alternative is to use a shock-fitting approach. To treat the solution of the adjoint equation in this manner, domain decomposition along the shock boundaries would have to be implemented, with separate integration by parts used in each domain. The resulting boundary integrals would then be forced to cancel by matching conditions along the interface. It seems reasonable that this level of sophistication is unwarranted since weakly smoothed shocks are more than adequate for most reasonable design problems. Furthermore, it must be remembered that *the solution of the adjoint equation is used only to obtain gradient information. It does not affect the flow solution accuracy in any way*. Therefore, if reasonable gradient information is obtained, the design process should still proceed. The only difficulty left is the treatment of the adjoint solution for cases that contain discontinuities in the forcing function at the surface.

4.11 IMPLICIT SMOOTHING

Two methods have been developed here to improve the solution robustness in the case of jump discontinuities. The first follows the work of Jameson in using the idea of implicit smoothing to remove any discrete jumps from the forcing term [55, 54]. The cost function of equation (4.6) is thus replaced with

$$I = \frac{1}{2}\oint_C \left(v_1 \Theta^2 + v_2 \left(\frac{d\Theta}{d\xi} \right)^2 \right) d\xi, \tag{4.20}$$

where v_1 and v_2 are parameters, and the periodic function $\Theta(\xi)$ satisfies the equation

$$v_1 \Theta - v_2 \frac{d^2\Theta}{d\xi^2} = q - q_d. \tag{4.21}$$

The new definition of I, like the one given by (4.6), is a particular form of a general class of cost functions that seek to minimize the difference between the target and actual pressure distributions. For the case of (4.20) δI may be rewritten as

$$\begin{aligned} \delta I &= \oint_C \left(v_1 \Theta\, \delta\Theta + v_2 \frac{d\Theta}{d\xi} \frac{d}{d\xi}\, \delta\Theta \right) d\xi \\ &= \oint_C \Theta \left(v_1 \delta\Theta - v_2 \frac{d^2}{d\xi^2}\, \delta\Theta \right) d\xi = \oint_C \Theta\, \delta q\, d\xi. \end{aligned}$$

Thus, Θ replaces $q - q_d$ in the previous formula and the boundary condition (4.17) is modified such that

$$\rho J \left(A_{12} \psi_\xi + A_{22} \psi_\eta \right) = -\frac{\partial}{\partial \xi}(\Theta) \ \text{ on } C. \tag{4.22}$$

By choosing appropriate values for v_1 and v_2 it is possible to smooth the cost function in such a way that greater robustness of the adjoint solver is maintained. Furthermore, even if at the initial point there is a significant difference between the reformulated and original cost functions, this difference will decrease as the design process proceeds. As the target design is approached the magnitude of boundary discontinuities will drop and thus require less smoothing and hence a greater correspondence between the modified and original cost functions.

4.12 Shock Wave Unweighting

Motivation for the second approach begins with the realization that the use of dissipation in the flow solution method forces the jumps in the boundary term arising from the trial solution to become somewhat smeared. This is not necessarily the case for the jumps created by the target distribution, which may have true discontinuities. Furthermore, target distributions with jump discontinuities will in general represent unattainable pressure distributions at least near the shock region. The implication is that there is little hope of matching point by point a target speed distribution through a shock structure. The shock structures obtained from practical flow solvers are often not models of physical phenomena, but are instead artificially created to permit efficient and robust solution calculation. In practice, even where the solution very closely matches the target distribution, points within and very near the shock structure tend to be mismatched compared with the rest of the discrete representation. In fact, it is quite possible that large differences between the target and the actual solution in the shock region may dominate the adjoint solution and hinder the attainment of the desired speed distribution.

On the basis of this insight, an alternative approach to promoting successful adjoint solutions can be formulated. The key is to unweight the integrand of the cost function in regions which span a shock wave in either the trial solution or the target solution. Therefore, the cost function is rewritten in the case of a single shock as,

$$I = \frac{1}{2}\oint_C (1 - \mathrm{g}_u(\xi_u))(q - q_d)^2 \left(\frac{ds}{d\xi}\right) d\xi, \qquad (4.23)$$

where $\mathrm{g}_u(\xi_u)$ is a Gaussian distribution centered at the shock location ξ_u. The width of the Gaussian must be adjusted in a manner similar to the adjustment of the terms v_1 and v_2 used in implicit smoothing. In the center of the shock structure the cost function goes to 0 as g_u approaches 1, while far away from the shock, the cost function retains its full value as g_u approaches 0. In essence, no attempt is made to mimic the shock structure in the solution of the adjoint equation. The test used to detect shocks compares C_p with C_p^* and checks the sign and magnitude of the pressure gradient. Note that this test is far more restrictive than in the case of implicit smoothing which tends to smooth any regions with large gradients in the forcing term, irrespective of whether they come from a shock or not. Shock structures are not the only locations where the boundary condition for the adjoint equation may contain high gradients or discontinuities. For example, the region about the leading edge stagnation point may have high gradients since it is unlikely that both the trial solution and the target distribution will have corresponding stagnation locations. While the implicit smoothing technique will smear the forcing function in the stagnation region, the shock unweighting technique will operate only in regions containing a shock in either the trial solution or the target.

4.13 OTHER COST FUNCTIONS

Thus far, the presentation has been restricted to the inverse problem. Traditional inverse methods, described in Sections (1.2.1) and (1.2.2), in general, mandate that the target is attainable because the convergence of the flow-field requires that the pressure match the target. As mentioned earlier, the inverse method presented here does not depend on the target pressure distribution's being attainable. The advantage of this difference will also be explored in *Chapter 8*. Even though the current formulation is more robust than many inverse methods, it thus far still suffers from the classic problem related to all inverse methods: it does not allow for the exploration of a variety of objective functions. For practical design of airfoils, objective functions other than simply the target speed are vital. Truly optimum designs, even from the inviscid perspective alone, must look directly at the aerodynamic coefficients governing performance (C_d, C_l, C_l/C_d, etc.). Even the most brilliant aerodynamicist will not be able, in general, to prescribe the 3-dimensional pressure distribution which achieves, say, minimum drag at a given C_l. To generalize the design method it is therefore necessary to develop other cost functions within the same framework.

4.13.1 Drag

For the case where the cost function is drag, (4.6) is replaced by

$$\begin{aligned} I = C_d &= C_a \cos\alpha + C_n \sin\alpha \\ &= \frac{1}{\frac{1}{2}\gamma P_\infty M_\infty^2 \bar{c}} \oint_C p \left(\frac{\partial y}{\partial \xi} \cos\alpha - \frac{\partial x}{\partial \xi} \sin\alpha \right) d\xi, \end{aligned} \tag{4.24}$$

where C_a and C_n are the axial and normal force coefficients respectively. The first variation of the cost function is now

$$\begin{aligned} \delta I &= \tilde{\delta} C_a \cos\alpha + \tilde{\delta} C_n \sin\alpha \\ &+ \left[\frac{\partial C_a}{\partial \alpha} \cos\alpha + \frac{\partial C_n}{\partial \alpha} \sin\alpha \right] \delta\alpha \\ &+ \left[-C_a \sin\alpha + C_n \cos\alpha \right] \delta\alpha, \end{aligned} \tag{4.25}$$

where $\tilde{\delta}$ refers to variations independent of changes in α. This split is necessary since the flow-field governing equations are written in terms of a fixed α. However, the problem commonly of interest in airfoil design is that of fixed lift or circulation Γ. To implement such a method, an outer iteration is incorporated in the flow solver which changes α by examining the lift curve slope. Without this additional expression to obtain α, the variation of the aerodynamic coefficients with respect to α must be treated separately; hence the form of equation (4.25). Now in order to obtain δI relative to a fixed lift, an additional constraint is necessary:

$$\begin{aligned} \delta C_l = 0 &= \delta \left[C_n \cos\alpha - C_a \sin\alpha \right] \\ &= \tilde{\delta} C_n \cos\alpha - \tilde{\delta} C_a \sin\alpha \\ &+ \left[\frac{\partial C_n}{\partial \alpha} \cos\alpha - \frac{\partial C_a}{\partial \alpha} \sin\alpha \right] \delta\alpha \\ &+ \left[-C_n \sin\alpha - C_a \cos\alpha \right] \delta\alpha. \end{aligned} \tag{4.26}$$

Combining (4.25) and (4.26) by eliminating $\delta\alpha$ gives

$$\begin{aligned} \delta C_d &= \tilde{\delta} C_a \cos\alpha + \tilde{\delta} C_n \sin\alpha \\ &- \frac{\left[C_l + \frac{\partial C_a}{\partial \alpha} \cos\alpha + \frac{\partial C_n}{\partial \alpha} \sin\alpha \right]}{\left[-C_d + \frac{\partial C_n}{\partial \alpha} \cos\alpha - \frac{\partial C_a}{\partial \alpha} \sin\alpha \right]} \left[\tilde{\delta} C_n \cos\alpha - \tilde{\delta} C_a \sin\alpha \right]. \end{aligned} \tag{4.27}$$

With all variations now in terms of $\delta C'_n$ and $\delta C'_a$ it is possible to carry out the variations further in terms of δp and the δ(metrics):

$$\delta I = \frac{1}{\frac{1}{2}\gamma P_\infty M_\infty^2 \bar{c}} \oint_C \delta p \left[\left(\frac{\partial y}{\partial \xi} \cos\alpha - \frac{\partial x}{\partial \xi} \sin\alpha \right) + \Omega \left(\frac{\partial x}{\partial \xi} \cos\alpha + \frac{\partial y}{\partial \xi} \sin\alpha \right) \right] d\xi$$
$$+ \frac{1}{\frac{1}{2}\gamma P_\infty M_\infty^2 \bar{c}} \oint_C p \left[\left(\delta\left(\frac{\partial y}{\partial \xi}\right) \cos\alpha - \delta\left(\frac{\partial x}{\partial \xi}\right) \sin\alpha \right) + \Omega \left(\delta\left(\frac{\partial x}{\partial \xi}\right) \cos\alpha + \delta\left(\frac{\partial y}{\partial \xi}\right) \sin\alpha \right) \right] d\xi \tag{4.28}$$

where

$$\Omega = \frac{\left[C_l + \frac{\partial C_a}{\partial \alpha} \cos\alpha + \frac{\partial C_n}{\partial \alpha} \sin\alpha \right]}{\left[-C_d + \frac{\partial C_n}{\partial \alpha} \cos\alpha - \frac{\partial C_a}{\partial \alpha} \sin\alpha \right]}.$$

Now since

$$p = \frac{\rho^\gamma}{\gamma M_\infty^2}$$

and

$$\delta p = \frac{\frac{\gamma}{\gamma-1} \rho\, \delta \left\{ 1 + \frac{\gamma-1}{2} M_\infty^2 \left(1 - \phi_s^2\right) \right\}}{\gamma M_\infty^2}$$
$$= -\rho \phi_s \delta\phi_s = -\rho\phi_s \left(\delta\left(\frac{\partial \phi}{\partial \xi}\right) \frac{\partial \xi}{\partial s} + \frac{\partial \phi}{\partial \xi} \delta\left(\frac{\partial \xi}{\partial s}\right) \right),$$

it can be shown that

$$\delta I = \frac{1}{\frac{1}{2}\gamma P_\infty M_\infty^2 \bar{c}} \oint_C \delta\phi \frac{\partial}{\partial \xi} \left\{ \rho\phi_s \frac{\partial \xi}{\partial s} \left[\left(\frac{\partial y}{\partial \xi} \cos\alpha - \frac{\partial x}{\partial \xi} \sin\alpha \right) + \Omega \left(\frac{\partial x}{\partial \xi} \cos\alpha + \frac{\partial y}{\partial \xi} \sin\alpha \right) \right] \right\} d\xi$$
$$- \frac{1}{\frac{1}{2}\gamma P_\infty M_\infty^2 \bar{c}} \oint_C \rho\phi_s \frac{\partial \phi}{\partial \xi} \delta\left(\frac{\partial \xi}{\partial s}\right) \left[\left(\frac{\partial y}{\partial \xi} \cos\alpha - \frac{\partial x}{\partial \xi} \sin\alpha \right) + \Omega \left(\frac{\partial x}{\partial \xi} \cos\alpha + \frac{\partial y}{\partial \xi} \sin\alpha \right) \right] d\xi$$
$$+ \frac{1}{\frac{1}{2}\gamma P_\infty M_\infty^2 \bar{c}} \oint_C p \left[\left(\delta\left(\frac{\partial y}{\partial \xi}\right) \cos\alpha - \delta\left(\frac{\partial x}{\partial \xi}\right) \sin\alpha \right) + \Omega \left(\delta\left(\frac{\partial x}{\partial \xi}\right) \cos\alpha + \delta\left(\frac{\partial y}{\partial \xi}\right) \sin\alpha \right) \right] d\xi. \tag{4.29}$$

Thus (4.19) is replaced by

$$\delta I = - \frac{1}{\frac{1}{2}\gamma P_\infty M_\infty^2 \bar{c}} \oint_C \rho\phi_s \frac{\partial \phi}{\partial \xi} \delta\left(\frac{\partial \xi}{\partial s}\right) \left[\left(\frac{\partial y}{\partial \xi} \cos\alpha - \frac{\partial x}{\partial \xi} \sin\alpha \right) + \Omega \left(\frac{\partial x}{\partial \xi} \cos\alpha + \frac{\partial y}{\partial \xi} \sin\alpha \right) \right] d\xi$$
$$+ \frac{1}{\frac{1}{2}\gamma P_\infty M_\infty^2 \bar{c}} \oint_C p \left[\left(\delta\left(\frac{\partial y}{\partial \xi}\right) \cos\alpha - \delta\left(\frac{\partial x}{\partial \xi}\right) \sin\alpha \right) + \Omega \left(\delta\left(\frac{\partial x}{\partial \xi}\right) \cos\alpha + \delta\left(\frac{\partial y}{\partial \xi}\right) \sin\alpha \right) \right] d\xi$$
$$+ \int_D \left(\bar{Q} \frac{\partial \psi}{\partial \xi} + \bar{P} \frac{\partial \psi}{\partial \eta} \right) d\xi\, d\eta \tag{4.30}$$

where the boundary condition on ψ, (4.17), is replaced with

$$\rho J \left(A_{12}\psi_\xi + A_{22}\psi_\eta\right) = +\frac{\partial}{\partial\xi}\left\{\frac{\rho\phi_s\frac{\partial\xi}{\partial s}}{\frac{1}{2}\gamma P_\infty M_\infty^2 \bar{c}}\left[\left(\frac{\partial y}{\partial\xi}\cos\alpha - \frac{\partial x}{\partial\xi}\sin\alpha\right) + \Omega\left(\frac{\partial x}{\partial\xi}\cos\alpha + \frac{\partial y}{\partial\xi}\sin\alpha\right)\right]\right\}. \tag{4.31}$$

Note that by setting $\Omega = 0$ in all of the above equations, the boundary terms can be formulated such that they obtain δC_d for a fixed α; $\delta\alpha = 0$. This is not useful here since the field equation for the adjoint was derived for fixed Γ. However, later in the Euler formulation, this approach may be useful. An important complication that is caused by this use of a fixed lift development is that values for $\frac{\partial C_a}{\partial\alpha}$ and $\frac{\partial C_n}{\partial\alpha}$ must be available prior to the calculation of the adjoint solution. For all the cases to be presented later in this thesis, where they are necessary, these coefficients are calculated by finite differencing two flow solutions with slightly different lift coefficients at the end of each design iteration. Note that this modification is also required for the design method using the Euler equations.

4.13.2 Combined cost functions

Other cost functions which rely on surface integrated aerodynamic quantities can be developed similarly. Further, it is possible to take combinations of these functions to create more sophisticated quantities of merit. For example, to achieve the best approximation to a target speed distribution while simultaneously reducing the drag, (4.6) and (4.24) can be combined, giving

$$I = \Lambda_1 \frac{1}{2}\oint_C (q - q_d)^2 \left(\frac{ds}{d\xi}\right) d\xi + \Lambda_2 C_d.$$

The variation of I may then be written

$$\begin{aligned}\delta I = \quad & \Lambda_1\left\{\frac{1}{2}\oint_C (q_d^2 - q^2)\,\delta\left(\frac{ds}{d\xi}\right) d\xi\right\}\\ + \quad & \frac{\Lambda_2}{\frac{1}{2}\gamma P_\infty M_\infty^2 \bar{c}}\left\{-\oint_C \rho\phi_s\frac{\partial\phi}{\partial\xi}\delta\left(\frac{\partial\xi}{\partial s}\right)\left[\left(\frac{\partial y}{\partial\xi}\cos\alpha - \frac{\partial x}{\partial\xi}\sin\alpha\right) + \Omega\left(\frac{\partial x}{\partial\xi}\cos\alpha + \frac{\partial y}{\partial\xi}\sin\alpha\right)\right] d\xi\right.\end{aligned}$$

$$+\oint_C p\left[\left(\delta\left(\frac{\partial y}{\partial \xi}\right)\cos\alpha - \delta\left(\frac{\partial x}{\partial \xi}\right)\sin\alpha\right) + \Omega\left(\delta\left(\frac{\partial x}{\partial \xi}\right)\cos\alpha + \delta\left(\frac{\partial y}{\partial \xi}\right)\sin\alpha\right)\right]d\xi\right\}$$

$$+ \int_D \frac{\partial \psi}{\partial \xi}\tilde{Q} + \frac{\partial \psi}{\partial \eta}\tilde{P}\, d\xi d\eta.$$

The surface boundary condition for the adjoint problem works out to be the simple summation,

$$\begin{aligned}\rho J\left(A_{12}\psi_\xi + A_{22}\phi_\eta\right) = \frac{\partial}{\partial \xi}\Bigg[& -\Lambda_1 \quad (q - q_d) \\ & +\Lambda_2 \quad \frac{\rho \phi_s \frac{\partial \xi}{\partial s}}{\frac{1}{2}\gamma P_\infty M_\infty^2 \bar{c}}\left[\left(\frac{\partial y}{\partial \xi}\cos\alpha - \frac{\partial x}{\partial \xi}\sin\alpha\right)\right. \\ & \left.\left. +\Omega\left(\frac{\partial x}{\partial \xi}\cos\alpha + \frac{\partial y}{\partial \xi}\sin\alpha\right)\right]\right]. \end{aligned} \tag{4.32}$$

Therefore, by implementing such a combined method and letting the user choose the weights $\Lambda_1, \Lambda_2, \cdots$, it is possible to create an extremely versatile design procedure.

Desired cost functions that relate to field properties instead of boundary properties, such as such limiting the shock strength throughout the entire domain, may also be treated within the framework of control theory-based design. In place of having a forcing term on the boundary condition of the adjoint solver, these cost functions would have source terms applied to the adjoint domain equations. Cost functions of this type will not be explored in this research.

Chapter 5

Control Theory Applied to the Euler Equations

This chapter develops the general application of control theory for aerodynamic shape optimization using the Euler equations. The work here follows the initial ideas proposed by Jameson [54] and later expanded by both Jameson [59, 58] and (independently) Lewis and Agarwal [76, 75]. The development parallels that of the preceding chapter in many respects. Nevertheless, major differences are present. First, the Euler equations operate on the vector valued function $\mathbf{w}$ as opposed to the scalar ϕ. Second, since the time dependent Euler equations are hyperbolic, instead of elliptic, the general property of self-adjointness is lost. This implies that the transpose operator present in the general derivation of equation (1.9) will play a key rôle in the solution of the adjoint formulation for the Euler equations. Seen another way, the second order potential flow equation is well posed as a boundary value problem while the first order Euler equations require careful treatment of the boundary conditions to avoid over-specification. These differences result in fundamental alterations in the development of the adjoint system.

As stated earlier, both the Euler and the potential flow equations are inviscid formulations. The Euler equations are less simplified because the flow is not assumed to be either irrotational or isentropic. For the study of airfoil design, this difference is most evident in the modeling of compression shock waves. The Euler equations correctly model these flow discontinuities and their associated entropy production.

5.1 The Euler Equations

Consider again the flow in a domain D. The profile defines the inner boundary C, while the outer boundary B is assumed to be distant from the profile. Just as in *Chapter 2*, let p, ρ, u, v, E and H denote the pressure, density, Cartesian velocity components, total energy and total enthalpy. Recall that for a perfect gas

$$p = (\gamma - 1)\rho \left\{ E - \frac{1}{2}\left(u^2 + v^2\right)\right\} \tag{5.1}$$

and

$$\rho H = \rho E + p. \tag{5.2}$$

Reiterating expression (2.3), the Euler equations may be written in differential form as

$$\frac{\partial \mathbf{w}}{\partial t} + \frac{\partial \mathbf{f}}{\partial x} + \frac{\partial \mathbf{g}}{\partial y} = 0 \quad \text{in } D, \tag{5.3}$$

where x and y are again the Cartesian coordinates, t is the time coordinate, and

$$\mathbf{w} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{bmatrix}, \quad \mathbf{f} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ \rho u H \end{bmatrix}, \quad \mathbf{g} = \begin{bmatrix} \rho v \\ \rho v u \\ \rho v^2 + p \\ \rho v H \end{bmatrix}. \tag{5.4}$$

As already defined, the coordinate transformations may be written as the matrix

$$K = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{bmatrix},$$

where the Jacobian is

$$J = \det(K) = \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi}.$$

Introduce contravariant velocity components

$$\begin{bmatrix} U \\ V \end{bmatrix} = K^{-1}\begin{bmatrix} u \\ v \end{bmatrix} = \frac{1}{J}\begin{bmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial x}{\partial \eta} \\ -\frac{\partial y}{\partial \xi} & \frac{\partial x}{\partial \xi} \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix}.$$

The Euler equations can thus be written in divergence form for the computational coordinate plane as

$$\frac{\partial W}{\partial t}+\frac{\partial F}{\partial \xi}+\frac{\partial G}{\partial \eta}=0 \quad \text{in } D, \tag{5.5}$$

with

$$W = J\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ \rho E\end{bmatrix},$$

$$F = J\begin{bmatrix}\rho U\\ \rho U u+\frac{\partial \xi}{\partial x}p\\ \rho U v+\frac{\partial \xi}{\partial y}p\\ \rho U H\end{bmatrix}, \quad G = J\begin{bmatrix}\rho V\\ \rho V u+\frac{\partial \eta}{\partial x}p\\ \rho V v+\frac{\partial \eta}{\partial y}p\\ \rho V H\end{bmatrix}. \tag{5.6}$$

Now since the solution method outlined in *Chapter 2* uses a computational coordinate system that conforms to the airfoil section in such a way that the surface C is represented by $\eta = 0$ (i.e., an O-mesh), equation (5.6) is satisfied in the domain, and the flow tangency,

$$V = 0 \quad \text{on } C, \tag{5.7}$$

is satisfied on C.

5.2 Cost Function for the Inverse Problem

As a first example of the use of control theory for the Euler equations, consider again the case of the inverse problem. In contrast to the potential flow formulation, here it is convenient to specify a target pressure distribution as opposed to a target velocity distribution.

5.2.1 NON-UNIQUENESS OF THE DESIGN PROCESS

NON-UNIQUENESS DUE TO NONLINEAR GOVERNING EQUATIONS

Before developing the necessary equations for this cost function it is worthwhile to consider the issue of the possibility of non-uniqueness of the design process. Since both the potential flow and Euler equations are nonlinear they do not preclude the possibility of non-uniqueness even in the flow solution. In fact, evidence which supports the presence of more than one stable flow solution, at least for inviscid flows given prescribed airfoil shapes and operating conditions, can be gathered from Jameson's works [100, 56]. It is interesting that such solutions can be shown to be present for both the potential flow and Euler equations, although it seems that they occur somewhat more often for the potential flow equation.

With such well documented non-uniqueness in flow analysis cases using either system, the possible consequences of non-uniqueness on the design process must be considered. In the case of design, let us say for the inverse problem presently being examined, the use of nonlinear governing equations as a constraint implies that there is a possibility of more than one airfoil shape producing exactly the same pressure distribution. For the Euler equations, this possibility may be accentuated by the fact that specifying only one quantity at the surface, such as pressure, is not enough to determine the flow uniquely due to the possible production of entropy. For the potential flow equation, prescribing the pressure (or the velocity) at the surface exactly determines the desired state. In any event, neither system seems to eliminate the unlikely case of having two or more airfoils leading to the same pressure distribution. In one respect, this may seem to be irrelevant from the design point of view since any one of the solutions satisfying the target pressure distribution is in fact a valid solution of the design problem. On the other hand, the presence of more than one valid design solution implies that a multimodal design space may be present. And since the design methods being used in this thesis are gradient-based, and hence assume a unimodal design space in order to achieve a global minimum, the eventual consequence of developing design methods in conjunction with nonlinear governing equations is that it increases the possibility that only a local optimum may be achieved during a particular design effort. The important point is that the possibility of multiple airfoils admitting the same target pressure distribution should not have consequences that prevent the design process from moving in the direction of an improved design. There remains the possibility that given a multimodal design space caused by non-uniqueness of the solution, the design procedure may jump between adjacent valleys during the course of a line search and thereby stall convergence.

NON-UNIQUENESS DUE TO POOR PROBLEM FORMULATION

A far more likely route to non-uniqueness in the design problem is also of concern. This alternate form is best illustrated by taking the case of drag as the cost function to be minimized. If the conditions under which the airfoil operates are such that a zero inviscid drag case can be achieved by modifying the chosen set of design variables, then it is highly likely that many such cases could be attained. In such a situation the design space is characterized by either a large flat basin or many local minima that all have the same minimum value. Again in one sense this does not pose a significant problem since achieving any one of the multiple minimum states still implies that the process has succeeded. However, if an airplane designer were asked to choose between many such airfoils, all having zero inviscid drag, he or she is likely to have a preference based on other aerodynamic or geometric properties. The main conclusion is that great care must be given to the choice of the cost function.

5.2.2 VARIATION OF THE COST FUNCTION FOR THE INVERSE PROBLEM

Returning to the problem at hand, the cost function may be defined as

$$I = \frac{1}{2}\oint_C (p - p_d)^2\, ds = \frac{1}{2}\oint_C (p - p_d)^2 \left(\frac{ds}{d\xi}\right) d\xi, \qquad (5.8)$$

where p_d is the desired pressure. The design problem is now treated as a control problem where the control function is the airfoil shape, which is to be chosen to minimize I subject to the constraints defined by the flow equations (5.5–5.7). A variation in the shape will cause a variation, δp, in the pressure in addition to a variation in the geometry, and consequently, the variation in the cost function becomes

$$\delta I = \oint_C (p - p_d)\delta p \left(\frac{ds}{d\xi}\right) d\xi + \frac{1}{2}\oint_C (p - p_d)^2 \delta \left(\frac{ds}{d\xi}\right) d\xi. \tag{5.9}$$

5.3 Variations of the Governing Equations

Following closely the development for potential flows, the next step is to determine how a variation in the airfoil shape causes variations in both δp and $\delta\left(\frac{ds}{d\xi}\right)$. Since p depends on w through the equation of state (5.3–5.4), the variation δp can be determined from the variation δw. Therefore the next goal is to develop the first variation of the steady state governing equations in terms of δw. Define the Jacobian matrices given in *Chapter 2*,

$$\mathbf{A_1} = \frac{\partial \mathbf{f}}{\partial \mathbf{w}}, \quad \mathbf{A_2} = \frac{\partial \mathbf{g}}{\partial \mathbf{w}}, \quad \mathbf{C_i} = \sum_j J K_{ij}^{-1} \mathbf{A_j}. \tag{5.10}$$

The relations for $\mathbf{C_i}$ may be written out as

$$\mathbf{C_1} = \frac{\partial \mathbf{f}}{\partial \mathbf{w}}\left(J\frac{\partial \xi}{\partial x}\right) + \frac{\partial \mathbf{g}}{\partial \mathbf{w}}\left(J\frac{\partial \xi}{\partial y}\right), \quad \mathbf{C_2} = \frac{\partial \mathbf{f}}{\partial \mathbf{w}}\left(J\frac{\partial \eta}{\partial x}\right) + \frac{\partial \mathbf{g}}{\partial \mathbf{w}}\left(J\frac{\partial \eta}{\partial y}\right).$$

Then the equation for δw in the steady state becomes

$$\frac{\partial}{\partial \xi}(\delta F) + \frac{\partial}{\partial \eta}(\delta G) = 0, \tag{5.11}$$

where

$$\begin{aligned} \delta F &= \mathbf{C_1}\delta\mathbf{w} + \delta\left(J\frac{\partial \xi}{\partial x}\right)\mathbf{f} + \delta\left(J\frac{\partial \xi}{\partial y}\right)\mathbf{g} \\ \delta G &= \mathbf{C_2}\delta\mathbf{w} + \delta\left(J\frac{\partial \eta}{\partial x}\right)\mathbf{f} + \delta\left(J\frac{\partial \eta}{\partial y}\right)\mathbf{g}. \end{aligned} \tag{5.12}$$

5.4 The Adjoint Variable

Now, multiplying by a vector co-state variable ψ and integrating over the domain, we have

$$\int_D \psi^T \left(\frac{\partial(\delta F)}{\partial \xi} + \frac{\partial(\delta G)}{\partial \eta}\right) d\xi d\eta = 0.$$

If $\boldsymbol{\psi}$ is differentiable, this may be integrated by parts to give

$$\int_D \left(\frac{\partial \boldsymbol{\psi}^T}{\partial \xi} \delta F + \frac{\partial \boldsymbol{\psi}^T}{\partial \eta} \delta G \right) d\xi d\eta = \oint_B \left(\bar{n}_\xi \boldsymbol{\psi}^T \delta F + \bar{n}_\eta \boldsymbol{\psi}^T \delta G \right) d\xi$$
$$+ \oint_C \left(n_\xi \boldsymbol{\psi}^T \delta F + \bar{n}_\eta \boldsymbol{\psi}^T \delta G \right) d\xi. \qquad (5.13)$$

Note that integrals along the cut line $(d\eta)$ have been cancelled since an O-type mesh and continuity of $\boldsymbol{\psi}$, δF, and δG have been assumed. Because the flow variables do not experience a jump along the cut line it was not necessary in the flow solution methodology for the Euler equations to decompose the solution into components, as was necessary for potential flow. Consequently, it will not be necessary to split the adjoint development for the Euler equations as was necessary for the potential flow adjoint equation.

Continuing with the derivation, after (5.13) is added to (5.9) the variation of the cost function may be written as

$$\delta I = \oint_C (p - p_d) \delta p \left(\frac{ds}{d\xi} \right) d\xi + \frac{1}{2} \oint_C (p - p_d)^2 \delta \left(\frac{ds}{d\xi} \right) d\xi$$
$$+ \int_D \left(\frac{\partial \boldsymbol{\psi}^T}{\partial \xi} \delta F + \frac{\partial \boldsymbol{\psi}^T}{\partial \eta} \delta G \right) d\xi d\eta$$
$$- \oint_B \left(n_\xi \boldsymbol{\psi}^T \delta F + \bar{n}_\eta \boldsymbol{\psi}^T \delta G \right) d\xi - \oint_C \left(\bar{n}_\xi \boldsymbol{\psi}^T \delta F + \bar{n}_\eta \boldsymbol{\psi}^T \delta G \right) d\xi \qquad (5.14)$$

where it is recalled that δF and δG may be defined by (5.12).

5.5 Boundary Conditions and the Adjoint Equation

It turns out that on the airfoil surface δG may be expanded in terms of δp directly instead of δw. Thus it follows from using equation (5.7) and also $\delta V = 0$ that

$$\delta G = J \begin{bmatrix} 0 \\ \frac{\partial \eta}{\partial x} \delta p \\ \frac{\partial \eta}{\partial y} \delta p \\ 0 \end{bmatrix} + p \begin{bmatrix} 0 \\ \delta \left(J \frac{\partial \eta}{\partial x} \right) \\ \delta \left(J \frac{\partial \eta}{\partial y} \right) \\ 0 \end{bmatrix} \qquad (5.15)$$

on the profile surface. At the outer boundary, incoming characteristics for ψ correspond to outgoing characteristics for $\delta \mathrm{w}$. Consequently, boundary conditions can be chosen for ψ such that

$$\bar{n}_\eta \psi^T C_2 \delta \mathrm{w} = 0. \tag{5.16}$$

Thus, by using the fact that $\bar{n}_\xi = 0$ and $\bar{n}_\eta = -1$ at the profile and $\bar{n}_\xi = 0$ and $\bar{n}_\eta = 1$ at the far field, combined with (5.15) and (5.16), equation (5.14) can be rewritten as

$$\begin{aligned}
\delta I = \quad & \oint_C (p - p_d)\delta p \left(\frac{ds}{d\xi}\right) d\xi + \frac{1}{2}\oint_C (p-p_d)^2 \delta\left(\frac{ds}{d\xi}\right) d\xi \\
+ \quad & \int_D \left(\frac{\partial \psi^T}{\partial \xi}\delta F + \frac{\partial \psi^T}{\partial \eta}\delta G\right) d\xi d\eta \\
- \quad & \oint_B \psi^T \left(\delta\left(J\frac{\partial \eta}{\partial x}\right)\mathbf{f} + \delta\left(J\frac{\partial \eta}{\partial y}\right)\mathbf{g}\right) d\xi \\
+ \quad & \oint_C \psi_2 \left(\left(J\frac{\partial \eta}{\partial x}\right)\delta p + p\delta\left(J\frac{\partial \eta}{\partial x}\right)\right) + \psi_3\left(\left(J\frac{\partial \eta}{\partial y}\right)\delta p + p\delta\left(J\frac{\partial \eta}{\partial y}\right)\right) d\xi .
\end{aligned} \tag{5.17}$$

Then if the coordinate transformation is such that $\delta\left(JK^{-1}\right)$ is negligible in the far field, the remaining integral over B also drops out.

Now suppose that ψ is the steady state solution of the adjoint equation

$$\frac{\partial \psi}{\partial t} - C_1^T \frac{\partial \psi}{\partial \xi} - C_2^T \frac{\partial \psi}{\partial \eta} = 0 \quad \text{in } D, \tag{5.18}$$

where the minus signs are a result of the reverse biasing of the equations. Then by letting ψ satisfy the boundary condition,

$$J\left(\psi_2 \frac{\partial \eta}{\partial x} + \psi_3 \frac{\partial \eta}{\partial y}\right) = -(p - p_d)\frac{ds}{d\xi} \quad \text{on } C, \tag{5.19}$$

equation (5.17) may be written as

$$\begin{aligned}
\delta I = \quad & \frac{1}{2}\oint_C (p-p_d)^2 \delta\left(\frac{ds}{d\xi}\right) d\xi \\
+ \quad & \int_D \left\{\frac{\partial \psi^T}{\partial \xi}\left(\delta\left(J\frac{\partial \xi}{\partial x}\right)\mathbf{f} + \delta\left(J\frac{\partial \xi}{\partial y}\right)\mathbf{g}\right) + \frac{\partial \psi^T}{\partial \eta}\left(\delta\left(J\frac{\partial \eta}{\partial x}\right)\mathbf{f} + \delta\left(J\frac{\partial \eta}{\partial y}\right)\mathbf{g}\right)\right\} d\xi d\eta \\
+ \quad & \oint_C \left\{\psi_2 \delta\left(J\frac{\partial \eta}{\partial x}\right) + \psi_3 \delta\left(J\frac{\partial \eta}{\partial y}\right)\right\} p \, d\xi
\end{aligned}$$

$$
\begin{aligned}
= \quad & \frac{1}{2}\oint_C (p-p_d)^2\,\delta\left(\frac{ds}{d\xi}\right) d\xi \\
+ \quad & \int_D \left\{ \frac{\partial \psi^T}{\partial \xi}\left(\delta\left(\frac{\partial y}{\partial \eta}\right)\mathbf{f} - \delta\left(\frac{\partial x}{\partial \eta}\right)\mathbf{g}\right) + \frac{\partial \psi^T}{\partial \eta}\left(-\delta\left(\frac{\partial y}{\partial \xi}\right)\mathbf{f} + \delta\left(\frac{\partial x}{\partial \xi}\right)\mathbf{g}\right)\right\} d\xi d\eta \\
+ \quad & \oint_C \left\{ \psi_2 \delta\left(-\frac{\partial y}{\partial \xi}\right) + \psi_3 \delta\left(\frac{\partial x}{\partial \xi}\right)\right\} p\, d\xi.
\end{aligned}
\tag{5.20}
$$

It is noted that the application of equation (5.19) for this system, unlike the corresponding equation (4.17) for the potential flow formulation, is not straightforward. Various aspects of the discretization of both (5.19) and (5.20) will be discussed in *Chapter 7*.

5.6 Algorithm Outline

Again the relatively simplified form of the final expression for the variation in the cost function (5.20) allows for its straightforward evaluation:

1. Solve the flow equation at the initial point according to *Chapter 2*.
2. Solve the adjoint equation

   $$\frac{\partial \psi}{\partial t} - C_1^T \frac{\partial \psi}{\partial \xi} - C_2^T \frac{\partial \psi}{\partial \eta} = 0 \quad \text{in } D,$$

   subject to

   $$J\left(\psi_2 \frac{\partial \eta}{\partial x} + \psi_3 \frac{\partial \eta}{\partial y}\right) = -(p-p_d)\frac{ds}{d\xi} \quad \text{on } C,$$

   ψ periodic along the cut line, and

   ψ satisfying matching conditions at the outer boundary defined by (5.16).

3. Solve for the $\delta(metric)$ terms in (5.20).
4. Calculate the gradient $\mathcal{G}$ or a specific variation δI using equation (5.20).

The method of solving the flow variables has already been discussed in *Chapter 2*. The evaluation of the $\delta(metric)$ terms as well as the specification of the design variables is addressed in *Chapter 6*. The details of the discretization and solution algorithm for the adjoint equation will be treated in *Chapter 7*.

5.7 DISCONTINUITIES

Here, as was the case for the potential flow equation, problems may arise when jumps appear at the boundary due to shock waves. Conveniently, techniques used to treat such problems can be quite similar for both equation sets. For the current Euler formulation, both the implicit smoothing (Section 4.11) and the shock wave unweighting (Section 4.12) can be used.

5.8 OTHER COST FUNCTIONS

The similarities between the potential flow formulation and the Euler equations also extend to the implementation of other cost functions. Probably the most important trait of using control theory is the easy way in which alternative or combined objective functions can be treated without incurring additional computational costs. Otherwise, if the goal of this work were simply to construct inverse design methods, the sophisticated development treated here could be completely avoided by using the more traditional inverse methods outlined in *Chapter 1*.

5.8.1 DRAG

To use drag as a cost function for the Euler formulation, much of the same development as in the case of drag for the potential flow formulation can be repeated. In fact Section (4.13.1) can be entirely duplicated from its beginning up through equation (4.28):

$$\begin{aligned}\delta I = \quad & \frac{1}{\frac{1}{2}\gamma M_\infty^2 c}\oint_C \delta p\left[\left(\frac{\partial y}{\partial \xi}\cos\alpha - \frac{\partial x}{\partial \xi}\sin\alpha\right) + \Omega\left(\frac{\partial x}{\partial \xi}\cos\alpha + \frac{\partial y}{\partial \xi}\sin\alpha\right)\right] d\xi \\ & + \frac{1}{\frac{1}{2}\gamma M_\infty^2 c}\oint_C p\left[\left(\delta\left(\frac{\partial y}{\partial \xi}\right)\cos\alpha - \delta\left(\frac{\partial x}{\partial \xi}\right)\sin\alpha\right) + \Omega\left(\delta\left(\frac{\partial x}{\partial \xi}\right)\cos\alpha + \delta\left(\frac{\partial y}{\partial \xi}\right)\sin\alpha\right)\right] d\xi\end{aligned}$$

(5.21)

where

$$\Omega = \frac{\left[C_l + \frac{\partial C_a}{\partial \alpha}\cos\alpha + \frac{\partial C_n}{\partial \alpha}\sin\alpha\right]}{\left[-C_d + \frac{\partial C_n}{\partial \alpha}\cos\alpha - \frac{\partial C_a}{\partial \alpha}\sin\alpha\right]}.$$

It is then possible to replace (5.20) by

$$\begin{aligned}\delta I = \quad & \frac{1}{\frac{1}{2}\gamma M_\infty^2 \bar{c}} \oint_C p \left[\left(\delta\left(\frac{\partial y}{\partial \xi}\right)\cos\alpha - \delta\left(\frac{\partial x}{\partial \xi}\right)\sin\alpha\right) + \Omega\left(\delta\left(\frac{\partial x}{\partial \xi}\right)\cos\alpha + \delta\left(\frac{\partial y}{\partial \xi}\right)\sin\alpha\right)\right] d\xi \\ + \quad & \int_D \left\{\frac{\partial \psi^T}{\partial \xi}\left(\delta\left(\frac{\partial y}{\partial \eta}\right)\mathbf{f} - \delta\left(\frac{\partial x}{\partial \eta}\right)\mathbf{g}\right) + \frac{\partial \psi^T}{\partial \eta}\left(-\delta\left(\frac{\partial y}{\partial \xi}\right)\mathbf{f} + \delta\left(\frac{\partial x}{\partial \xi}\right)\mathbf{g}\right)\right\} d\xi d\eta \\ + \quad & \oint_C \left\{\psi_2 \delta\left(-\frac{\partial y}{\partial \xi}\right) + \psi_3 \delta\left(\frac{\partial x}{\partial \xi}\right)\right\} p \, d\xi. \qquad (5.22)\end{aligned}$$

Implicit in (5.22) is the fact that the boundary condition at the surface of equations (5.19) is replaced by

$$\begin{aligned}& J\left(\psi_2 \frac{\partial \eta}{\partial x} + \psi_3 \frac{\partial \eta}{\partial y}\right) = \\ & \frac{1}{\frac{1}{2}\gamma M_\infty^2 \bar{c}} \frac{\partial}{\partial \xi}\left[\left(\frac{\partial y}{\partial \xi}\cos\alpha - \frac{\partial x}{\partial \xi}\sin\alpha\right) + \Omega\left(\frac{\partial x}{\partial \xi}\cos\alpha + \frac{\partial y}{\partial \xi}\sin\alpha\right)\right] d\xi. \qquad (5.23)\end{aligned}$$

5.8.2 LIFT OVER DRAG

A parameter that is often used in aerodynamic design is the lift to drag ratio. From basic aerodynamics, the maximization of lift/drag at a fixed M_∞ and specific fuel consumption maximizes cruise range. If the weight, and hence the C_l, is also constrained, the problem reduces to the one just explored in which drag is minimized at a given C_l. However, it may be of interest to obtain the maximum lift/drag for a given α. To maximize C_l/C_d, its inverse C_d/C_l is minimized. Thus, the cost function is defined as

$$I = C_d/C_l.$$

The variation becomes

$$\delta I \ = \ \frac{\delta C_d}{C_l} - \frac{C_d}{C_l^2}\delta C_l$$

$$
\begin{aligned}
= & \frac{\tilde{\delta} C_a \cos\alpha + \tilde{\delta} C_n \sin\alpha}{C_l} - \frac{C_d}{C_l^2}\left[\tilde{\delta} C_n \cos\alpha - \tilde{\delta} C_a \sin\alpha\right] \\
& + \left\{\frac{-C_a \sin\alpha + C_n \cos\alpha}{C_l} - \frac{C_d}{C_l^2}\left[-C_n \sin\alpha - C_a \cos\alpha\right]\right\}\delta\alpha \\
& + \left\{\frac{\frac{\partial C_a}{\partial\alpha}\cos\alpha + \frac{\partial C_n}{\partial\alpha}\sin\alpha}{C_l} - \frac{C_d}{C_l^2}\left[\frac{\partial C_n}{\partial\alpha}\cos\alpha - \frac{\partial C_a}{\partial\alpha}\sin\alpha\right]\right\}\delta\alpha.
\end{aligned}
$$

For the fixed α case, the last two terms with $\delta\alpha$ drop out, giving

$$
\begin{aligned}
\delta I = & \frac{1}{\frac{1}{2}\gamma M_\infty^2 \bar{c}}\oint_C \delta p\left[\frac{\frac{dy}{d\xi}\cos\alpha - \frac{dx}{d\xi}\sin\alpha}{C_l} + \frac{C_d}{C_l^2}\left(\frac{dx}{d\xi}\cos\alpha + \frac{dy}{d\xi}\sin\alpha\right)\right]d\xi \\
& + \frac{1}{\frac{1}{2}\gamma M_\infty^2 \bar{c}}\oint_C p\left[\frac{\delta\left(\frac{dy}{d\xi}\right)\cos\alpha - \delta\left(\frac{dx}{d\xi}\right)\sin\alpha}{C_l} + \frac{C_d}{C_l^2}\left(\delta\left(\frac{dx}{d\xi}\right)\cos\alpha + \delta\left(\frac{dy}{d\xi}\right)\sin\alpha\right)\right]d\xi.
\end{aligned}
$$

The boundary condition for the adjoint equation using this cost function becomes

$$
J\left(\psi_2\frac{\partial\eta}{\partial x} + \psi_3\frac{\partial\eta}{\partial y}\right) = -\frac{\frac{dy}{d\xi}\cos\alpha - \frac{dx}{d\xi}\sin\alpha + \frac{C_d}{C_l}\left(\frac{dx}{d\xi}\cos\alpha + \frac{dy}{d\xi}\sin\alpha\right)}{\frac{1}{2}\gamma M_\infty^2 \bar{c} C_l} \quad \text{on } C. \qquad (5.24)
$$

The final expression for the first variation in the cost function may be written as

$$
\begin{aligned}
\delta I = & \frac{1}{\frac{1}{2}\gamma M_\infty^2 \bar{c}}\oint_C p\left[\frac{\delta\left(\frac{dy}{d\xi}\right)\cos\alpha - \delta\left(\frac{dx}{d\xi}\right)\sin\alpha}{C_l} + \frac{C_d}{C_l^2}\left(\delta\left(\frac{dx}{d\xi}\right)\cos\alpha + \delta\left(\frac{dy}{d\xi}\right)\sin\alpha\right)\right]d\xi \\
& + \int_D\left\{\frac{\partial\psi^T}{\partial\xi}\left(\delta\left(\frac{\partial y}{\partial\eta}\right)\mathbf{f} - \delta\left(\frac{\partial x}{\partial\eta}\right)\mathbf{g}\right) + \frac{\partial\psi^T}{\partial\eta}\left(-\delta\left(\frac{\partial y}{\partial\xi}\right)\mathbf{f} + \delta\left(\frac{\partial x}{\partial\xi}\right)\mathbf{g}\right)\right\}d\xi d\eta \\
& + \oint_C\left\{\psi_2\delta\left(-\frac{\partial y}{\partial\xi}\right) + \psi_3\delta\left(\frac{\partial x}{\partial\xi}\right)\right\}p\, d\xi.
\end{aligned}
$$

(5.25)

Just as in the case of the potential flow formulation, any combination of the above cost functions can be weighted and summed to create a combined objective function. The possibility of prescribing field-based objectives such as minimizing entropy production also exists but is not explored here.

5.9 Second Order Design Methods Using the Euler Equations

As a final note, the development here to obtain δI via an adjoint method could be adjusted such that it would provide $\delta^2 I$ and hence the Hessian matrix. This was suggested in Section (3.7) of *Chapter 3* and is explored as a general development in *Appendix A1*. To give a greater understanding of how such a technique could be used in conjunction with the continuous sensitivity analysis method, *Appendix A2* develops a second order full Newton method for the case of the Euler equations. Although the method will not be implemented in this work, the basic approach is clear. Furthermore much of the hard work of building an adjoint solver for the Euler formulation that is contained in this thesis could in fact be reused in such an approach since the main element needed for the second order method is the solution of the same adjoint equation.

Chapter 6

GRID PERTURBATIONS AND DESIGN VARIABLES

This chapter contains two distinct parts. The first establishes a general grid perturbation method used to calculate efficiently the needed grid variational terms in both equation (4.19) and (5.20). The second section defines two possible families of design variables which are tested in this work.

6.1 GRID PERTURBATION METHOD

Equation (1.6) restated here,

$$\delta I = \frac{\partial I^T}{\partial w}\delta w + \frac{\partial I^T}{\partial \mathcal{X}}\delta \mathcal{X} + \frac{\partial I^T}{\partial \mathcal{F}}\delta \mathcal{F},$$

contains variations in the flow field variables (δw), variations in the grid point locations ($\delta \mathcal{X}$), and variations in the surface ($\delta \mathcal{F}$). *Chapters 4* and *5* developed techniques to eliminate (δw) from the variation in the cost function resulting in equations (4.19) and (5.20). These equations are the specific forms of the more general expression (1.12). Thus with the solution of the adjoint equations in hand, the only remaining variational terms are $\delta \mathcal{X}$ and $\delta \mathcal{F}$. It is helpful here to restate the final forms presented in *Chapters 4* and *5*:

For the potential flow formulation:

$$\delta I \quad = \quad +\frac{1}{2}\oint_C \left(q_d^2 - q^2\right)\delta\left(\frac{ds}{d\xi}\right)\, d\xi$$

$$+ \int_D \left(\tilde{Q} \frac{\partial \psi}{\partial \xi} + \tilde{P} \frac{\partial \psi}{\partial \eta} \right) d\xi \, d\eta. \tag{6.1}$$

For the Euler equations formulation:

$$\begin{aligned}
\delta I = \quad & \frac{1}{2} \oint_C (p - p_d)^2 \, \delta \left(\frac{ds}{d\xi} \right) d\xi \\
+ \quad & \oint_C \left\{ \psi_2 \delta \left(- \frac{\partial y}{\partial \xi} \right) + \psi_3 \delta \left(\frac{\partial x}{\partial \xi} \right) \right\} p \, d\xi \\
+ \quad & \int_D \left\{ \frac{\partial \psi^T}{\partial \xi} \left(\delta \left(\frac{\partial y}{\partial \eta} \right) \mathbf{f} - \delta \left(\frac{\partial x}{\partial \eta} \right) \mathbf{g} \right) + \frac{\partial \psi^T}{\partial \eta} \left(- \delta \left(\frac{\partial y}{\partial \xi} \right) \mathbf{f} + \delta \left(\frac{\partial x}{\partial \xi} \right) \mathbf{g} \right) \right\} d\xi d\eta.
\end{aligned} \tag{6.2}$$

In both cases the first terms are surface integrals which involve only variations in functions containing surface locations. These integrals are thus functions of $\delta \mathcal{F}$, which for a particular design variable can be calculated directly for insignificant computational costs. That is, the contribution to expression (1.12) from these surface integrals in both (6.1) and (6.2) can be evaluated by calculating only the $\delta \mathcal{F}$ terms by finite differences. No calculation of $\delta \mathcal{X}$ is necessary since it is not required. However, the last term in each case is a volume integral over the domain that also contains variations in grid point locations. These variations in the mesh metric terms unfortunately require complete knowledge of subsequent meshes.

One way to obtain these variations is by using an existing grid generation algorithm and applying finite differences to calculate the necessary information. While this approach would still obviate the use of multiple flow solutions to determine the gradient, it would require the mesh generator to be used for each design variable to form the finite difference approximations for the variations of the mesh metrics throughout the domain. Thus, the number of mesh generations required would be proportional to the number of design variables. And since flow solutions are typically much more computationally expensive than the grid generation, such an approach should ensure a significant savings over using finite differences for both grid generation and flow solutions. For three-dimensional design, however, where both the number of design variables and the computational cost of grid generation can be high, this approach would be excessively expensive. Further, for complicated three-dimensional configurations, where automatic grid generation still does not exist, such an approach becomes impractical.

This motivates the need to find a method which bypasses these difficulties. Here, the cost of repeated grid generations is removed from the gradient calculations by using a grid perturbation method. In this method, which was also used by Burgreen and Baysal [11], an initial structured curvilinear body-fitted grid is first created for the initial configuration by any grid generation process that may or may not be automated. Thus, the geometry as well as this initial grid become inputs to the optimization process. New grids, which conform to the surface as it is modified, are generated by shifting the grid points along each grid index line projecting from the surface. The points are shifted in real space by amounts that are attenuated proportional to the arc length away from the surface. If the outer boundary of the grid domain is held constant, the modification to the grid has the form

$$\begin{aligned} x^{new} &= x^{old} + \mathcal{R}\left(x_s^{new} - x_s^{old}\right) \\ y^{new} &= y^{old} + \mathcal{R}\left(y_s^{new} - y_s^{old}\right), \end{aligned} \tag{6.3}$$

where x and y represent the interior grid points, and x_s and y_s represent the surface grid points. $\mathcal{R}$ represents the arc length along the surface-projecting mesh line measured on the original grid, from the outer domain, and normalized so that $\mathcal{R} = 1$ at the inner surface.

The implications of using this type of grid perturbation scheme, that admits the possibility of crossed grid lines for large perturbations, are as follows: First, with respect to gradient calculations, analytic treatment of equations (6.3) permits the direct evaluation of grid variations ($\delta\mathcal{X}$) in terms of the surface variations ($\delta\mathcal{F}$). The result is that the volume integrals in both (6.1) and (6.2) may be reduced to surface integrals that depend only upon variations in $\mathcal{F}$. They may thus be treated inexpensively with finite differencing of $\mathcal{F}$ for each design variable. Second, since no regridding is required to calculate δI, the issue of possible crossed grid lines never arises for the gradient evaluations. Finally, with respect to line searches, where new grids must still be explicitly calculated, the method should only be employed when grid lines do not cross. If singularities begin to develop in the mesh, a secondary smoothing technique can be used to create new grids. In practice, the grid perturbation method has proven to be robust, and no failures due to singularities have occurred during optimization of typical convex geometries. To complete the analysis, the linear dependence of these perturbation equations must be extended to specific requirements for both the Euler and potential flow formulations.

6.2 Development of Grid Variations for the Potential Flow Equation

Examining equation (6.1) reveals that the dependence on the mesh variations is contained in the final volume integral through the terms $\tilde{Q}$ and $\tilde{P}$, which from *Chapter 4* are

$$\begin{aligned}
\tilde{Q}(\delta J, \delta A_{11}, \delta A_{12}, \delta A_{22}) &= \rho U \delta J \\
&+ \rho J \phi_\xi \left(1 - \frac{U\phi_\xi}{2c^2}\right) \delta A_{11} \\
&+ \rho J \phi_\eta \left(1 - \frac{U\phi_\xi}{c^2}\right) \delta A_{12} \\
&+ \rho J \phi_\eta \left(-\frac{U\phi_\eta}{2c^2}\right) \delta A_{22} \\
\tilde{P}(\delta J, \delta A_{11}, \delta A_{12}, \delta A_{22}) &= \rho V \delta J \\
&+ \rho J \phi_\eta \left(1 - \frac{V\phi_\eta}{2c^2}\right) \delta A_{22} \\
&+ \rho J \phi_\xi \left(1 - \frac{V\phi_\eta}{c^2}\right) \delta A_{12} \\
&+ \rho J \phi_\xi \left(-\frac{V\phi_\xi}{2c^2}\right) \delta A_{11}.
\end{aligned}$$

Thus, it is clear that the volume integral in equation (6.1) requires the variational terms δJ, δA_{11}, δA_{12} and δA_{22}. However, since we have an analytic expression for new grids in terms of an initial grid and surface perturbations, it is possible to construct these required variational terms such that the volume integral may be reduced to a surface integral. Consider first the term δJ. It is realized that equation (6.3) may be written in variational form as

$$\begin{aligned} \delta x(\xi,\eta) &= \mathcal{R}(\eta)\delta x_s(\xi) \\ \delta y(\xi,\eta) &= \mathcal{R}(\eta)\delta y_s(\xi) \end{aligned} \tag{6.4}$$

where δx_s and δy_s are defined on the airfoil surface. Thus we may write,

$$\begin{aligned} \delta\left(\frac{\partial x}{\partial \xi}\right) &= \mathcal{R}\,\delta\left(\frac{\partial x_s}{\partial \xi}\right) \\ \delta\left(\frac{\partial x}{\partial \eta}\right) &= \frac{\partial \mathcal{R}}{\partial \eta}\delta x_s \\ \delta\left(\frac{\partial y}{\partial \xi}\right) &= \mathcal{R}\,\delta\left(\frac{\partial y_s}{\partial \xi}\right) \\ \delta\left(\frac{\partial y}{\partial \eta}\right) &= \frac{\partial \mathcal{R}}{\partial \eta}\delta y_s. \end{aligned} \tag{6.5}$$

Recalling that J is defined as

$$J = \det|K| = \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi}$$

enables us to determine the first variation of the mesh Jacobian as

$$\delta J = \frac{\partial(\delta x)}{\partial \xi}\frac{\partial y}{\partial \eta} + \frac{\partial x}{\partial \xi}\frac{\partial(\delta y)}{\partial \eta} - \frac{\partial(\delta x)}{\partial \eta}\frac{\partial y}{\partial \xi} - \frac{\partial x}{\partial \eta}\frac{\partial(\delta y)}{\partial \xi}.$$

Substituting in equation (6.5) shows that

$$\begin{aligned} \delta J = \quad & \mathcal{R}\left[\frac{\partial(\delta x_s)}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial(\delta y_s)}{\partial \xi}\frac{\partial x}{\partial \eta}\right] \\ + \quad & \frac{\partial \mathcal{R}}{\partial \eta}\left[-\delta x_s\frac{\partial y}{\partial \xi} + \delta y_s\frac{\partial x}{\partial \xi}\right] \end{aligned}$$

or

$$\delta J = \left\{ \begin{array}{c} \frac{\partial(\delta x_s)}{\partial\xi}\left[\mathcal{R}\frac{\partial y}{\partial\eta}\right] \\ +\frac{\partial(\delta y_s)}{\partial\xi}\left[-\mathcal{R}\frac{\partial x}{\partial\eta}\right] \\ +\delta x_s\left[-\frac{\partial\mathcal{R}}{\partial\eta}\frac{\partial y}{\partial\xi}\right] \\ +\delta y_s\left[\frac{\partial\mathcal{R}}{\partial\eta}\frac{\partial x}{\partial\xi}\right] \end{array} \right\} = \left\{ \begin{array}{c} \frac{\partial(\delta x_s)}{\partial\xi}\bar{\mathbf{J}}_1 \\ +\frac{\partial(\delta y_s)}{\partial\xi}\bar{\mathbf{J}}_2 \\ +\delta x_s\bar{\mathbf{J}}_3 \\ +\delta y_s\bar{\mathbf{J}}_4 \end{array} \right\}. \tag{6.6}$$

The other necessary metric variations follow similarly. Recall that

$$\mathcal{A} = K^{-1}K^{T^{-1}} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix},$$

and

$$\begin{aligned} A_{11} &= \frac{\left(\frac{\partial y}{\partial\eta}\right)^2 + \left(\frac{\partial x}{\partial\eta}\right)^2}{J^2} \\ A_{12} &= \frac{-\left(\frac{\partial y}{\partial\xi}\frac{\partial y}{\partial\eta} + \frac{\partial x}{\partial\xi}\frac{\partial x}{\partial\eta}\right)}{J^2} \\ A_{22} &= \frac{\left(\frac{\partial y}{\partial\xi}\right)^2 + \left(\frac{\partial x}{\partial\xi}\right)^2}{J^2}. \end{aligned}$$

The first variations are,

$$\begin{aligned} \delta A_{11} &= -2\frac{\left[\left(\frac{\partial y}{\partial\eta}\right)^2 + \left(\frac{\partial x}{\partial\eta}\right)^2\right]}{J^3}\delta J + 2\frac{\left[\frac{\partial y}{\partial\eta}\delta\left(\frac{\partial y}{\partial\eta}\right) + \frac{\partial x}{\partial\eta}\delta\left(\frac{\partial x}{\partial\eta}\right)\right]}{J^2} \\ \delta A_{12} &= +2\frac{\left[\frac{\partial y}{\partial\xi}\frac{\partial y}{\partial\eta} + \frac{\partial x}{\partial\xi}\frac{\partial x}{\partial\eta}\right]}{J^3}\delta J - \frac{\left[\frac{\partial y}{\partial\xi}\delta\left(\frac{\partial y}{\partial\eta}\right) + \delta\left(\frac{\partial y}{\partial\xi}\right)\frac{\partial y}{\partial\eta} + \frac{\partial x}{\partial\xi}\delta\left(\frac{\partial x}{\partial\eta}\right) + \delta\left(\frac{\partial x}{\partial\xi}\right)\frac{\partial x}{\partial\eta}\right]}{J^2} \\ \delta A_{22} &= -2\frac{\left[\left(\frac{\partial y}{\partial\xi}\right)^2 + \left(\frac{\partial x}{\partial\xi}\right)^2\right]}{J^3}\delta J + 2\frac{\left[\frac{\partial y}{\partial\xi}\delta\left(\frac{\partial y}{\partial\xi}\right) + \frac{\partial x}{\partial\xi}\delta\left(\frac{\partial x}{\partial\xi}\right)\right]}{J^2}. \end{aligned}$$

Substituting in the expressions (6.5) gives,

$$\delta A_{11} = \left\{ \begin{array}{c} \frac{\partial(\delta x_s)}{\partial\xi}\left[\frac{-2A_{11}}{J}\frac{\partial y}{\partial\eta}\mathcal{R}\right] \\ +\frac{\partial(\delta y_s)}{\partial\xi}\left[\frac{+2A_{11}}{J}\frac{\partial x}{\partial\eta}\mathcal{R}\right] \\ +\delta x_s\left[\frac{+2A_{11}}{J}\frac{\partial y}{\partial\xi}\frac{\partial\mathcal{R}}{\partial\eta} + \frac{2}{J^2}\frac{\partial x}{\partial\eta}\frac{\partial\mathcal{R}}{\partial\eta}\right] \\ +\delta y_s\left[\frac{-2A_{11}}{J}\frac{\partial x}{\partial\xi}\frac{\partial\mathcal{R}}{\partial\eta} + \frac{2}{J^2}\frac{\partial y}{\partial\eta}\frac{\partial\mathcal{R}}{\partial\eta}\right] \end{array} \right\} = \left\{ \begin{array}{c} \frac{\partial(\delta x_s)}{\partial\xi}\bar{\mathbf{A}}_{11_1} \\ +\frac{\partial(\delta y_s)}{\partial\xi}\bar{\mathbf{A}}_{11_2} \\ +\delta x_s\bar{\mathbf{A}}_{11_3} \\ +\delta y_s\bar{\mathbf{A}}_{11_4} \end{array} \right\}$$

$$\delta A_{12} = \left\{ \begin{array}{c} \frac{\partial(\delta x_s)}{\partial \xi}\left[\frac{-2A_{12}}{J}\frac{\partial y}{\partial \eta}\mathcal{R} - \frac{\mathcal{R}}{J^2}\frac{\partial x}{\partial \eta}\right] \\ +\frac{\partial(\delta y_s)}{\partial \xi}\left[\frac{+2A_{12}}{J}\frac{\partial x}{\partial \eta}\mathcal{R} - \frac{\mathcal{R}}{J^2}\frac{\partial y}{\partial \eta}\right] \\ +\delta x_s\left[\frac{+2A_{12}}{J}\frac{\partial y}{\partial \xi}\frac{\partial \mathcal{R}}{\partial \eta} + \frac{1}{J^2}\frac{\partial x}{\partial \xi}\frac{\partial \mathcal{R}}{\partial \eta}\right] \\ +\delta y_s\left[\frac{-2A_{12}}{J}\frac{\partial x}{\partial \xi}\frac{\partial \mathcal{R}}{\partial \eta} + \frac{1}{J^2}\frac{\partial y}{\partial \xi}\frac{\partial \mathcal{R}}{\partial \eta}\right] \end{array} \right\} = \left\{ \begin{array}{c} \frac{\partial(\delta x_s)}{\partial \xi}\bar{\mathbf{A}}_{12_1} \\ +\frac{\partial(\delta y_s)}{\partial \xi}\bar{\mathbf{A}}_{12_2} \\ +\delta x_s\bar{\mathbf{A}}_{12_3} \\ +\delta y_s\bar{\mathbf{A}}_{12_4} \end{array} \right\}$$

$$\delta A_{22} = \left\{ \begin{array}{c} \frac{\partial(\delta x_s)}{\partial \xi}\left[\frac{-2A_{22}}{J}\frac{\partial y}{\partial \eta}\mathcal{R} + \frac{2\mathcal{R}}{J^2}\frac{\partial x}{\partial \xi}\right] \\ +\frac{\partial(\delta y_s)}{\partial \xi}\left[\frac{+2A_{22}}{J}\frac{\partial x}{\partial \eta}\mathcal{R} + \frac{2\mathcal{R}}{J^2}\frac{\partial y}{\partial \xi}\right] \\ +\delta x_s\left[\frac{+2A_{22}}{J}\frac{\partial y}{\partial \xi}\frac{\partial \mathcal{R}}{\partial \eta}\right] \\ +\delta y_s\left[\frac{-2A_{22}}{J}\frac{\partial x}{\partial \xi}\frac{\partial \mathcal{R}}{\partial \eta}\right] \end{array} \right\} = \left\{ \begin{array}{c} \frac{\partial(\delta x_s)}{\partial \xi}\bar{\mathbf{A}}_{22_1} \\ +\frac{\partial(\delta y_s)}{\partial \xi}\bar{\mathbf{A}}_{22_2} \\ +\delta x_s\bar{\mathbf{A}}_{22_3} \\ +\delta y_s\bar{\mathbf{A}}_{22_4} \end{array} \right\}, \tag{6.7}$$

where the $\bar{\mathbf{J}}$ and $\bar{\mathbf{A}}$ terms are introduced to ease the algebra. Thus, after the solution to the adjoint equation has been calculated, the contribution to the volume integral in the η direction can be evaluated without regard to any design variables. These variation-independent integrals have the form,

$$\begin{aligned} \mathcal{N}_{p_i} = \int \quad & \left\{\rho\left[U\bar{\mathbf{J}}_i + J\phi_\xi\left(1 - \frac{U\phi_\xi}{2c^2}\right)\bar{\mathbf{A}}_{11_i} + J\phi_\eta\left(1 - \frac{U\phi_\xi}{c^2}\right)\bar{\mathbf{A}}_{12_i} + J\phi_\eta\left(-\frac{U\phi_\eta}{2c^2}\right)\bar{\mathbf{A}}_{22_i}\right]\frac{\partial\psi}{\partial\xi}\right. \\ & \left. +\rho\left[V\bar{\mathbf{J}}_i + J\phi_\eta\left(1 - \frac{V\phi_\eta}{2c^2}\right)\bar{\mathbf{A}}_{22_i} + J\phi_\xi\left(1 - \frac{V\phi_\eta}{c^2}\right)\bar{\mathbf{A}}_{12_i} + J\phi_\xi\left(-\frac{V\phi_\xi}{2c^2}\right)\bar{\mathbf{A}}_{11_i}\right]\frac{\partial\psi}{\partial\eta}\right\} d\eta \end{aligned} \tag{6.8}$$

where $i = 1 : 4$. It follows that $\mathcal{N}_{p_i}$ is a function only of ξ, and the expression (6.1) for δI can be reduced to

$$\begin{aligned} \delta I &= \frac{1}{2}\oint_C \left(q_d^2 - q^2\right)\delta\left(\frac{ds}{d\xi}\right) d\xi \\ &+ \oint_C \left(\frac{\partial(\delta x_s)}{\partial\xi}\mathcal{N}_{p_1} + \frac{\partial(\delta y_s)}{\partial\xi}\mathcal{N}_{p_2} + \delta x_s\mathcal{N}_{p_3} + \delta y_s\mathcal{N}_{p_4}\right) d\xi. \end{aligned} \tag{6.9}$$

Equation (6.9) is in the form where only terms that are related to surface variations $(\delta\mathcal{F})$ are present, and the dependence on the variations of the field grid points $(\delta\mathcal{X})$

has been eliminated. This enables evaluation of equation (6.9) by finite differences for each design variable in turn at a very low cost.

6.3 Final Design Algorithm for Potential Flow Formulation

The complete design algorithm using the potential flow equation as the field constraint is outlined below:

1. Solve the potential flow equation according to Section (2.4).
2. Smooth the cost function (Section 4.11) or unweight the shocks (Section 4.12) if necessary.
3. Solve the adjoint equation (4.16) subject to boundary conditions such as (4.17), (4.22), (4.31) or (4.32).
4. Solve for $\mathcal{N}_{p_i}$ by equation (6.8).
5. Determine the gradient:
   - Perturb each design variable and evaluate equation (6.9).
   - Construct each component of the gradient by dividing by the perturbation of the corresponding design variable.
   - Assemble the gradient vector.
6. Feed the gradient vector $\mathcal{G}$ to the quasi-Newton algorithm (3.21) to calculate the search direction s.
7. Perform a line search in s according to Section (3.6) and estimate the minimum by repeated flow solver evaluations.
8. Return to (1).

The only undefined portion of the above algorithm is the choice of design variables, which is treated in Section (6.6). Since the procedure employed in this research is a continuous sensitivity approach, the details of the discretization of the differential adjoint equations as well as their solution procedure must be defined. *Chapter 7* treats these important aspects.

6.4 Development of Grid Variations for the Euler Equations

Just as in the case of the potential flow formulation, the final area integral in equation (6.2) is a function of the variation of the mesh metrics. To avoid recomputing the entire mesh for each design variable to obtain these metrics, the analytic mesh perturbation equations (6.3) may again be exploited. It turns out that the necessary variation terms for the Euler equations given in (6.2) are considerably simpler than those for the potential flow equation, and are given by (6.6) and (6.7).

It is convenient to rewrite (6.2) after integrating by parts as

$$\begin{aligned} \delta I = & \frac{1}{2}\oint_C (p-p_d)^2 \delta\left(\frac{ds}{d\xi}\right) d\xi \\ & - \oint_C \psi^T \left\{-\delta\left(\frac{\partial y}{\partial \xi}\right)\bar{\mathbf{f}} + \delta\left(\frac{\partial x}{\partial \xi}\right)\bar{\mathbf{g}}\right\} d\xi \\ & - \int_D \psi^T \frac{\partial}{\partial \xi}\left(\delta\left(\frac{\partial y}{\partial \eta}\right)\mathbf{f} - \delta\left(\frac{\partial x}{\partial \eta}\right)\mathbf{g}\right) \\ & + \psi^T \frac{\partial}{\partial \eta}\left(\delta\left(-\frac{\partial y}{\partial \xi}\right)\mathbf{f} + \delta\left(\frac{\partial x}{\partial \xi}\right)\mathbf{g}\right) d\xi d\eta \end{aligned} \tag{6.10}$$

where $\bar{\mathbf{f}}$ and $\bar{\mathbf{g}}$ are again the flux components f and g with the pressure terms deleted from the momentum equation. And since the flux is set to zero across the airfoil surface, this second integral drops out resulting in the expression

$$\begin{aligned} \delta I = & \frac{1}{2}\oint_C (p-p_d)^2 \delta\left(\frac{ds}{d\xi}\right) d\xi \\ & - \int_D \psi^T \frac{\partial}{\partial \xi}\left(\delta\left(\frac{\partial y}{\partial \eta}\right)\mathbf{f} - \delta\left(\frac{\partial x}{\partial \eta}\right)\mathbf{g}\right) \\ & + \psi^T \frac{\partial}{\partial \eta}\left(\delta\left(-\frac{\partial y}{\partial \xi}\right)\mathbf{f} + \delta\left(\frac{\partial x}{\partial \xi}\right)\mathbf{g}\right) d\xi d\eta. \end{aligned} \tag{6.11}$$

Fortunately the necessary metric variations in (6.11) have already been developed for the mesh perturbation algorithm defined in Section (6.1). They were needed in an intermediate step for the development of the necessary metric variations used in the potential flow formulation and were given by equation (6.5). Again following the approach used for the potential flow formulation, we may substitute these expressions into (6.10) and integrate the second term along the index direction projecting from the configuration surface without any dependence on particular design variables since the metric variations are fully determined by the surface perturbations. Thus, the expression for the variation in the cost function can be reduced to surface integrals only:

$$\begin{aligned}\delta I = & \quad \frac{1}{2}\oint_C (p-p_d)^2\,\delta\left(\frac{ds}{d\xi}\right)d\xi \\ & - \oint_C \left\{\delta y_s \mathcal{N}_{e_1} + \delta x_s \mathcal{N}_{e_2} + \delta\left(\frac{\partial y}{\partial \xi}\right)_s \mathcal{N}_{e_3} + \delta\left(\frac{\partial x}{\partial \xi}\right)_s \mathcal{N}_{e_4}\right\} d\xi \qquad (6.12)\end{aligned}$$

where

$$\begin{aligned}\mathcal{N}_{e_1} &= \int \psi^T \frac{\partial \mathcal{R}}{\partial \eta}\frac{\partial \mathbf{f}}{\partial \xi}\,d\eta \\ \mathcal{N}_{e_2} &= \int -\psi^T \frac{\partial \mathcal{R}}{\partial \eta}\frac{\partial \mathbf{g}}{\partial \xi}\,d\eta \\ \mathcal{N}_{e_3} &= \int -\psi^T \mathcal{R}\frac{\partial \mathbf{f}}{\partial \eta}\,d\eta \\ \mathcal{N}_{e_4} &= \int \psi^T \mathcal{R}\frac{\partial \mathbf{g}}{\partial \eta}\,d\eta. \qquad (6.13)\end{aligned}$$

Equation (6.12) has reduced the expression for δI into line integrals along the surface where the only remaining unknowns are the surface variationals. As with the potential flow method, these surface variationals may be easily determined for any modification in the surface using finite differences.

6.5 Final Design Algorithm Using the Euler Formulation

The complete design algorithm using the Euler equations as the field constraint is outlined bellow:

1. Solve the Euler equations according to Section (2.5).
2. Smooth the cost function (Section 4.11) or unweight the shocks (Section 4.12) if necessary.

3. Solve the adjoint equation (5.18) subject to boundary conditions such as (5.19), (5.23), or (5.24).

4. Solve for $\mathcal{N}_{c_i}$ by equation (6.13).

5. Determine the gradient:

   - Perturb each design variable and evaluate equation (6.9).
   - Construct each component of the gradient by dividing by the perturbation of the corresponding design variable.
   - Assemble the gradient vector.

6. Feed the gradient vector $\mathcal{G}$ to the quasi-Newton algorithm (3.21) to calculate the search direction s.

7. Perform a line search in s according to Section (3.6) and determine the minimum by repeated flow solver evaluations.

8. Return to (1).

Thus the complete design algorithm has been defined for both formulations. Each design method has been defined as a general procedure from the continuous sensitivity stand point. Various cost functions or combinations of cost functions can be treated within this context. The fact that a continuous sensitivity approach is used forces a detailed description of the discretization and solution strategy for the adjoint equations. This description is addressed in *Chapter 7*. The remainder of this chapter will be dedicated to the open issue remaining in the above list—the choice of design variables.

6.6 Design Variables

The advent of new adjoint-based design procedures, such as the ones explored here, will force a reconsideration of other aspects of the aerodynamic design process that heretofore have been quite entrenched. The conclusion of this work (*Chapter 10*) will try to illuminate some of these necessary reconsiderations. However, one aspect which cannot be left solely to future examinations, and without which this research would be incomplete, is an exploration of possible design space parameterizations.

6.6.1 MESH POINTS AS DESIGN VARIABLES

In Jameson's first works on the use of control theory [54, 55, 59], every surface mesh point was used as a design variable. For three-dimensional wing design cases this led to as many as 4,224 design variables [59]. The use of the adjoint method eliminated the unacceptable costs that such a large number of design variables would incur for traditional finite difference methods. Theoretically, this choice would give the ultimate freedom in possible design shapes permitted by the resolution of the mesh. The approach does present some difficulties. First, even though the gradient is well defined by this choice, it is quite probable that it will not be smooth or may even have discontinuities. The problem is best exemplified by using the case where only one grid point is used as a design variable, with the rest of the grid points being fixed. Not only will the predicted change from the adjoint-based gradient have little chance of matching the actual change for any except infinitesimal variations, but the flow solution process may itself become ill-conditioned. Motion of a single point violates the C^2 continuity of the surface that is usually necessary in order to obtain smooth solutions. The resulting flow solutions from single point motion, if even attainable, would have at least slope discontinuities. This contradicts the development of the adjoint formulations, where continuity in the 1st derivative of the solution was assumed throughout.

At the very least, mesh point-based design variables create a situation where the solution of the adjoint equations only provides a gradient that is compatible with infinitesimal changes. The process is analogous to that of using a simple explicit time marching solution to solve a system of coupled equations. While the predicted corrections may be accurate for infinitesimal time steps, these methods have a finite stability region. The simple forward Euler scheme is unstable, for example, for purely hyperbolic systems. Jameson, who introduced the use of continuous sensitivity analysis for transonic flows, realized this even in his first works [55, 58] where the surface mesh points were used as design variables. Following his own developments to stabilize the solution process of explicit Runge-Kutta type algorithms by implicit residual smoothing, he used a similar approach to smooth the gradient. After the procedure, the smoothed gradient defines a new search direction in which a significant step can be taken without developing problems in the flow solution algorithm since all resulting shapes developed along this direction are insured to be smooth. He proved that the smoothing algorithm employed for the gradient still ensures that an improvement must exist in the new search direction. While the method has been successfully demonstrated for a wide variety of design problems, it is sensitive to the level and character of the smoothing used.

Another way to understand the problem associated with using each grid point as a design variable is that this design parameterization admits very high as well as low frequencies into the shape of interest. This admittance of high frequencies into the design space will in turn allow it to be characterized by a higher degree of nonlinearity (i.e., valleys and canyons with steeper walls and more of them). In theory, a global optimum in this high frequency design space could still be found. This minimum would then necessarily have a lower value of the objective function than a design space for the same problem that was resolved with only lower frequencies. Unfortunately the degree of nonlinearity that would be admitted into the design space is such that finding the minimum is likely to become computationally difficult, with the use of simple gradient methods resulting in outright failure. Jameson's way out of this difficulty was to smooth the gradient and hence the local design space by using in effect a low-pass filter.

If the use of mesh point design variables were extended to treat complete aircraft configurations, at least tens of thousands of design variables would be necessary. Extravagant numbers of design variables preclude the use of descent algorithms such as Newton or standard quasi-Newton approaches simply because of the high cost of the associated matrix operations. The use of a simple descent procedure, such as steepest descent, has the advantage that significant errors can be tolerated initially in the gradient evaluation.

Mesh point-based methods therefore favor tighter coupling of the flow solver, the adjoint solver, and the design problem to accelerate convergence. Ta'asan et al. [71] have taken advantage of this by formulating the design problem as a "one-shot" procedure where all three systems are advanced simultaneously, through a multigrid algorithm. Essentially, each level in the multigridding of both the flow solution and the adjoint solution develops different solutions and hence different gradient information. Ta'asan et al. have attempted to capitalize on this property by relaxing the gradient information developed in each step of each multigrid level through a simplified optimization step in these transient gradient directions. In Jameson's original works, the numbers of complete multigrid steps for the flow solver and the adjoint solver are adjusted to give optimum results. In the limiting case where only one step is taken, the scheme is very similar to that adopted by Ta'asan.

6.6.2 HICKS-HENNE FUNCTIONS AS DESIGN VARIABLES

An alternative choice of design space that leads to quite different conclusions for the optimization algorithm was proposed by Hicks and Henne [39, 37]. In their parameterization of the design space, a set of smooth functions that perturb the initial geometry are defined. The main result is that far fewer design variables are needed to provide for an adequately open design space. Hicks initially adopted the approach because his work in the past exclusively used finite difference-based gradient design methods that are inherently restricted to a few dozen design variables. As a consequence of the fact that these early methods had to content themselves with at most a hundred design variables, they retained considerable freedom in the choice of the descent algorithm. One choice of design variables for airfoils suggested by Hicks and Henne [38] has the following "sine bump" form:

$$b(x) = \left[\sin\left(\pi x^{\frac{\log(.5)}{\log(t_1)}}\right)\right]^{t_2}, \quad 0 \le x \le 1$$

Here t_1 locates the maximum of the bump in the range $0 \le x \le 1$ at $x = t_1$, since the maximum occurs when $x^{\hat{\alpha}} = \frac{1}{2}$, where $\hat{\alpha} = \log\frac{1}{2}/\log t_1$, or $\hat{\alpha}\log t_1 = \log\frac{1}{2}$. Parameter t_2 controls the width of the bump.

When distributed over the entire chord on both upper and lower surfaces, these analytic perturbation functions admit a large possible design space. They can be chosen such that symmetry, thickness, or volume can be explicitly constrained, thus avoiding the use of constrained optimization algorithms to address geometric constraints. Further, particular choices of these variables can concentrate the design effort in regions where refinement is needed, while leaving the rest of the airfoil section virtually undisturbed. The disadvantage of these functions is that they are not orthogonal, and there is no simple way to form a basis from these functions which is complete for the space of continuous functions that vanish at $x = 0$ and $x = 1$. Thus, they do not guarantee that a solution, for example, of the inverse problem for a realizable target pressure distribution will necessarily be attained. Nevertheless, they have proved to be effective in realizing design improvements when only a limited number of design variables can be admitted. A design process using these basis spaces can be accelerated toward convergence either by tighter coupling of the individual design elements, as was the case when using the mesh points themselves, or through the use of higher order optimization algorithms. The Hicks-Henne functions also have one last advantage over using the mesh points in that there is no need to smooth the resulting solutions as the design proceeds, since by construction higher frequencies are not admitted and thus the design spaces are naturally well posed.

6.6.3 B-SPLINE CONTROL POINTS AS DESIGN VARIABLES

Another choice of design variables that has gained favor is the use of B-spline control points, most recently through the work of Baysal and Burgreen [11], and others [108]. Instead of applying perturbation functions to existing airfoils, this method starts with an initial airfoil which is analytically defined as a B-spline curve. The design is accomplished by directly modifying the analytic curve via its control points.

To understand this it is helpful to outline B-splines, or rather Bézier curves which form their underlying basis. Consider a set of control points $\bar{b}_0^0$, $\bar{b}_1^0$, $\bar{b}_2^0$, ..., $\bar{b}_i^0$, ..., $\bar{b}_m^0$, representing 2-space coordinates. A Bézier curve can be constructed through de

Casteljau's algorithm:

$$\bar{\mathbf{b}}_i^j(\bar{t}) = (1-\bar{t})\bar{\mathbf{b}}_i^{j-1}(\bar{t}) + \bar{t}\bar{\mathbf{b}}_{i+1}^{j-1}(\bar{t}) \qquad \begin{cases} j = 1, \ldots, m \\ i = 0, \ldots, m-j \end{cases}$$

where m is the degree of the polynomial. For the simple quadratic case with only three control points ($\bar{\mathbf{b}}_0^0$, $\bar{\mathbf{b}}_1^0$, $\bar{\mathbf{b}}_2^0$), the algorithm results in:

$$\begin{aligned} \bar{\mathbf{b}}_0^1(\bar{t}) &= (1-\bar{t})\bar{\mathbf{b}}_0^0(\bar{t}) + \bar{t}\bar{\mathbf{b}}_1^0(\bar{t}) \\ \bar{\mathbf{b}}_1^1(\bar{t}) &= (1-\bar{t})\bar{\mathbf{b}}_1^0(\bar{t}) + \bar{t}\bar{\mathbf{b}}_2^0(\bar{t}) \\ \\ \bar{\mathbf{b}}_0^2(\bar{t}) &= (1-\bar{t})\bar{\mathbf{b}}_0^1(\bar{t}) + \bar{t}\bar{\mathbf{b}}_1^1(\bar{t}). \qquad (6.14) \end{aligned}$$

Thus, (6.14) defines a curve which spans 2-space between $\bar{\mathbf{b}}_0^0$ and $\bar{\mathbf{b}}_2^0$ through the

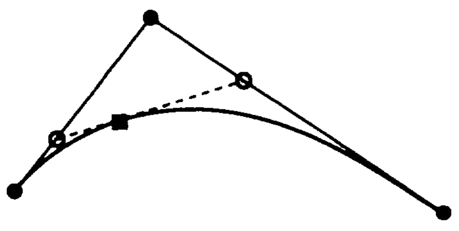


Figure 6.1: *Bézier Curve of Degree 2*

variation of the non-dimensional parameter $\bar{t}$. Figure (6.1) illustrates such a curve and its geometric construction. The degree m of the Bézier curve is one less than the number of control points. As with any polynomial representation, higher orders eventually become computationally unmanageable, with 10 being a rational limit. Unfortunately, a reasonable number of control points (20+) is required to give an airfoil the needed geometric freedom. To overcome this difficulty, B-splines curves may be constructed as piecewise Bézier curves. These have the advantage of limiting the degree of the polynomial to a manageable user-defined level, while still maintaining the analytic representation for the surface shape. A complete development and treatment of Bézier curves and B-spline curves is beyond the scope of this research but can be found in reference [20]. The important aspects concerning the use of B-spline control points as design variables for aerodynamic design can be explored without a detailed presentation.

Like the Hicks-Henne functions, B-splines allow for a reduced number of design variables, and thus permit the use of, say, a quasi-Newton design procedure. If the upper and lower surfaces of an airfoil are separated, the method easily admits camber or thickness constraints explicitly within the design space. To explain this it should be noted that the polygon defined by the B-spline control points[1] forms a general outline of the airfoil defined by the curve. Further, as more control points are included in the definition of the curve, the closer this correlation becomes between the control point polygon and the curve it defines. Thus for an airfoil defined by many control points, the vertical distance between corresponding control points on the upper and lower surfaces roughly defines the thickness distribution. Local control is also possible by choosing only a limited number of control points as active design variables. This method in practice may have an advantage over the Hicks-Henne functions in that a more complete basis space of admitted airfoils is permitted with the same number of design variables. Finally, since these curves and surfaces are now the natural entities used in most CAD environments, they provide a straightforward way of integrating CAD and aerodynamic design.

One possible drawback of B-spline-based design variables has already been hinted at. The fact that B-spline curves are nothing more than a patched set of polynomials

[1]The B-spline polygon is defined by sequentially connecting the B-spline control points.

means that the degree of these polynomials may dramatically affect their performance. For applications to airfoils, it seems necessary to use B-splines of at least degree 3 which ensures C^2 continuity of the resulting curve. It is perhaps desirable to use even higher polynomials, since the primary aerodynamic state variable of interest, pressure, in subsonic flow is a general function of the curvature (C^2) of the airfoil shape. Smooth pressure distributions thus require smoothness in the curvature of the airfoil, or C^3 continuity. This leads us to another possible problem with these design variables—namely that since piecewise polynomials of at least degree 3 are being used with a fair density of control points so as to capture a large possible family of airfoils, surface shapes containing high frequency modes are again possible, just as in the case of grid point design variables.

The results to be presented in *Chapters 8* and *9* will explore the use of both Hicks-Henne perturbation functions and B-spline control points. This is by no means a complete treatment of design variables, but gives an introduction to how important the problem of parameterizing the design space will become in the future as adjoint-based approaches decouple the cost of design from the number of design variables used.

Chapter 7

Adjoint Discretization and Solution Procedures

The mathematical development of a control theory approach to airfoil design is now complete for both a potential flow and an Euler formulation. However, since this research employs continuous sensitivity analysis as opposed to discrete sensitivity analysis, details of the discretization for the two different adjoint systems must also be outlined. Also, regardless of whether a continuous or a discrete implementation of control theory is applied, the solution methodology for the large scale adjoint systems must also be discussed. As stated in the introduction of this research, one important advantage of the continuous sensitivity approach is the ease of recycling the flow solution methodology for the solution of the corresponding co-state system. In this chapter, the details of the discretization and solution strategy for both co-state systems will be described.

7.1 Discretization of the Potential Flow Adjoint Equation

For the potential flow equation, Section (2.4) describes the discretization of the domain equations and their associated boundary conditions. Recall that the potential flow equation in differential form was given by

$$\frac{\partial}{\partial \xi}(\rho J U) + \frac{\partial}{\partial \eta}(\rho J V) = 0 \quad \text{in } D, \tag{7.1}$$

and that its discrete counterpart including artificial dissipation was

$$\bar{\delta}_\xi \left(\bar{\nu}_\eta (\rho J U) + P\right) + \bar{\delta}_\eta \left(\bar{\nu}_\xi (\rho J V) + Q\right) - \epsilon \bar{\delta}_{\xi\eta} \left[\left(\bar{A}^{(\xi)} + \bar{A}^{(\eta)} \right) \delta_{\xi\eta} \phi \right] = 0. \quad (7.2)$$

The potential flow adjoint equation in differential form developed in *Chapter 5* is given by

$$\begin{aligned} & \frac{\partial}{\partial \xi} \left[\rho J \left(A_{11} - \frac{U^2}{c^2} \right) \frac{\partial \psi}{\partial \xi} + \rho J \left(A_{12} - \frac{UV}{c^2} \right) \frac{\partial \psi}{\partial \eta} \right] \\ + & \frac{\partial}{\partial \eta} \left[\rho J \left(A_{12} - \frac{UV}{c^2} \right) \frac{\partial \psi}{\partial \xi} + \rho J \left(A_{22} - \frac{V^2}{c^2} \right) \frac{\partial \psi}{\partial \eta} \right] = 0. \end{aligned} \quad (7.3)$$

The object now is to construct a discrete scheme for (7.3) that is consistent. While equation (7.3) seems more complicated than (7.1), it is simply an expansion of (7.1) due to the linearization. Referring back to the original stencil depicted in Figures (2.1) and (2.2), ϕ was assumed to be defined at the mesh points while ρ, J, U, and V were calculated at the cell centers. Equation (7.2) was then constructed by using the secondary control volume shown in Figure (2.2). To maintain this basic structure, the following discrete scheme is used:

$$\begin{aligned} & \bar{\delta}_\xi \left[\bar{\nu}_\xi \left(\rho J \left(A_{11} - \frac{U^2}{c^2} \right) \right) \bar{\delta}_\xi (\psi) + \bar{\nu}_\eta \left(\rho J \left(A_{12} - \frac{UV}{c^2} \right) \right) \delta_\eta (\psi) \right] \\ + \; & \bar{\delta}_\eta \left[\bar{\nu}_\xi \left(\rho J \left(A_{12} - \frac{UV}{c^2} \right) \right) \bar{\delta}_\xi (\psi) + \bar{\nu}_\eta \left(\rho J \left(A_{22} - \frac{V^2}{c^2} \right) \right) \delta_\eta (\psi) \right] = 0. \end{aligned} \quad (7.4)$$

This implies that ψ, like ϕ, is defined at the mesh points and the terms $\rho J \left(A_{11} - \frac{U^2}{c^2} \right)$, $\rho J \left(A_{12} - \frac{UV}{c^2} \right)$, and $\rho J \left(A_{22} - \frac{V^2}{c^2} \right)$ are calculated at the edge centers. Note that no artificial viscosity or difference rotation terms have thus far been added to the scheme.

7.2 Dissipation Used for the Potential Flow Adjoint Equation

As it turns out the choice of the discretization in (7.4) is a simple finite difference stencil for (7.3). Most notably the average operators in (7.4) do not encompass the differences in ψ. This is different from the stencil used for the flow equations, where motivated by the desire to preserve a conservative scheme, the averages essentially operated on the differences in ϕ. This switch in stencil type from the one used in the potential flow equations has the consequence that no difference rotation terms need be added to prevent odd-even decoupling in the solution since, ignoring cross derivatives, (7.4) is already in the form of a standard five point scheme. However, artificial dissipation of the form used in the flow equation is needed. For the flow equations, the artificial dissipation terms were given by equations (2.20). These may be duplicated for the solution of the adjoint equation with a change in the direction of upwind biasing to reflect the reverse flow character of the adjoint system. Thus,

$$\begin{aligned}\hat{P}_{\psi} &= \hat{\mu}\frac{\rho J}{c^2}\left(U^2\psi_{\xi\xi} + UV\psi_{\xi\eta}\right)\\ \hat{Q}_{\psi} &= \hat{\mu}\frac{\rho J}{c^2}\left(UV\psi_{\xi\eta} + V^2\psi_{\eta\eta}\right),\end{aligned}$$

and P_{ψ} and Q_{ψ} can be calculated at the cell edges by

$$P_{\psi_{i+\frac{1}{2},j}} = \begin{cases} -\hat{P}_{\psi_{i+1,j}} & \text{if } U > 0 \\ \hat{P}_{\psi_{i,j}} & \text{if } U < 0 \end{cases}$$

$$Q_{\psi_{i,j+\frac{1}{2}}} = \begin{cases} -\hat{Q}_{\psi_{i,j+1}} & \text{if } V > 0 \\ \hat{Q}_{\psi_{i,j}} & \text{if } V < 0. \end{cases} \tag{7.5}$$

This would also closely approximate what would happen in a discrete sensitivity analysis method, where the switch in the biasing is contained in the transpose operator on the flux Jacobian. The entire discrete scheme for the adjoint equation may now be written as

$$\begin{aligned} & \bar{\delta}_{\xi}\left[\bar{\nu}_{\xi}\left(\rho J\left(A_{11} - \frac{U^2}{c^2}\right)\right)\bar{\delta}_{\xi}(\psi) + \bar{\nu}_{\eta}\left(\rho J\left(A_{12} - \frac{UV}{c^2}\right)\right)\bar{\delta}_{\eta}(\psi)\right] \\ + \; & \bar{\delta}_{\eta}\left[\bar{\nu}_{\xi}\left(\rho J\left(A_{12} - \frac{UV}{c^2}\right)\right)\bar{\delta}_{\xi}(\psi) + \bar{\nu}_{\eta}\left(\rho J\left(A_{22} - \frac{V^2}{c^2}\right)\right)\bar{\delta}_{\eta}(\psi)\right] \\ + \; & \bar{\delta}_{\xi}\left(P_{\psi}\right) + \bar{\delta}_{\eta}\left(Q_{\psi}\right) = 0. \end{aligned} \tag{7.6}$$

7.3 Iteration Scheme for the Potential Flow Adjoint Equation

With (7.6) defined, a convergent iteration scheme must be constructed. It is possible to create this iteration scheme for the adjoint solver by using the flow solver iteration scheme as a template. Equation (2.24) rewritten here in terms of the adjoint variable is used as the basic solution procedure:

$$\left(\mathcal{Z} - \ddot{\delta}_\xi \left(\bar{\nu}\left(\bar{A}^{(\xi)}\right)\bar{\delta}_\xi\right) - \bar{\delta}_{\hat{\xi}}\left(\bar{P}\bar{\delta}_\xi^2\right)\right)\left(\mathcal{Z} - \bar{\delta}_\eta\left(\nu\left(\bar{A}^{(\eta)}\right)\bar{\delta}_\eta\right) - \ddot{\delta}_{\hat{\eta}}\left(\bar{Q}\bar{\delta}_\eta^2\right)\right)\hat{\delta}\psi^{n+1} = -\omega\mathcal{Z}\mathcal{L}\psi^n. \tag{7.7}$$

Fortunately, this generalized ADI scheme can be reused for the solution of the adjoint equation with only minor modifications. In order to remain consistent with the reverse flow nature of the adjoint system, the ADI scheme is swept in two parts from the trailing edge forward in the ξ direction. Further, the η sweep is run from the airfoil surface out toward the outer boundary. This is exactly opposite to the directions used for the flow equations.[1] As previously stated, the one-sided differences incorporated into $\mathcal{Z}$ are forced to have upwind biasing consistent with the sweep direction.[2] This choice of the difference stencils and the sweep directions favorably influences the convergence of the iteration algorithm since consistency is maintained with the choice of the biasing of P_ψ and Q_ψ in the supersonic regions of the flow. The boundary conditions specified by (4.17), (4.22), (4.31) and (4.32) are treated by standard finite differencing. It will be noted that this simplified treatment for the wall boundary condition of the potential flow adjoint equation will not carry over to the case of the Euler adjoint equation. The treatment of the remaining adjoint boundary conditions at the cut and far field are straightforward since they are periodic and explicitly zero respectively.

[1]See *Chapter 2*

[2]See section 2.4.4.

7.4 Convergence Acceleration of the Potential Flow Adjoint Equation

Just as in the case of the flow solver, the ADI iteration scheme for the adjoint equation displays only marginally acceptable convergence performance. To enhance the performance of the scheme, the same multigrid acceleration algorithm used for the flow equation is recycled. Section (2.4.5) described how the multigrid algorithm was applied to the flow solver. No changes are required for its reuse in the solution of the co-state system. In fact in the actual implementation, both the flow solver and the adjoint solver share common subroutines for the treatment of all the necessary multigrid operations.

7.5 Discretization of the Euler Adjoint Equation

The discretization of the potential flow adjoint equation was done for both the domain and boundary conditions without regard to how consistent this discretization was with that obtained by discrete sensitivity analysis. And as the solutions in *Chapter 8* will illustrate, this implementation results in acceptable results. Unfortunately, these consistency issues prove more demanding of attention when the Euler equations are considered. This section will illustrate the differences that can exist between various discretizations for both the adjoint domain equations and their associated boundary conditions.

The conservative discretization used for the solution of the Euler equations was presented in Section (2.5). Recall from equation (2.29) that the integral form of these equations may be written as

$$\frac{d}{dt}\int_{\mathcal{V}} \mathbf{w}\, d\mathcal{V} + \int_{\mathcal{S}} \mathbf{F}\cdot\mathbf{n}\, d\mathcal{S} = 0. \tag{7.8}$$

The semi-discrete equivalent was written as

$$\begin{aligned}\frac{dW_{i,j}}{dt} = \quad & -\left\{\left(\frac{1}{2}F^{+}_{i+1,j} + \frac{1}{2}F^{+}_{i,j}\right) - \left(\frac{1}{2}F^{-}_{i,j} + \frac{1}{2}F^{-}_{i-1,j}\right)\right.\\ & \left. + \left(\frac{1}{2}G^{+}_{i,j+1} + \frac{1}{2}G^{+}_{i,j}\right) - \left(\frac{1}{2}G^{-}_{i,j} + \frac{1}{2}G^{-}_{i,j-1}\right)\right\}.\end{aligned} \tag{7.9}$$

Meanwhile the adjoint differential equations for the Euler formulation were given by (5.18) as

$$\begin{aligned}\frac{\partial \psi}{\partial t} = \quad & \left(J\frac{\partial \xi}{\partial x}\left(\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right)^T + J\frac{\partial \xi}{\partial y}\left(\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right)^T\right)\frac{\partial \psi}{\partial \xi} \\ + & \left(J\frac{\partial \eta}{\partial x}\left(\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right)^T + J\frac{\partial \eta}{\partial y}\left(\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right)^T\right)\frac{\partial \psi}{\partial \eta}. \qquad (7.10)\end{aligned}$$

The main difference between (7.10) and the corresponding flow equations, (5.5),

$$\frac{\partial W}{\partial t} = -\frac{\partial F}{\partial \xi} - \frac{\partial G}{\partial \eta} = 0 \qquad (7.11)$$

is that (7.10) is a linear equation which is not in strong conservation form. The most straightforward discretization of (7.10) is written as

$$\begin{aligned}\frac{\partial \psi_{i,j}}{\partial t} = \quad & \left(J\frac{\partial \xi}{\partial x}\left(\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right)^T + J\frac{\partial \xi}{\partial y}\left(\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right)^T\right)_{i,j}\left(\frac{1}{2}\psi_{i+1,j} - \frac{1}{2}\psi_{i-1,j}\right) \\ + & \left(J\frac{\partial \eta}{\partial x}\left(\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right)^T + J\frac{\partial \eta}{\partial y}\left(\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right)^T\right)_{i,j}\left(\frac{1}{2}\psi_{i,j+1} - \frac{1}{2}\psi_{i,j-1}\right), \qquad (7.12)\end{aligned}$$

where $\left(\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right)^T$ and $\left(\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right)^T$ are defined in Section (2.5.2). Equation (7.12), which will later be augmented with dissipation, will be used as the primary discretization of the domain equations for the Euler adjoint and is referred to as the *Type 1* fluxes. However, before proceeding to the development of the artificial viscosity terms and the necessary boundary conditions, it is beneficial to determine how different this is from the analogous discrete sensitivity stencil. A superficial examination of (7.12) indicates that a slight difference must exist, because (7.12) does not contain terms for $\psi_{i,j}$ on the right hand side, irrespective of the physical mesh used, while (7.9) is a function of $\mathbf{w}_{i,j}$ in all but a Cartesian mesh. Therefore, if the transpose of the linearized form of (7.9) was obtained, it would necessarily include $\psi_{i,j}$.

The first step in determining the formulation necessary for the discrete sensitivity analysis is to develop the delta form of (7.9). For steady state solutions, the fully implicit, discretely equivalent flow equation for (7.9) may be written, again without artificial viscosity, as

$$\left(J\frac{\partial \xi}{\partial x}\right)_{i+\frac{1}{2},j}\frac{1}{2}\left(\left[\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right]_{i+1,j}\delta \mathbf{w}_{i+1,j} + \left[\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right]_{i,j}\delta \mathbf{w}_{i,j}\right)$$

$$
\begin{aligned}
&+ \left(J\frac{\partial\xi}{\partial y}\right)_{i+\frac{1}{2},j} \frac{1}{2}\left(\left[\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right]_{i+1,j} \delta\mathbf{w}_{i+1,j} + \left[\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right]_{i,j} \delta\mathbf{w}_{i,j}\right) \\
&- \left(J\frac{\partial\xi}{\partial x}\right)_{i-\frac{1}{2},j} \frac{1}{2}\left(\left[\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right]_{i,j} \delta\mathbf{w}_{i,j} + \left[\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right]_{i-1,j} \delta\mathbf{w}_{i-1,j}\right) \\
&- \left(J\frac{\partial\xi}{\partial y}\right)_{i-\frac{1}{2},j} \frac{1}{2}\left(\left[\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right]_{i,j} \delta\mathbf{w}_{i,j} + \left[\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right]_{i-1,j} \delta\mathbf{w}_{i-1,j}\right) \\
&+ \left(J\frac{\partial\eta}{\partial x}\right)_{i,j+\frac{1}{2}} \frac{1}{2}\left(\left[\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right]_{i,j+1} \delta\mathbf{w}_{i,j+1} + \left[\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right]_{i,j} \delta\mathbf{w}_{i,j}\right) \\
&+ \left(J\frac{\partial\eta}{\partial y}\right)_{i,j+\frac{1}{2}} \frac{1}{2}\left(\left[\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right]_{i,j+1} \delta\mathbf{w}_{i,j+1} + \left[\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right]_{i,j} \delta\mathbf{w}_{i,j}\right) \\
&- \left(J\frac{\partial\eta}{\partial x}\right)_{i,j-\frac{1}{2}} \frac{1}{2}\left(\left[\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right]_{i,j} \delta\mathbf{w}_{i,j} + \left[\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right]_{i,j-1} \delta\mathbf{w}_{i,j-1}\right) \\
&- \left(J\frac{\partial\eta}{\partial y}\right)_{i,j-\frac{1}{2}} \frac{1}{2}\left(\left[\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right]_{i,j} \delta\mathbf{w}_{i,j} + \left[\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right]_{i,j-1} \delta\mathbf{w}_{i,j-1}\right) \\
&= -\left\{\left(\frac{1}{2}F^{+}_{i+1,j} + \frac{1}{2}F^{+}_{i,j}\right) - \left(\frac{1}{2}F^{-}_{i,j} + \frac{1}{2}F^{-}_{i-1,j}\right)\right. \\
&\qquad \left. + \left(\frac{1}{2}G^{+}_{i,j+1} + \frac{1}{2}G^{+}_{i,j}\right) - \left(\frac{1}{2}G^{-}_{i,j} + \frac{1}{2}G^{-}_{i,j-1}\right)\right\}. \qquad (7.13)
\end{aligned}
$$

Equation (7.13) is simply Newton's method applied directly to the discrete system of equation (7.9). Its solution would be identical to solutions from any other procedure for solving (7.9). Now using the linear algebra steps presented in the introduction, (1.7-1.9), it follows that the discrete sensitivity discretization of the adjoint equation is given by taking the transpose of the right hand side of (7.13):

$$
\begin{aligned}
&- \left\{\left(J\frac{\partial\xi}{\partial x}\right)_{i+\frac{1}{2},j} \left[\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right]^T_{i,j} + \left(J\frac{\partial\xi}{\partial y}\right)_{i+\frac{1}{2},j} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right]^T_{i,j}\right\} \frac{\boldsymbol{\psi}_{i+1,j}}{2} \\
&+ \left\{\left(J\frac{\partial\xi}{\partial x}\right)_{i-\frac{1}{2},j} \left[\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right]^T_{i,j} + \left(J\frac{\partial\xi}{\partial y}\right)_{i-\frac{1}{2},j} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right]^T_{i,j}\right\} \frac{\boldsymbol{\psi}_{i-1,j}}{2} \\
&- \left\{\left(J\frac{\partial\eta}{\partial x}\right)_{i,j+\frac{1}{2}} \left[\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right]^T_{i,j} + \left(J\frac{\partial\eta}{\partial y}\right)_{i,j+\frac{1}{2}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right]^T_{i,j}\right\} \frac{\boldsymbol{\psi}_{i,j+1}}{2} \\
&+ \left\{\left(J\frac{\partial\eta}{\partial x}\right)_{i,j-\frac{1}{2}} \left[\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right]^T_{i,j} + \left(J\frac{\partial\eta}{\partial y}\right)_{i,j-\frac{1}{2}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right]^T_{i,j}\right\} \frac{\boldsymbol{\psi}_{i,j-1}}{2}
\end{aligned}
$$

$$
\begin{aligned}
&+ \left\{\left(J\frac{\partial\xi}{\partial x}\right)_{i+\frac{1}{2},j} - \left(J\frac{\partial\xi}{\partial x}\right)_{i-\frac{1}{2},j} + \left(J\frac{\partial\eta}{\partial x}\right)_{i,j+\frac{1}{2}} - \left(J\frac{\partial\eta}{\partial x}\right)_{i,j-\frac{1}{2}}\right\} \left[\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right]^T_{i,j} \frac{\psi_{i,j}}{2} \\
&+ \left\{\left(J\frac{\partial\xi}{\partial y}\right)_{i+\frac{1}{2},j} - \left(J\frac{\partial\xi}{\partial y}\right)_{i-\frac{1}{2},j} + \left(J\frac{\partial\eta}{\partial y}\right)_{i,j+\frac{1}{2}} - \left(J\frac{\partial\eta}{\partial y}\right)_{i,j-\frac{1}{2}}\right\} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right]^T_{i,j} \frac{\psi_{i,j}}{2} \\
&= -\frac{\partial I}{\partial \mathbf{w}} \qquad (7.14)
\end{aligned}
$$

Equation (7.14) is referred to as the *Type 2* fluxes. For use later, this equation is also written as

$$
\mathcal{M}_j^{i+}\psi_{i+1,j} + \mathcal{M}_j^{i-}\psi_{i-1,j} + \mathcal{M}_{j+}^{i}\psi_{i,j+1} + \mathcal{M}_{j-}^{i}\psi_{i,j-1} + \mathcal{M}_j^{i}\psi_{i,j} = -\frac{\partial I}{\partial \mathbf{w}},
$$

where the matrices $\mathcal{M}_j^{i+}$, $\mathcal{M}_j^{i-}$, $\mathcal{M}_{j+}^{i}$, $\mathcal{M}_{j-}^{i}$, and $\mathcal{M}_j^{i}$ have been introduced for convenience. By comparing (7.14) with (7.12) it is seen that the discrete sensitivity analysis form is much more complicated than the *Type 1* choice of discretization used for the continuous sensitivity approach. Note that this new stencil is a function of $\psi_{i,j}$, and is clearly a more consistent representation if the goal is to obtain an exact match with finite-difference gradients. However, due to the extra complexity of (7.14), its operational count and hence its computational cost will be higher. *Chapter 9* will illustrate some test cases using both discretizations for the Euler adjoint equation. As a final note, it is realized that (7.14) is not a full discrete sensitivity analysis method since the same linearization and transposition operations have not been applied to the discrete form of the artificial dissipation as well as to the large number of very subtle boundary condition contributions. No attempt will be made to realize such a full discrete sensitivity analysis approach.

7.6 Dissipation Used for the Euler Adjoint Equation

Due to the absence of $\psi_{i,j}$ on the left hand side equation (7.12) permits odd-even decoupling, and by itself it is ill-conditioned. The system must be augmented with artificial dissipation just as was done for the Euler flow equations. The resulting discrete equations used in this research are simply

$$\begin{aligned}\frac{\partial \psi_{i,j}}{\partial t} = & \left(J\frac{\partial \xi}{\partial x}\left(\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right)^T + J\frac{\partial \xi}{\partial y}\left(\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right)^T\right)_{i,j}\left(\frac{1}{2}\psi_{i+1,j} - \frac{1}{2}\psi_{i-1,j}\right) \\ + & \left(J\frac{\partial \eta}{\partial x}\left(\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right)^T + J\frac{\partial \eta}{\partial y}\left(\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\right)^T\right)_{i,j}\left(\frac{1}{2}\psi_{i,j+1} - \frac{1}{2}\psi_{i,j-1}\right) \\ + & \mathcal{D}\left(\psi_{i,j}\right), \end{aligned} \tag{7.15}$$

where $\mathcal{D}$ has the same form as that defined in Section (2.5.3). Note that the sign of $\mathcal{D}$ is the same as that in equation (2.33) while the sign of the equivalent $\mathcal{Q}$ is the opposite. This reflects the fact that the transpose operator does not change the sign of the 2nd or 4th order differences present in $\mathcal{D}$, but does change the sign of the first order operator in $\mathcal{Q}$. The choice of augmenting (7.12) by $\mathcal{D}$ without stepping through the integration by parts that was required to obtain (7.12) avoids possible complications related both to additional boundary terms arising in the system, and to the lack of self-adjointness of $\mathcal{D}$. To adhere to strict discrete consistency we would have to apply summation by parts to equation (2.33). However, since $\mathcal{D}$ is thought of as artificial in the first place, the approximation taken here is consistent within the framework of the continuous sensitivity approach.

$\mathcal{D}$ can now be written following Section (2.5.3) as

$$\mathcal{D}_{i,j} = \mathcal{D}^2_{i,j} - \mathcal{D}^4_{i,j},$$

with

$$\begin{aligned}\mathcal{D}^2_{i,j} &= \mathbf{d}^2_{i+\frac{1}{2},j} - \mathbf{d}^2_{i-\frac{1}{2},j} + \mathbf{d}^2_{i,j+\frac{1}{2}} - \mathbf{d}^2_{i,j-\frac{1}{2}}, \\ \mathcal{D}^4_{i,j} &= \mathbf{d}^4_{i+\frac{1}{2},j} - \mathbf{d}^4_{i-\frac{1}{2},j} + \mathbf{d}^{(4}_{i,j+\frac{1}{2}} - \mathbf{d}^4_{i,j-\frac{1}{2}}.\end{aligned}$$

As an example of the terms on the right, the $i + \frac{1}{2}$ contributions are given by

$$\begin{aligned}\mathbf{d}^2_{i+\frac{1}{2},j} &= \varepsilon^2_{i+\frac{1}{2},j}\left(\psi_{i+1,j} - \psi_{i,j}\right), \\ \mathbf{d}^4_{i+\frac{1}{2},j} &= \varepsilon^4_{i+\frac{1}{2},j}\left(\psi_{i+2,j} - 3\psi_{i+1,j} + 3\psi_{i,j} - \psi_{i-1,j}\right).\end{aligned}$$

The dissipation that is applied to the fourth component of the ψ vector is in this case no different from the other components. For the Euler flow equations, the dissipation was applied to ρH instead of $\rho E = \mathrm{w}_4$ for the fourth component. Other than this change, all the same rules outlined in Section (2.5.3) are used to determine the coefficients of ε^2 and ε^4, and hence the dissipation. This approximation amounts to assuming that the dissipation is entirely self-adjoint at the discrete level.

7.7 Iteration Scheme and Convergence Acceleration for the Euler Adjoint Equation

For the Euler equations, the fact that a simple explicit scheme was used as the iteration algorithm for the flow solver makes it exceedingly simple to apply the same algorithm for the adjoint solver. It was shown that things were not so simple for the potential flow equation where a semi-implicit ADI method was used, and hence the stencil of the approximate factors in the scheme, as well as the sweep directions, required adjustment. Without repetition the multistage Runge-Kutta-like time stepping procedure as well as the method of calculating the local CFL numbers discussed in Section (2.5.2) are reused here to solve the adjoint equations. Further, the convergence is accelerated according to Section (2.5.4) by multigridding and residual smoothing. Enthalpy damping is neglected because no simple total enthalpy analogy was derived for the Euler adjoint equations in this research. This does not imply that such an analogy does not exist. It will remain as an area of future research to try and develop such an analogy and employ it to enhance convergence.

7.8 Boundary Condition Discretization for Euler Adjoint Equations

Unlike the adjoint for the potential flow formulation, where the surface boundary condition determines the value of a single scalar quantity with a single auxiliary equation, the situation for the Euler adjoint formulation is more challenging. Four components of the vector ψ must be determined at the wall with only a single auxiliary equation. This is identical to the situation for the corresponding surface boundary conditions for Euler flow solvers. The only wall boundary condition for the Euler flow equations is the no-flux condition. While this is also true for the potential flow equation, it fully determines the result since there is only a single scalar unknown. This important difference has the following implication for the solution of the two adjoint systems:

1. For the potential flow adjoint equation:
   - The operator for ψ in the domain is second order and Laplacian like, resulting in a boundary value problem requiring the same number of equations as unknowns at the boundaries.
   - Differences in the discretization and application of (4.17), if consistent, may affect the quantitative accuracy of the adjoint solution but not its qualitative aspects.
2. For the Euler adjoint equation:
   - The operator for ψ in the domain is first order, resulting in a mixed initial boundary value problem that requires fewer equations than unknowns at the boundary.
   - The boundary conditions for Euler adjoint equation must be applied, just as in the case of the Euler flow equations, with careful attention to the characteristics of the system to avoid over-specification.

With these points established, three alternative wall boundary conditions for the Euler adjoint equations will be presented, all satisfying (5.19).

7.8.1 BOUNDARY CONDITION *Type 1*

The first discretization of (5.19) has the following form:

$$\begin{aligned}\psi_1^- &= 0\\ \psi_2^- &= -\Phi J\frac{\partial\eta}{\partial x}\left[(p-p_d)\frac{ds}{d\xi}\right]\end{aligned}$$

$$\psi_3^- = -\Phi J \frac{\partial \eta}{\partial y}\left[(p - p_d)\frac{ds}{d\xi}\right]$$

$$\psi_4^- = 0 \tag{7.16}$$

where ψ^- refers to fictitious values of ψ below the surface, ψ^+ are the values above the surface, and

$$\Phi = \frac{1}{J^2\left(\left(\frac{\partial\eta}{\partial x}\right)^2 + \left(\frac{\partial\eta}{\partial y}\right)^2\right)}.$$

It is noted that all the metric components are calculated at the wall itself. Figure (7.1)

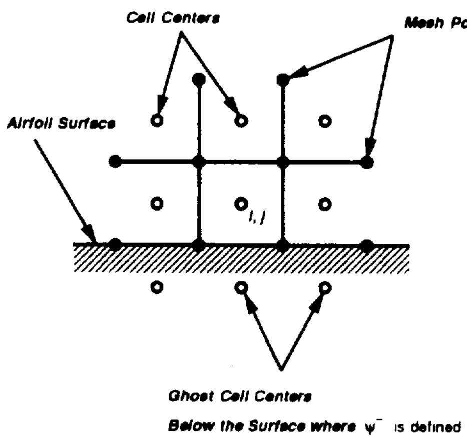


Figure 7.1: Computational Mesh for the Euler Adjoint Equation at the Airfoil Surface

illustrates the stencil in the coordinate system transformed into the computational plane. Thus an application of (7.15) just above the surface would refer to values of ψ below the surface which would be determined by (7.16). Equation (7.16) can be shown, through multiplication of ψ_2^- by $\frac{\partial\eta}{\partial x}$, and ψ_3^- by $\frac{\partial\eta}{\partial y}$, to be a linear combination of (5.19) centered about the cell below the wall. This is but one of an infinite number of linear combinations that would satisfy the condition that the *normal adjoint velocity*, $\left(\frac{\partial\eta}{\partial x}\psi_2 + \frac{\partial\eta}{\partial y}\psi_3\right)$, is equal to some scaled quantity related to the cost function; that is, equation (5.19) is satisfied. The particular choice of (7.16) is such that at the surface, the *tangential adjoint velocity*, $\left(\frac{\partial\xi}{\partial x}\psi_2 + \frac{\partial\xi}{\partial y}\psi_3\right)$, goes to zero with an orthogonal mesh at the surface ($\frac{\partial x}{\partial\eta}\frac{\partial x}{\partial\xi} + \frac{\partial y}{\partial\eta}\frac{\partial y}{\partial\xi} = 0$), while ψ_1 and ψ_4 are arbitrarily set to zero. The nomenclature *normal* and *tangential adjoint velocities* refers to the correspondence between the definitions of the contravariant velocities and these quantities; they are the same relations, where ψ replaces w for the adjoint expressions. *This results in the interesting analogy that the wall boundary condition (5.19) effectively sets the transpiration velocity for the adjoint variable at the surface as equal to some scaled quantity related to the cost function.* This conclusion reveals quite elegant implications for the connection between the adjoint-based design methods and those that use true transpiration at the surface. The choice for ψ_1, ψ_4 and the zero tangential adjoint velocity in (7.16) does not necessarily violate any expression in the derivation of the adjoint equations. However, just as in the solution of the Euler flow equations, over-specifying the boundary conditions can have disastrous effects. When solutions were attempted using (7.16), convergence could be achieved for the adjoint system. However, the accuracy of the gradient, when compared with those obtained by finite differences, was quite poor. A close investigation of the adjoint solution revealed that the system decoupled at the surface, resulting in dramatic jumps in all components of ψ at the surface. With these jumps in ψ, the solution quite clearly is not continuously differentiable and it fails to satisfy the rules of integration by parts which were used in the derivation of the method.

7.8.2 BOUNDARY CONDITION *Type 2*

Instead of fixing the three undetermined components of the solution to some arbitrary value (zero) at the surface, they should actually be allowed to "float." This should result in continuous solutions for ψ, and mimics the correct wall boundary condition for the Euler flow equations, where only zero flux is specified, with the other components determined from extrapolation. Thus, a second set of discrete boundary conditions at the airfoil surface may be defined as

$$\begin{aligned}
\psi_1^- &= \psi_1^+ \\
\psi_2^- &= \psi_2^+ - 2\Phi J \frac{\partial \eta}{\partial x}\left[(p - p_d)\frac{ds}{d\xi} + J\frac{\partial \eta}{\partial x}\psi_2^+ + J\frac{\partial \eta}{\partial y}\psi_3^+\right] \\
\psi_3^- &= \psi_3^+ - 2\Phi J \frac{\partial \eta}{\partial y}\left[(p - p_d)\frac{ds}{d\xi} + J\frac{\partial \eta}{\partial x}\psi_2^+ + J\frac{\partial \eta}{\partial y}\psi_3^+\right] \\
\psi_4^- &= \psi_4^+ . \qquad (7.17)
\end{aligned}$$

It is easily shown that the discrete representation of (5.19) contained in (7.17) is

$$\frac{J}{2}\left(\frac{\partial \eta}{\partial x}\left(\psi_2^- + \psi_2^+\right) + \frac{\partial \eta}{\partial y}\left(\psi_3^- + \psi_3^+\right)\right) = -(p - p_d)\frac{ds}{d\xi}.$$

Furthermore, by multiplying the equation for ψ_2^- by $\frac{\partial \xi}{\partial x}$ and the equation for ψ_3^- by $\frac{\partial \xi}{\partial y}$ and summing the result, it can be shown that for an orthogonal mesh ($\frac{\partial \xi}{\partial y}\frac{\partial \eta}{\partial y} + \frac{\partial \xi}{\partial x}\frac{\partial \eta}{\partial x} = 0$)

$$\left(\frac{\partial \xi}{\partial x}\psi_2^- + \frac{\partial \xi}{\partial y}\psi_3^-\right) = \left(\frac{\partial \xi}{\partial x}\psi_2^+ + \frac{\partial \xi}{\partial y}\psi_3^+\right).$$

Again, just as in boundary condition *Type 1*, all the mesh metrics are evaluated at the edge centers. In practice, the use of (7.17) as the wall boundary condition for the adjoint system gives gradients that are very consistent with those obtained by finite differences.

7.8.3 BOUNDARY CONDITION *Type 3*

The fact that even with equation (7.17), no correction due to the application of pressure gradient boundary condition (2.42) in the flow solution appears in the development of the adjoint boundary condition indicates a possible source of inconsistency. Furthermore, just as in the development of *Type 1* and *Type 2* interior flux routines it may be beneficial to examine the particular form of the boundary condition discretization that results from discrete sensitivity analysis. Thus, it may be beneficial to formulate the wall boundary condition that would result from an application of discrete sensitivity analysis which includes the pressure gradient term. The same approach used earlier in this chapter to develop the field discretization of the discrete adjoint will again be employed. First, a crude approximation to the no-penetration boundary condition used for the Euler equations may be written as

$$\begin{aligned} w_1^- &= w_1^+ \\ w_2^- &= J\frac{\partial \eta}{\partial y}q_t^+ + J\frac{\partial \eta}{\partial x}q_n^+ \\ w_3^- &= J\frac{\partial \eta}{\partial y}q_n^+ - J\frac{\partial \eta}{\partial x}q_t^+ \\ w_4^- &= w_4^+ \end{aligned} \tag{7.18}$$

where

$$\begin{aligned} q_t^+ &= J\frac{\partial \eta}{\partial y}w_2^+ - J\frac{\partial \eta}{\partial x}w_3^+ \\ q_n^+ &= -J\frac{\partial \eta}{\partial x}w_3^+ - J\frac{\partial \eta}{\partial y}w_2^+ \end{aligned}$$

are the tangential and normal mass fluxes calculated at the cell centers just above the surface. Taking the first variation of these equations gives

$$\begin{aligned} \delta w_1^- &= \delta w_1^+ \\ \delta w_2^- &= \left(\frac{\partial x}{\partial s}^2 - \frac{\partial y}{\partial s}^2\right)\delta w_2^+ + 2\frac{\partial x}{\partial s}\frac{\partial y}{\partial s}\delta w_3^+ \\ \delta w_3^- &= 2\frac{\partial x}{\partial s}\frac{\partial y}{\partial s}\delta w_2^+ - \left(\frac{\partial x}{\partial s}^2 - \frac{\partial y}{\partial s}^2\right)\delta w_3^+ \\ \delta w_4^- &= \delta w_4^+ \\ \delta p^- &= \delta p^+, \end{aligned} \tag{7.19}$$

where the relationship between δp^- below the surface and δp^+ above the surface acts as a place holder for the more complex relationship used later. Equation (7.19) determines the unknowns below the wall as a function of values above the wall. If either equation (7.12) or (7.13) is used as the equation above the wall and into the domain, the system may be closed by specifying

$$\delta p^+ = \frac{\gamma - 1}{2}\frac{\left(w_2^2 + w_3^2\right)}{w_1^2}\delta w_1 - (\gamma - 1)\frac{w_2}{w_1}\delta w_2 - (\gamma - 1)\frac{w_3}{w_1}\delta w_3 + (\gamma - 1)\delta w_4,$$

where all the terms on the right are evaluated above the wall. Just as in the formulation of the discrete adjoint in the domain, the transpose of this boundary system gives a discrete sensitivity adjoint boundary condition. After a bit of tedious book-keeping these boundary conditions may be defined for the cell-centered values of $\psi_{i,j}$ just above the surface as

$$\left[\mathcal{M}_j^{i+}\right]\begin{bmatrix}\psi_{1_{i+1,j}}\\ \psi_{2_{i+1,j}}\\ \psi_{3_{i+1,j}}\\ \psi_{4_{i+1,j}}\end{bmatrix}+\left[\mathcal{M}_j^{i-}\right]\begin{bmatrix}\psi_{1_{i-1,j}}\\ \psi_{2_{i-1,j}}\\ \psi_{3_{i-1,j}}\\ \psi_{4_{i-1,j}}\end{bmatrix}+\left[\mathcal{M}_{j+}^{i}\right]\begin{bmatrix}\psi_{1_{i,j+1}}\\ \psi_{2_{i,j+1}}\\ \psi_{3_{i,j+1}}\\ \psi_{4_{i,j+1}}\end{bmatrix}$$

$$+\begin{bmatrix}\psi_{1_{i,j-1}}\\ \psi_{2_{i,j-1}}\left(\frac{\partial x}{\partial s}^2-\frac{\partial y}{\partial s}^2\right)+\psi_{3_{i,j-1}}\left(2\frac{\partial x}{\partial s}\frac{\partial y}{\partial s}\right)\\ \psi_{2_{i,j-1}}\left(2\frac{\partial x}{\partial s}\frac{\partial y}{\partial s}\right)-\psi_{3_{i,j-1}}\left(\frac{\partial x}{\partial s}^2-\frac{\partial y}{\partial s}^2\right)\\ \psi_{4_{i,j-1}}\end{bmatrix}+\begin{bmatrix}\psi_{p_{i,j}}\frac{\gamma-1}{2}\frac{(\mathbf{w}_2^2+\mathbf{w}_3^2)}{\mathbf{w}_1^2}\\ -\psi_{p_{i,j}}(\gamma-1)\frac{\mathbf{w}_2}{\mathbf{w}_1}\\ -\psi_{p_{i,j}}(\gamma-1)\frac{\mathbf{w}_3}{\mathbf{w}_1}\\ \psi_{p_{i,j}}(\gamma-1)\end{bmatrix}$$

$$+\quad\left[\mathcal{M}_j^{i}\right]\begin{bmatrix}\psi_{1_{i,j}}\\ \psi_{2_{i,j}}\\ \psi_{3_{i,j}}\\ \psi_{4_{i,j}}\end{bmatrix}=0 \tag{7.20}$$

with ψ_p being an auxiliary unknown analogous to pressure for the flow equations. Note that this is the same as equation (7.14) — the standard discrete sensitivity discretization for the domain fluxes — where the $\psi_{i,j-1}$ terms have been replaced by the appropriate boundary term. Below the surface, where the i, j index is still considered the cell-centered point just above the surface, the transpose operator gives

$$\begin{bmatrix}\psi_{1_{i,j-1}}\\ \psi_{2_{i,j-1}}\\ \psi_{3_{i,j-1}}\\ \psi_{4_{i,j-1}}\end{bmatrix}=\left[\mathcal{M}_{j-}^{i}\right]\begin{bmatrix}\psi_{1_{i,j}}\\ \psi_{2_{i,j}}\\ \psi_{3_{i,j}}\\ \psi_{4_{i,j}}\end{bmatrix}. \tag{7.21}$$

To close the system, the equations for ψ_p are defined with the same transpose by,

$$\psi_{p_{i,j-1}}-\psi_{p_{i,j}} \quad = \quad \left\{\frac{(p-p_d)}{2}\right\}_{i,j-\frac{1}{2}}\left(\frac{ds}{d\xi}\right)_{i,j-\frac{1}{2}}$$

$$
\begin{aligned}
-\psi_{p_{i,j-1}} &= \left\{\frac{(p-p_d)}{2}\right\}_{i,j-\frac{1}{2}} \left(\frac{ds}{d\xi}\right)_{i,j-\frac{1}{2}} \\
\psi_{p_{i,j}} &= \{(p-p_d)\}_{i,j-\frac{1}{2}} \left(\frac{ds}{d\xi}\right)_{i,j-\frac{1}{2}}. \qquad (7.22)
\end{aligned}
$$

The system (7.20—7.22) is now complete and is the exact discrete sensitivity analysis boundary condition for the adjoint equation if (7.18) is used as the wall boundary condition for the Euler equations. Most Euler solvers attempt a better approximation for the pressure boundary condition than $p^- = p^+$ seen in (7.18). For the flow solver used in this research, the expression for pressure at the surface is obtained from equation (2.42):

$$(x_\xi^2 + y_\xi^2)p_\eta = (x_\xi x_\eta + y_\xi y_\eta)p_\xi + \rho(y_\eta u - x_\eta v)(v x_{\xi\xi} - u y_{\xi\xi}).$$

The discrete form of this equation can be written as

$$-\left(x_\xi^2 + y_\xi^2\right)_{i,j-\frac{1}{2}}(p_{i,j} - p_{i,j-1}) + (x_\xi x_\eta + y_\xi y_\eta)_{i,j-\frac{1}{2}} \frac{(p_{i+1,j} - p_{i-1,j})}{2} + \frac{\mathcal{B}_1\mathcal{B}_2}{\mathcal{B}_3},$$

where

$$\mathcal{B}_1 = \left(y_{\eta_{i,j-\frac{1}{2}}} \frac{\left(\mathbf{w}2_{i,j} + \mathbf{w}2_{i,j-1}\right)}{2} - x_{\eta_{i,j-\frac{1}{2}}} \frac{\left(\mathbf{w}3_{i,j} + \mathbf{w}3_{i,j-1}\right)}{2}\right),$$

$$\mathcal{B}_2 = \left(x_{\xi\xi_{i,j-\frac{1}{2}}} \frac{\left(\mathbf{w}3_{i,j} + \mathbf{w}3_{i,j-1}\right)}{2} - y_{\xi\xi_{i,j-\frac{1}{2}}} \frac{\left(\mathbf{w}2_{i,j} + \mathbf{w}2_{i,j-1}\right)}{2}\right),$$

$$\mathcal{B}_3 = \frac{\left(\mathbf{w}3_{i,j} + \mathbf{w}3_{i,j-1}\right)}{2}.$$

Taking the first variation of this equation gives

$$
\begin{aligned}
&- \left(x_\xi^2 + y_\xi^2\right)_{i,j-\frac{1}{2}}(\delta p_{i,j} - \delta p_{i,j-1}) + (x_\xi x_\eta + y_\xi y_\eta)_{i,j-\frac{1}{2}} \frac{(\delta p_{i+1,j} - \delta p_{i-1,j})}{2} \\
&+ \frac{\left(y_{\eta_{i,j-\frac{1}{2}}} \frac{\left(\delta\mathbf{w}2_{i,j} + \delta\mathbf{w}2_{i,j-1}\right)}{2} - x_{\eta_{i,j-\frac{1}{2}}} \frac{\left(\delta\mathbf{w}3_{i,j} + \delta\mathbf{w}3_{i,j-1}\right)}{2}\right)\mathcal{B}_2}{\mathcal{B}_3} \\
&+ \frac{\mathcal{B}_1\left(x_{\xi\xi_{i,j-\frac{1}{2}}} \frac{\left(\delta\mathbf{w}3_{i,j} + \delta\mathbf{w}3_{i,j-1}\right)}{2} - y_{\xi\xi_{i,j-\frac{1}{2}}} \frac{\left(\delta\mathbf{w}2_{i,j} + \delta\mathbf{w}2_{i,j-1}\right)}{2}\right)}{\mathcal{B}_3} \\
&- \frac{\mathcal{B}_1\mathcal{B}_2}{\mathcal{B}_3^2} \frac{\left(\delta\mathbf{w}3_{i,j} + \delta\mathbf{w}3_{i,j-1}\right)}{2} = 0.
\end{aligned}
$$

Again after some book-keeping the expressions for the adjoint variable below the wall in equation (7.21) may be augmented to give

$$\begin{bmatrix} \psi_{1_{i,j-1}} \\ \psi_{2_{i,j-1}} \\ \psi_{3_{i,j-1}} \\ \psi_{4_{i,j-1}} \end{bmatrix} + \begin{bmatrix} -\frac{\mathcal{B}_1\mathcal{B}_2}{2\mathcal{B}_3^2}\psi_{p_{i,j-1}} \\ \frac{\left(y_{\eta_{i,j-\frac{1}{2}}}\mathcal{B}_2 - y_{\xi\xi_{i,j-\frac{1}{2}}}\mathcal{B}_1\right)}{2\mathcal{B}_3}\psi_{p_{i,j-1}} \\ \frac{\left(-x_{\eta_{i,j-\frac{1}{2}}}\mathcal{B}_2 + x_{\xi\xi_{i,j-\frac{1}{2}}}\mathcal{B}_1\right)}{2\mathcal{B}_3}\psi_{p_{i,j-1}} \\ 0 \end{bmatrix} = \left[\mathcal{M}^i_{j-}\right] \begin{bmatrix} \psi_{1_{i,j}} \\ \psi_{2_{i,j}} \\ \psi_{3_{i,j}} \\ \psi_{4_{i,j}} \end{bmatrix}. \tag{7.23}$$

Similarly the equations above the wall are also augmented, resulting in

$$\left[\mathcal{M}^{i+}_j\right]\begin{bmatrix} \psi_{1_{i+1,j}} \\ \psi_{2_{i+1,j}} \\ \psi_{3_{i+1,j}} \\ \psi_{4_{i+1,j}} \end{bmatrix} + \left[\mathcal{M}^{i-}_j\right]\begin{bmatrix} \psi_{1_{i-1,j}} \\ \psi_{2_{i-1,j}} \\ \psi_{3_{i-1,j}} \\ \psi_{4_{i-1,j}} \end{bmatrix} + \left[\mathcal{M}^i_{j+}\right]\begin{bmatrix} \psi_{1_{i,j+1}} \\ \psi_{2_{i,j+1}} \\ \psi_{3_{i,j+1}} \\ \psi_{4_{i,j+1}} \end{bmatrix}$$

$$+ \begin{bmatrix} \psi_{1_{i,j-1}} \\ \psi_{2_{i,j-1}}\left(\frac{\partial x}{\partial s}^2 - \frac{\partial y}{\partial s}^2\right) + \psi_{3_{i,j-1}}\left(2\frac{\partial x}{\partial s}\frac{\partial y}{\partial s}\right) \\ \psi_{2_{i,j-1}}\left(2\frac{\partial x}{\partial s}\frac{\partial y}{\partial s}\right) - \psi_{3_{i,j-1}}\left(\frac{\partial x}{\partial s}^2 - \frac{\partial y}{\partial s}^2\right) \\ \psi_{4_{i,j-1}} \end{bmatrix}$$

$$+ \begin{bmatrix} \psi_{p_{i,j}}\frac{\gamma-1}{2}\frac{(\mathbf{w}_2^2+\mathbf{w}_3^2)}{\mathbf{w}_1^2} \\ -\psi_{p_{i,j}}(\gamma-1)\frac{\mathbf{w}_2}{\mathbf{w}_1} \\ -\psi_{p_{i,j}}(\gamma-1)\frac{\mathbf{w}_3}{\mathbf{w}_1} \\ \psi_{p_{i,j}}(\gamma-1) \end{bmatrix} + \begin{bmatrix} -\frac{\mathcal{B}_1\mathcal{B}_2}{2\mathcal{B}_3^2}\psi_{p_{i,j-1}} \\ \frac{\left(y_{\eta_{i,j-\frac{1}{2}}}\mathcal{B}_2 - y_{\xi\xi_{i,j-\frac{1}{2}}}\mathcal{B}_1\right)}{2\mathcal{B}_3}\psi_{p_{i,j-1}} \\ \frac{\left(-x_{\eta_{i,j-\frac{1}{2}}}\mathcal{B}_2 + x_{\xi\xi_{i,j-\frac{1}{2}}}\mathcal{B}_1\right)}{2\mathcal{B}_3}\psi_{p_{i,j-1}} \\ 0 \end{bmatrix}$$

$$+ \left[\mathcal{M}^i_j\right]\begin{bmatrix} \psi_{1_{i,j}} \\ \psi_{2_{i,j}} \\ \psi_{3_{i,j}} \\ \psi_{4_{i,j}} \end{bmatrix} = 0. \tag{7.24}$$

The system is now closed by equations for ψ_p that are given by

$$\begin{aligned} (x_\xi^2 + y_\xi^2)_{i,j-\frac{1}{2}}\,\psi_{p_{i,j-1}} &= \left\{\frac{(p-p_d)}{2}\right\}_{i,j-\frac{1}{2}}\left(\frac{ds}{d\xi}\right)_{i,j-\frac{1}{2}} \\ -(x_\xi^2 + y_\xi^2)_{i,j-\frac{1}{2}}\,\psi_{p_{i,j-1}} - \psi_{p_{i,j}} + (x_\xi x_\eta + y_\xi y_\eta)_{i-1,j-\frac{1}{2}}\,\psi_{p_{i-1,j-1}} &= \left\{\frac{(p-p_d)}{2}\right\}_{i,j-\frac{1}{2}}\left(\frac{ds}{d\xi}\right)_{i,j-\frac{1}{2}}. \end{aligned}$$

(7.25)

Equations (7.23—7.25) will be used as boundary condition *Type 3*. It must be restated that this set of boundary conditions still does not exactly duplicate that which would be obtained by direct application of the discrete sensitivity method. First, the assumed discretization of the Euler equation wall boundary condition used in the derivation of (7.23—7.25) is not identical to that used in the flow solver. Second, the dissipation contribution at the boundary that would result from discrete sensitivity analysis has been ignored. Comparisons among all three boundary conditions are given in *Chapter 9*.

7.8.4 FAR FIELD BOUNDARY CONDITION

While various choices for the discretization of the far field boundary conditions for the Euler adjoint equations that conform to equation (5.16) are possible, in this research these efforts were avoided. Instead, for all cases to be presented the outer boundary condition was kept very far from the geometry of interest and $\psi_{1-4} = 0$ was specified as the boundary condition.

Chapter 8

RESULTS FOR POTENTIAL FLOW DESIGN METHOD

The primary emphasis of the results will be those for the Euler formulation presented in the next chapter. In the case of the potential flow method the scope of the results will be limited since many of the same conclusions can be drawn from both methods.

8.1 CONVERGENCE

Before presenting validity checks of the design method, a short examination of the convergence characteristics of its various elements is necessary. Figure (8.1) shows the convergence characteristics of the flow and adjoint solvers for test Case 2 to be presented in Section (8.3). It features an inverse design at constant circulation starting from a NACA 0012 airfoil section with a specified target speed distribution of an RAE 2822 airfoil operating at Mach 0.75 and $\alpha = 1°$. A total of 50 Hicks-Henne functions distributed over the upper and lower surfaces are used to modify the design with the scalar weights of these functions acting as the design variables. The choice of the widths and locations of these bumps can be found in Tables (B.1 and B.2) of *Appendix B1*. It is noted from the tables that there is also a *scaling* parameter. Scaling in this context refers to a constant multiplier which is used to give the perturbation functions realistic values with respect to the airfoil coordinates. The convergence histories of the average residual for the state and co-state systems are shown in Figure (8.1) for the initial point in the design space. The flow equations are started from uniform flow with zero circulation, while the adjoint is started from zero values throughout the domain. Each iteration represents a full multigrid cycle of 5 meshes with 193x33 being the finest mesh, and one ADI sweep per mesh on a W-cycle. It is seen that both systems show reasonable convergence rates with the flow solver being somewhat better in this example. In general the adjoint system, although linear in its dependent variable, tends not to converge as well as the flow equations. However, it must be remembered that it took many years of research to tune iteration algorithms to optimize convergence rates for flow solvers. Here, it is gratifying that by simply using the identical algorithm for the adjoint solution, albeit with reverse biasing and sweep directions, comparable convergence rates were attainable. One tendency observed during various tests with the adjoint solver was that its convergence rate depended to a large extent on the character of the forcing function on the right hand side of the equation. While this as well as other issues were examined to some extent, a solid understanding of the reason that the adjoint solver often did not converge as well as the flow solver was not obtained. Further, just as in the use of flow solvers, different cases may experience drastically different adjoint solver convergence rates.

8.2 GRADIENT ACCURACY

The first step to validate the potential flow design method is to check the accuracy of the gradient calculated via the adjoint system. Although a complete treatment of the sensitivity of the gradient to convergence levels for both the flow and adjoint solvers could be helpful, these results tend to mirror trends obtained by the Euler design method and thus the reader is referred to the results in *Chapter 9*. Here only a one-to-one comparison is made between a highly accurate finite difference-based gradient and the adjoint-based gradient. The problem chosen for this comparison is again test Case 2 from the results to be presented in Section (8.3). This choice in many ways represents a good standard since it uses a well proven set of design variables, it has a unique solution, and it represents a difficult transonic problem. Figure (8.2) shows the comparison of gradients for the two approaches at the initial design point. The significant difference between the two approaches is that the gradient by means of the adjoint approach took 5.3 CPU seconds while the gradient for the finite difference approach took 18.2 seconds. *Chapter 9* presents a better understanding of the performance improvement to be expected by using an adjoint approach.

The two curves of Figure (8.2) are a simple connection of the points that represent the gradient with respect to each individual design variable in Tables (B.1 and B.2). The method of calculating the gradient for the adjoint approach is given in *Chapter 6*, while the gradient for the finite difference method simply uses, for each element, the difference in the cost functions directly evaluated from two nearby flow solutions. And since identical flow solvers and mesh generation/perturbation techniques were used for both methods, the two curves in Figure (8.2) should in theory be identical. The stepsize of the design variables used to calculate the gradients was 0.0001, while the tolerances of the convergence for the flow and adjoint solvers were 10^{-9} and 10^{-3} respectively. Results to be presented in *Chapter 9* indicate that these stepsizes and convergence tolerances are more than adequate to obtain highly consistent results for both methods. Therefore, the differences in the two curves, which have a very similar form, but are off in magnitude at some locations, probably result from an inconsistency of the particular discretization used for the continuous adjoint equation. Unlike the Euler equations, where the choice of the implementation of the adjoint boundary condition at the wall is not straightforward, the potential flow adjoint has a simple and easily implemented wall boundary condition. Thus it appears that a probable contribution to the discrepancies in the adjoint gradient is inconsistencies in the difference stencil of the adjoint domain equations. This is not surprising considering the complicated rotated differencing and upwind density biasing that was used for the flow solver. No attempt was made to capture the rotated difference effects in the adjoint discretization. In the limit of the continuous system these gradients should agree. For the potential flow design method, a more careful assessment of the various adjoint discretization options will not be explored. However, such studies did show significant improvements for the Euler formulation. The question that concerns us here is whether such errors in the gradient for the discrete system are tolerable in the context of the design problem.

8.3 TEST CASES

Some test cases are presented here for the potential flow design algorithm. For additional examples of using the design method developed here the reader is urged to review Reference [92] which was presented as a direct result of this work. These test cases can be categorized into two basic groups. The first group of test cases address the problem of attaining a desired pressure distribution for lifting airfoils. In all of these cases the target pressure distributions were developed by running the same FLO42 flow solver for a known airfoil. Thus, these airfoil targets must be thought of as fully realizable. For demonstration purposes, pains were taken to ensure that the flow solver was run with identical dissipation coefficients so that even the shock structures should in theory be realizable as part of the design. The most convenient method of designing in inverse mode with the present potential flow design method is to determine the lift coefficient associated with the target pressure distribution, and match this lift with the initial airfoil. The design progresses with the flow and adjoint systems being driven by constant circulation instead of fixed angle of attack. This choice is necessary since the potential flow adjoint method developed in *Chapter 4* is only applicable for the treatment of constant circulation cases. The reason that the solutions which will be presented show slight changes in C_l from the initial to the final designs is that C_l was calculated from a pressure integration and not determined by an explicit formula related to circulation. As a final note, in all of the test cases to be presented for the potential flow formulation, there was no need to apply either shock wave unweighting or smoothing to the forcing term of the adjoint boundary condition.

8.3.1 INVERSE DESIGN

The first example using this technique, Test Case 1, drives the NACA 0012 airfoil toward the target velocity distribution for the NACA 64A410 airfoil at $M_\infty = 0.735$, $\alpha = 0°$, and $C_l = 0.73$. Figure (8.3) shows the initial and final results for this test where C_ps instead of velocities are plotted for clarity. This case requires a shift in the shock location and a significant change in the profile shape for the target pressure distribution to be obtained. Figure (8.3) shows that the final solution almost exactly recovers the pressure distribution and the airfoil shape. Even the points through the shock region are matched one for one. Of course this level of comparison in the shock structure was attainable since the same level of dissipation was used to create the targets. If the shock structure happened to be quite different it might have proven to be a case where smoothing or shock wave unweighting of the adjoint boundary forcing term would have been needed. Slight discrepancies in the pressure distribution are apparent on the forward upper surface of the airfoil while no differences are perceptible in the airfoil shape. The slight difference can be attributed to either incompleteness in the design space or inaccuracies in the gradients.

In Test Case 2 the design program is again used in inverse mode and involves driving the NACA 0012 airfoil at $M_\infty = 0.75$ toward the shape that attains the target velocity distribution of the RAE airfoil at the same Mach number, $\alpha = 1.0^\circ$, and $C_l = 0.64$. Due to the steep favorable pressure gradient at the leading edge upper surface and the strong shock exhibited (see Figure (8.4)) by the RAE airfoil at these conditions, this case represents quite a difficult test for the design method. In the observed results, there are no discrepancies evident between the target and the final pressure distributions. However, a slight discrepancy is seen in the final airfoil shape. Observing that the final α is different from the $\alpha = 1^\circ$ used for obtaining the targets by almost 0.1° indicates that the designed airfoil has an effective rotation built into the coordinates that was compensated for by the constant lift design process. However, both airfoils have leading and trailing edges at 0.0; hence, the difference between the airfoils is not a simple rotation. As with Case 1, even the details of the shock wave structure are recovered by the design method.

8.3.2 Drag Minimization

The last test case introduces drag as the cost function. Again the design process is carried out in the fixed circulation mode. In Figure (8.5), the camber distribution of an airfoil is optimized to attain minimum drag. The design starts from a NACA 64A410 airfoil operating at $M_\infty = 0.75$, and $C_l = 0.79$. Under these conditions the airfoil produces 204 counts of drag according to potential flow theory. Eighteen Hicks-Henne functions as specified in Table (B.5) are chosen to modify both the upper and lower surfaces simultaneously such that thickness is maintained. The design then proceeds by maintaining constant lift and minimizing the large initial wave drag. After 4 design iterations, where only the camber was allowed to vary, the method has successfully reduced the drag to 10 counts. Both the initial and final states are depicted in Figure (8.5). Subsequent design iterations resulted in no significant further reduction in the cost function. This has at least three possible explanations. First, there is the possibility that no further reduction in drag is possible with the lift coefficient and thickness constraints. Second, it may be that the inaccuracies in the gradient were such that an improvement in the design could not be achieved in the specified search direction. Finally, there is the possibility that the parameterization of the design space was inadequate to allow a further improvement.

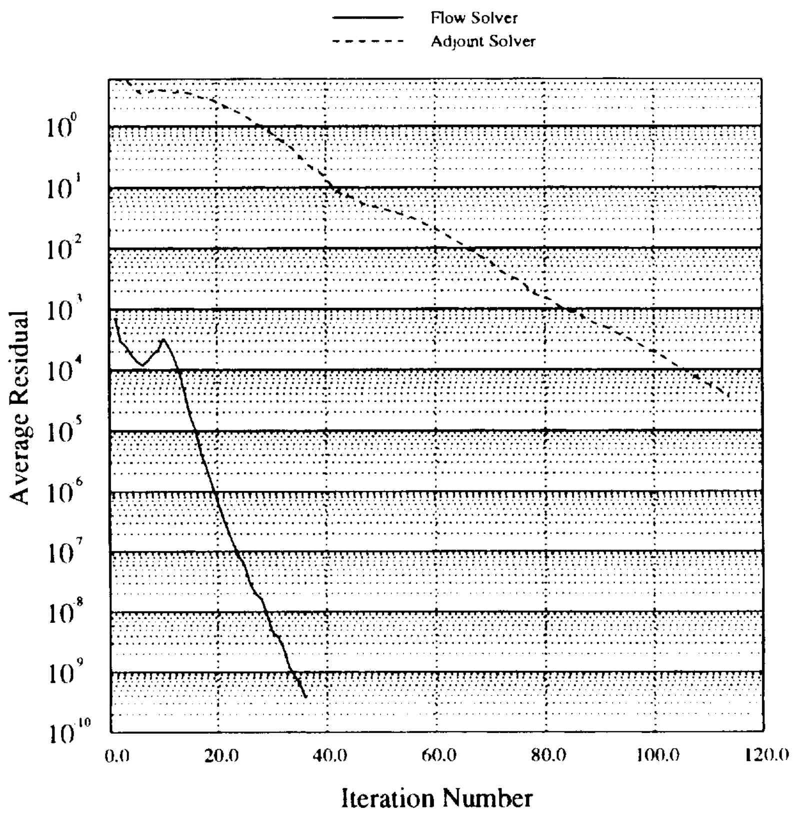


Figure 8.1: *Convergence Rates in Terms of Average Residual for FLO42 and ADJ42. Mesh = 193x33. 5 Mesh Multigrid Cycle.*

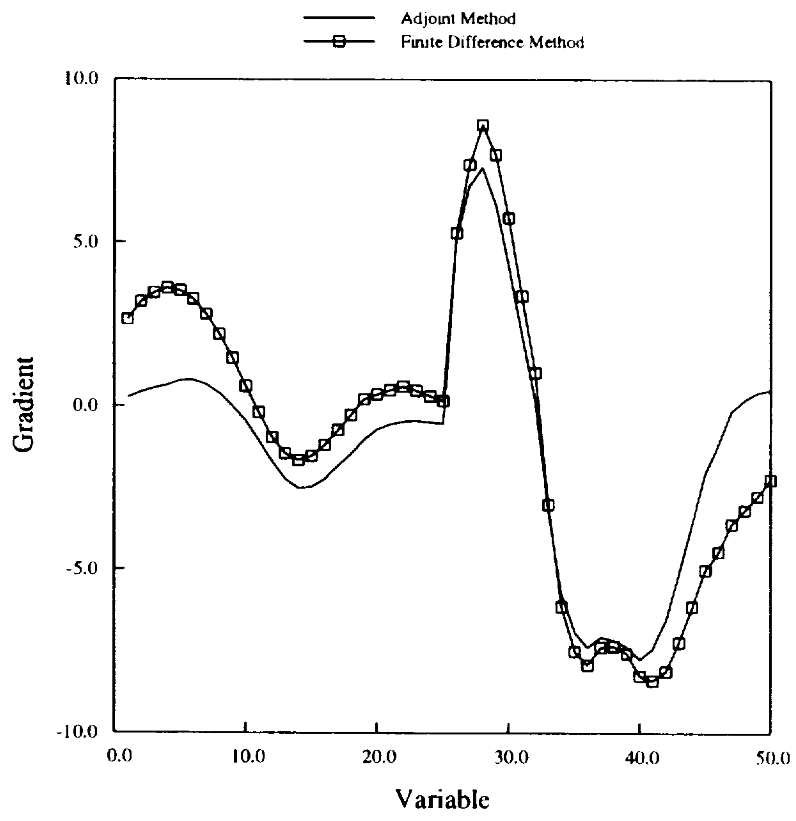
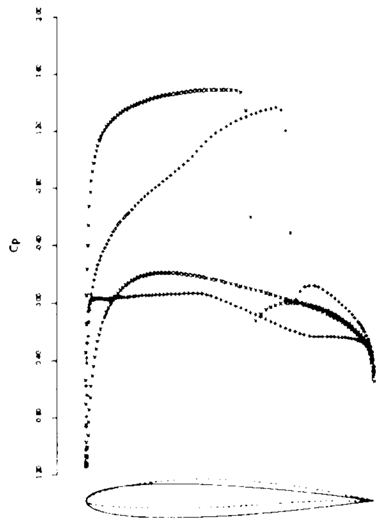
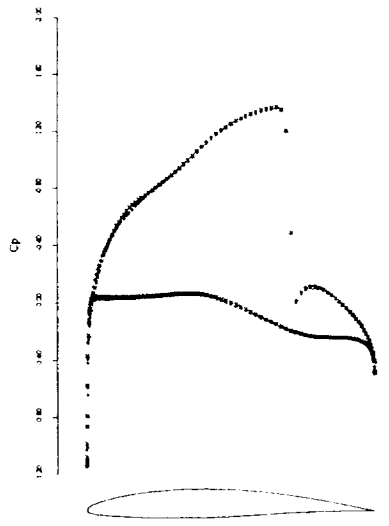


Figure 8.2: *Comparison of Finite Difference vs. Adjoint Gradients for Potential Flow Design Method. Test Case 2.*
Design Variables Defined in Table 1
Average Residuals: FLO42 — 10^{-9}, ADJ42 — 10^{-3}.
Stepsize = 0.0001. Variable Scaling = 0.01.

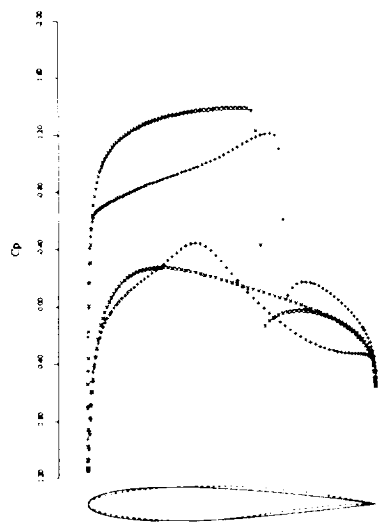


8.3a: *Initial Condition, I = 42.275*
$C_l = 0.7242$, $C_d = 0.0246$, $\alpha = 2.147°$

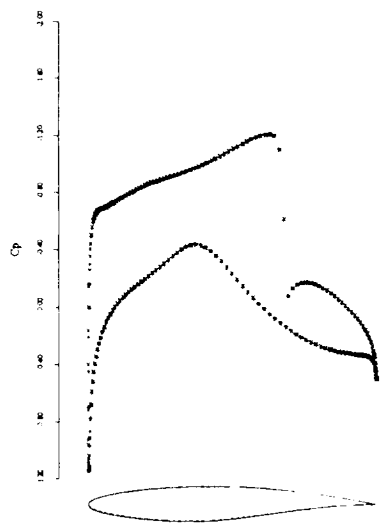


8.3b: *26 Design Iterations, I = 0.032*
$C_l = 0.7260$, $C_d = 0.0094$, $\alpha = 0.012°$

Figure 8.3: *Case 1:* $M = 0.735$, ***Fixed Circulation Mode.*** *Inverse Design*
—, × *Initial Airfoil: NACA 0012.*
- - -, + *Target* C_p*: NACA 64A410,* $M = 0.735$, $\alpha = 0.0°$.

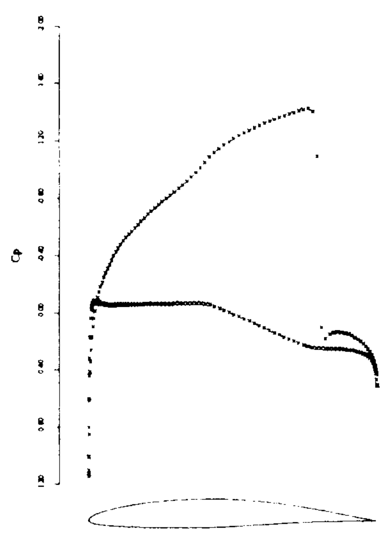


8.4a: *Initial Condition, I = 20.781*
$C_l = 0.6429$, $C_d = 0.0227$, $\alpha = 2.097°$

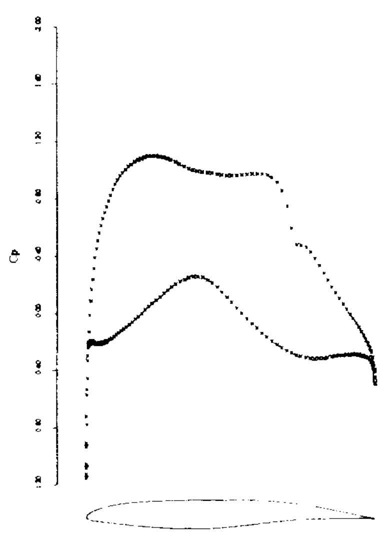


8.4b: *45 Design Iterations, I = 0.032*
$C_l = 0.6444$, $C_d = 0.0048$, $\alpha = 1.069°$

Figure 8.4: *Case 2:* $M = 0.75$ ***Fixed Circulation Mode.*** *Inverse Design*
—, × *Initial Airfoil: NACA 0012.*
- - -, + *Target* C_p*: RAE 2822,* $M = 0.75$, $\alpha = 1.0°$.



8.5a: *Initial Condition, I = 10.189*
$C_l = 0.7927$, $C_d = 0.0204$, $\alpha = -0.367°$



8.5b: *4 Design Iterations, I = 0.507*
$C_l = 0.7939$, $C_d = 0.0010$, $\alpha = 0.405°$

Figure 8.5: *Case 3:* $M = 0.75$, ***Fixed Circulation Mode. Drag Minimization Design. 18 Hicks-Henne Design Variables.***
—, × *Initial Airfoil: NACA 64A410.*

Chapter 9

RESULTS FOR EULER EQUATIONS DESIGN METHOD

This chapter presents the results for the Euler design method. The results shall be grouped into four distinct sections: convergence characteristics, gradient comparisons, test cases using Hicks-Henne functions, and test cases using B-spline control points. It is hoped that these results, combined with those presented in the previous chapter, will adequately demonstrate the utility and efficiency of adjoint-based design methods. The primary test case that is used for exercising the Euler design method will be analogous to test Case 2 used in the potential flow method (it is also Case 3 for the Euler formulation to be presented in Section (9.3)): starting from the NACA 0012 airfoil, and by using the difference between the actual and target pressures as the objective, the design method drives the profile shape toward the RAE 2822 airfoil operating at Mach = 0.75 and $\alpha = 1°$. Fifty design variables of the Hicks-Henne type are chosen as described in Tables (B.1) and (B.2) of *Appendix B1*. As with the potential flow formulation, this represents a good check case since it features proven design variables, a problem with a unique minimum, and the challenges of transonic flow.

9.1 CONVERGENCE

The convergence for the Euler state and co-state systems is shown in Figure (9.1). The iterations refer to complete multigrid W-cycles through 4 meshes (193x33 being the finest). One Runge-Kutta-like multistage time step is taken on each mesh in the sequence. For this test case the flow solver again shows a better convergence rate than the adjoint solver. The flow solver is started from uniform flow while the adjoint solver is started by zeroing out all adjoint variable components within the domain. It will be the topic of further research to improve convergence acceleration for the co-state system. While in the present case the same multigridding and residual smoothing algorithms employed in the state system are used to enhance the convergence rate for the co-state system, it is quite possible that significant retuning of these techniques could yield improvements. Furthermore, no attempt was made in the present work to construct the analogue to the enthalpy damping employed in the state system for use in the co-state system. Such an analogue, or yet unexplored techniques, could significantly enhance the adjoint solver convergence performance. One obvious difference between the two systems is the fact that the co-state system is linear while the state system is nonlinear. Thus it may be possible to develop iteration schemes that take advantage of this linear property to enhance convergence. The choice of the artificial viscosity to stabilize and smooth the discrete solution was constructed for the co-state system as a simple extension of that used for the state system. While the choice of these artificial terms may impact the gradient accuracy, alternative choices may be beneficial to the convergence rate.

9.2 GRADIENT COMPARISONS

9.2.1 FINITE DIFFERENCE GRADIENTS

To validate the adjoint-based design method using the Euler equations, a check of the gradient accuracies is of considerable importance. The first step in this validation is to obtain very accurate gradients of the discrete system for comparison purposes. Here, the finite difference method will be used as the bench mark. Figures (9.2) and (9.3) show the sensitivity of the finite difference gradients to both the flow solver accuracy and the stepsize used in the differencing. These sensitivities were carried out on the gradient calculation for the initial point of test Case 3. While requirements for both the stepsize limit and the flow solver accuracy will change as the design proceeds, for the purposes of gradient comparisons it is sufficient to stay with this initial state. Figure (9.2) shows that with 64-bit arithmetic at least 4 orders of convergence in the flow solver are needed to render an accurate gradient for a step size of 0.0001. Figure (9.3) shows that an acceptable stepsize to avoid discretization error for this initial state is 0.1. Thus by using a convergence tolerance of 10^{-9} and a stepsize of 0.0001 we are assured of a highly accurate approximation to the gradient of the discrete system.

9.2.2 ADJOINT GRADIENTS

With this accurate gradient in hand a similar analysis is performed for the adjoint approach. In this case the sensitivity to the stepsize (to which the adjoint approach is insensitive) is replaced by the sensitivity to the adjoint equation convergence level. The boundary condition for the adjoint equation that is chosen is the *Type 3* boundary condition described in *Chapter 7* while the domain fluxes are calculated with the more consistent *Type 2* flux routines also described in *Chapter 7*. Figures (9.4) and (9.5) respectively show the sensitivity of the gradient to the flow and adjoint solver convergence levels. What is striking from these plots is that *the gradient for the adjoint method is quite insensitive to the convergence level of either the flow or the adjoint systems*. For the limiting case for both systems (i.e. where the average residual is 10^{-2} for the flow, and 10^{+0} for the adjoint [1]) where very slight discrepancies in the gradients are just appearing, only 10 multigrid cycles have been completed for each system. The same level of accuracy is not obtained for the finite difference gradient method until 10^{-7} is achieved in average residual of the flow solver. This indicates a 5 orders of magnitude less stringent convergence requirement for the flow solution when using the adjoint-based gradient. To some extent it would be possible to increase the stepsizes used for the finite difference method since 0.0001 was very conservative. Remember that 0.1 was shown to be absolutely necessary to maintain gradient accuracy. This increase in stepsize would have the effect of reducing the

[1] Note the residual for the adjoint is normalized by the maximum value of ψ in the converged field. This was necessary since scaling of the boundary condition introduced to scale the cost function for optimization purposes also scales the residual and the ψ because the system is linear.

convergence requirements. However as the minimum is approached, and the norm of the gradient becomes smaller, this larger stepsize could cause problems. The adjoint-based gradients would not be affected in any way by these changes since they are entirely independent of stepsize. Here, the value of 0.0001 was chosen since in practice it was shown to be a sufficiently conservative value for the finite difference method even when the design was close to the optimum and the norm of the gradient was very small. These convergence trends are consistent with the error analysis that was explored in Section (3.7). The importance of the significant difference between the convergence requirements for the two methods cannot be overstated. It implies, first, that the gradients for the adjoint method will be cheaper not only from the vast reduction of the required number of equivalent flow analyses (2 vs $\bar{n} + 1$) but also because of the drastic reduction in computational cost for these 2 solutions due to nonstringent convergence requirements. Secondly, it implies that design methods that do not involve multiple accurate flow field evaluations during a line search (now the most expensive part of the design procedure) should be explored with renewed interest. To underline this result, the computational cost of the adjoint-based gradient with the reliable levels of average residual set to 10^{-4} for the flow, and 10^{-1} for the adjoint, was 8.2 CPU seconds while the cost for the finite difference method using a stepsize of 0.0001 and a flow solver average residual of 10^{-7} was 59.3 CPU seconds. This even includes the fact that for the finite difference method, the flow solver was restarted from a previously converged solution and thus required only a few additional multigrid iterations for each component of the gradient. The final result represents a 7.2 fold reduction in computational time for the adjoint-based gradients. This must be viewed as a very conservative estimate of the level of improvement that can be realized since considerably less-converged flow and adjoint solutions would reduce the CPU time for the adjoint-based gradient further. Furthermore, the benefit in three dimensions would be even more marked becuase the number of design variables would typically be greater.

9.2.3 GRADIENT COMPARISONS

In Figure (9.6) a straight comparison is made between the gradients for the two methods. For this comparison the stepsizes and convergence tolerances were set to the values defined above for the CPU time comparisons. The adjoint method used *Type 2* flux routines and *Type 3* boundary conditions. The result shows that there is a slight discrepancy for the adjoint approach despite the work that was done to institute improved wall boundary conditions and domain discretizations. From the development of the method it is clear that there are many areas where discrepancies between the current implementation and that of maintaining strict discrete consistency still exist. Examples of these differences are evident in the treatment of the discrete dissipation terms. Both in the matrix transpose operation, and in the surface and volume integrals that form the gradient, these dissipation terms have been neglected. As was seen in the potential flow discretization, small errors can be tolerated within the gradient calculations without affecting the design process significantly. The error level that is apparent in Figure (9.6) is far smaller than that shown in Figure (8.2) for the potential flow formulation. To understand this improvement, an examination of the alternate adjoint equation flux and wall boundary condition routines is explored.

9.2.4 BOUNDARY CONDITIONS AND FLUXES

Figure (9.7) shows a comparison of different adjoint wall boundary conditions all using the *Type 2* flux routines. For these comparisons the very tight convergence tolerances of 10^{-9} for the flow and 10^{-5} for the adjoint were used to ensure consistency. It is seen that the original *Type 1* boundary condition gave a rather poor estimate of the gradient. This was the original wall boundary condition that was used in our early work [65], and yet even then we were able to obtain working design methods. The *Type 2* and *Type 3* boundary conditions both vastly improve the gradient. No distinct difference is seen between these last two, leading to the conjecture that the discrepancies still present in the adjoint-based gradient are probably not associated with the wall boundary condition. Turning to comparisons for the flux routines where the wall boundary condition is fixed as *Type 3*, Figure (9.8) shows a comparison for the *Type 1* vs. *Type 2* fluxes. Although not as dramatic as with the change in the wall boundary conditions, a small but significant improvement is seen by resorting to the *Type 2* fluxes. It must be remembered that the *Type 2* fluxes involve a significant increase in the operational count over the *Type 1* fluxes. It will be the work of future research to determine how accurately these gradients should be calculated. In all of the test cases to follow, the *Type 3* boundary conditions and the *Type 2* fluxes will be used.

9.3 Hicks-Henne Test Cases

Several test cases are conducted with the Euler design method. Additional test cases are provided in References [65, 94] which were presented during the course of this larger body of research. To show the robustness of the method, most of the solutions are obtained using the same set of design variables. Further, no smoothing or shock wave unweighting of the boundary condition for the adjoint solver was used in any of these results with the exception of one case. It is gratifying that the adjoint solver remained stable in the presence of the strong discontinuities that are apparent in the adjoint equation boundary conditions. Unless otherwise stated, all of the target pressure distributions used to drive the design problems were calculated for known airfoils with the same basic flow solver (FLO82) as that used in the design. Furthermore, as with the potential flow method, the dissipation coefficients were kept identical between the analysis and the design so that even the shock structure remained realizable. However, unlike the potential flow cases, the inverse design cases here all involve design at a constant α. To facilitate this procedure, the airfoils obtained through various sources were rotated and renormalized such that both their leading and trailing edges were located at $y = 0$. As a final point it is noted that in many of the cases to follow, the design method was terminated once changes in the cost function became negligible.

9.3.1 INVERSE DESIGN

The initial and final results of Test Case 1 for the Euler design method are shown in Figure (9.9). The NACA 0012 airfoil is modified to achieve the pressure distribution of a NACA 64A410 airfoil at Mach 0.735 and $\alpha = 0°$. Fifty Hicks-Henne design variables distributed around the airfoil and specified in Tables (B.1) and (B.2) are used to modify the shape. The design is run in fixed alpha mode with the final airfoil and pressure distribution almost exactly matching the target. As was seen in the potential flow cases, details of the shock structure have been matched. The corresponding Case 1 for the potential flow algorithm displayed some discrepancies in the final pressure distribution; no such perceptible discrepancies are apparent here for the Euler method.

Test Case 2 is an inverse mode design and is displayed in Figure (9.10). This time, the NACA 0012 airfoil is driven towards the target pressure distribution of the GAW 72 airfoil operating at Mach 0.70 and $\alpha = 0°$. Under these conditions the target airfoil displays a very strong shock, yet the design method is able to converge to the desired shape without visible discrepancies using the 50 Hicks-Henne design variables of Tables (B.1) and (B.2). Once again even the shock structure was matched to visual resolution. If the target solution here were either more smeared or more crisp it would obviously become an unrealizable target in terms of the dissipation that was held constant during the design. It is noted that in such a case the strong oscillation that would appear in the boundary forcing term for the adjoint equation in and around the shock(s) could force the employment of either smoothing or shock-wave unweighting in the boundary condition definition.

Test Case 3 also uses the same set of design variables and the NACA 0012 as a starting condition, as illustrated in Figure (9.11). The target pressure distribution is from the RAE 2822 airfoil at $\alpha = 1°$ and Mach 0.75. Again the design method converges to the target, almost exactly matching even the shock position and strength. Figure (9.11) also shows the progression during the design process toward the minimum. Of particular note in the two intermediate solutions is the fact that both the airfoil surfaces and the resulting C_ps retain a very smooth character with this choice of design variables.

Test Case 4 represents a greater challenge. The Korn airfoil at Mach 0.75 and $\alpha = 0.175°$ is chosen for the target pressure. The challenge is presented by the fact that the Korn airfoil at this condition has shock free supercritical flow with virtually no wave drag. Figure (9.12) shows the NACA 0012 airfoil being driven towards the desired target, with the design method employing 54 Hicks-Henne design variables as per Tables (B.3) and (B.4). The increase in the number of variables was necessary in order to give the slightly greater freedom in the design space that this problem required. As the final design is approached there is a tendency to produce a double shock pattern instead of a smooth recompression. It appears that the design space for this problem is more nonlinear than in the previous test cases. The final design is very close, but does show some discrepancies from the desired pressure distribution. The remaining discrepancies, which are not visually detectable in the airfoil shapes, are still probably related to an incomplete design space since this shock-free case is very delicate.

In Test Case 5 the RAE 2822 airfoil is revisited for a target pressure distribution. However, the potential flow solver (FLO42) was used to provide the target pressure distribution. Thus this pressure distribution is not realizable by the Euler equations because the shock wave of the target is modeled as an isentropic jump. It is interesting that this proved to be the only case in which the shock wave unweighting was found to be necessary. Results without the unweighting achieved a solution that did not come very close to matching the target pressure distribution even outside of the shock region. The final results using the unweighting with 50 design variables (Tables (B.1) and (B.2)), shown in Figure (9.13), very closely approximates the desired pressure distribution, but of course does not match it exactly. An examination of the final airfoil reveals a striking difference between it and the target airfoil. It can be seen that for such strong shock cases, the potential flow equation can give quite incorrect results.

9.3.2 DRAG MINIMIZATION

In two of the drag reduction cases to follow the fixed lift mode of the design procedure is exercised. This or some other method of retaining lift is necessary since minimizing drag at a fixed alpha would simply remove the drag due to lift by reverting to zero lift. However, it turns out that even specifying a large lift coefficient as an implicit constraint and drag as the cost function to be minimized does not necessarily lead to sensible airfoil shapes.

Test Case 6 minimizes drag at fixed lift. The NACA 64A410 airfoil at Mach 0.75 and $C_l = 1.000$ and $C_d = 0.0416$ is used as a starting condition. Figure (9.14) illustrates that by choosing 18 Hicks-Henne design variables, Table (B.5), which modify the camber, the optimization procedure is able to reduce the drag to just 1 count in 13 design cycles. This is accomplished while the lift coefficient, Mach number and thickness distribution remain unchanged. Note that the upper surface pressure distribution is both supercritical and shock free. Even though this newly designed airfoil displays an enormous amount of lift for a transonic zero drag design it is highly doubtful that it would ever be used in practice. In particular the gradient of the upper surface pressure distribution near the trailing edge is probably too steep. It is quite likely that such an airfoil would experience flow separation and shocks in viscous flow. Also, since an airfoil with essentially zero drag has been designed for this lift, thickness and Mach number, there is nothing that precludes the existence of many such airfoils. The design code presumably happened to find one of an infinite number of such airfoils. The conclusion from this design case is that the design problem should be more precisely specified by the addition of more requirements beyond the minimization of drag at a given C_l.

In Test Case 7, drag is again used as the objective function except that it is augmented with a target pressure distribution to form a combined objective function. Since the target pressure distribution indirectly specifies a target C_l, the design code is once again run in constant alpha mode. In this example case the target pressure distribution from Test Case 3 is augmented by the drag coefficient to form a combined cost function. The design variables are prescribed as in the earlier case from Tables (B.1) and (B.2). Figure (9.15) illustrates that after 19 design cycles the drag has been reduced from 28 counts for the initial NACA 0012 airfoil at $\alpha = 1^\circ$ to 7 counts. It should be noted from Case 3 that had the target been achieved precisely, the drag coefficient would have been 45 counts. The resulting airfoil looks very like the RAE 2822 airfoil and in fact matches its pressure distribution over much of its surface. The pressure distribution in the region near and ahead of the shock is of course quite different to accommodate the significantly weaker shock.

In the final example using the Hicks-Henne functions (Test Case 8) we once again return to the design for minimum drag at a fixed lift. Here we try to recamber the NACA 64A410 airfoil so that it will operate at a high lift coefficient and Mach number. The design is carried out at Mach = 0.75 and $C_l = 0.8$. However, unlike Case 6, a shock-free target pressure distribution from the Korn airfoil is now specified over the majority of the upper surface in addition to C_d as part of the cost function. The advantage is that the pressure distribution of the Korn airfoil does not display the drastic pressure recovery that was obtained in design Test Case 3. Since the Korn airfoil achieves this pressure distribution at $C_l = 0.62$ the new design should be a higher lift Korn-like airfoil. Furthermore, because we are choosing 24 Hicks-Henne design variables (Table (B.6)) that modify only camber, the thickness distribution will still be that of the NACA 64A410. Figure (9.16) shows that the final design has dropped the drag from 230 counts to 3 counts in 20 design iterations. In addition the upper surface pressure distribution looks much like that of its target, the Korn distribution. Hence it is expected that this design would be far more practical than the one designed in Case 6.

9.4 B-Spline Test Cases

The last four test cases presented in this research explore the use of B-spline control points as design variables. The main emphasis of these examples will be to compare the effectiveness and reliability of these design variables with that achieved in the previous two sections by the Hicks-Henne functions. To make this comparison as fair as possible some of the same test cases attempted with the Hicks-Henne functions will be repeated using the B-spline control points. Since typical CAD systems use B-spline surfaces to represent the geometry, B-splines were used directly to represent the initial as well as subsequent designs.

The first example (Test Case 9) will be a repeat of Test Case 1. The goal is to perform an inverse design at fixed alpha and Mach number. Recall that the initial starting point was a NACA 0012 airfoil operating at $\alpha = 0°$ and Mach = 0.75. The target pressure distribution is specified as that of a NACA 64A410 airfoil operating at the same alpha and Mach number. Eighteen B-spline control points on each of the upper and lower surfaces were used to define a close approximation to the starting NACA 0012 airfoil. To ensure a smooth and continuous airfoil throughout the design, the degree of the B-spline curve was set to 4, the control points at both the leading edge and trailing edges remained fixed, and the points on either side of the leading edge were forced to move in unison. This left 31 of the 36 control points as design variables. These were allowed to move freely in the y coordinate direction only. The locations of these control points are provided in Table (B.7). The final design after 40 iterations is shown with starting point in Figure (9.17). Just as when the Hicks-Henne functions were used, the B-spline design space was able to fully recover both the pressure distribution and the airfoil shape. A notable difference between this case and Case 1 is that in Figure (9.17) there is a slightly perceptible oscillation in the final pressure distribution while Figure (9.9) shows no such oscillation. The small differences between the number of design iterations and the final values of the cost functions are not considered significant since they may be problem specific. However, since such a good result was obtained here with only 31 design variables as opposed to 50 for the Hicks-Henne case, some advantage appears to be offered by the B-spline control points.

Test Case 10 essentially repeats Test Case 2 with much of the same discussion for Test Case 9 applying here. The reader is referred to the discussion of Test Case 2 for the specific design goals and conditions. The design variables were the same 31 variables described for Test Case 9 and listed in Table (B.7). Again there is a noteworthy visible oscillation in the final pressures on the forward upper surface in Figure (9.18).

In Test Case 11 the design of Test Case 3 is repeated with the 31 B-spline control point design variables listed in Table (B.7). Figure (9.19) shows two interim solutions in addition to the initial and final solutions. A comparison between these plots and those in Figure (9.11) shows that the interim solutions are much more oscillatory for the B-spline control points than they are for the Hicks-Henne functions. Further, the reduction in the cost function for the B-splines is not nearly as good at iterations 5 and 10 even though the cost function for the final solution is actually better than that of the Hicks-Henne case. One possible cause of these oscillations is that the B-spline control points may localize the design variables to a point that they act more like using the mesh points themselves as design variables. Thus the high frequency information contained in the control points can easily cause oscillations in the design, with the result that there is a greater degree of nonlinearity and a poorer conditioning of the design space as compared with that for the Hicks-Henne functions. Although they are not presented here, similarly poor interim solutions were evident in Test Cases 9 and 10. Further tests using the B-spline control points indicated that the number of control points determined the extent to which these oscillations in the interim designs existed. When the number of control points and hence the degree of the design space was increased, the gross oscillations in the interim solutions also increased. In extreme cases the design process stopped far from the high quality solutions presented here. One possible cause of the failure is that a local minimum in a multimodal design space was reached. For a modest number of control points, say 10 per surface, the severity of the oscillation in the interim solutions was greatly reduced. However, such a small number of control points far from guarantees that achievable pressure distributions would be attained. A prudent step could be to use a low-pass filter when using B-spline control points as design variables, so as to smooth the control in a manner similar to that employed by Jameson [54, 55, 59]. Even with such drastic interim solutions it was gratifying that the final solution in Figure (9.19) actually did better in terms of the cost function when compared with Test Case 3.

The final Test Case (12) in this thesis repeats Test Case 4 using the B-spline control points listed in Table (B.7). In this difficult shock-free case the B-spline control points were able to achieve better final results than in Case 4 for the Hicks-Henne functions. It seems from this example as well as the others presented in this section that, for a given number of design variables, the B-spline control points indeed gives a larger design space than the Hicks-Henne functions. However this greater geometric flexibility comes at the possible price of a greater high frequency content in the design space that may in turn increase the likelihood of a multimodal design space. Although this study of design variables is far from comprehensive it gives an introductory example of the problems that must be explored with respect to the design space parameterization.

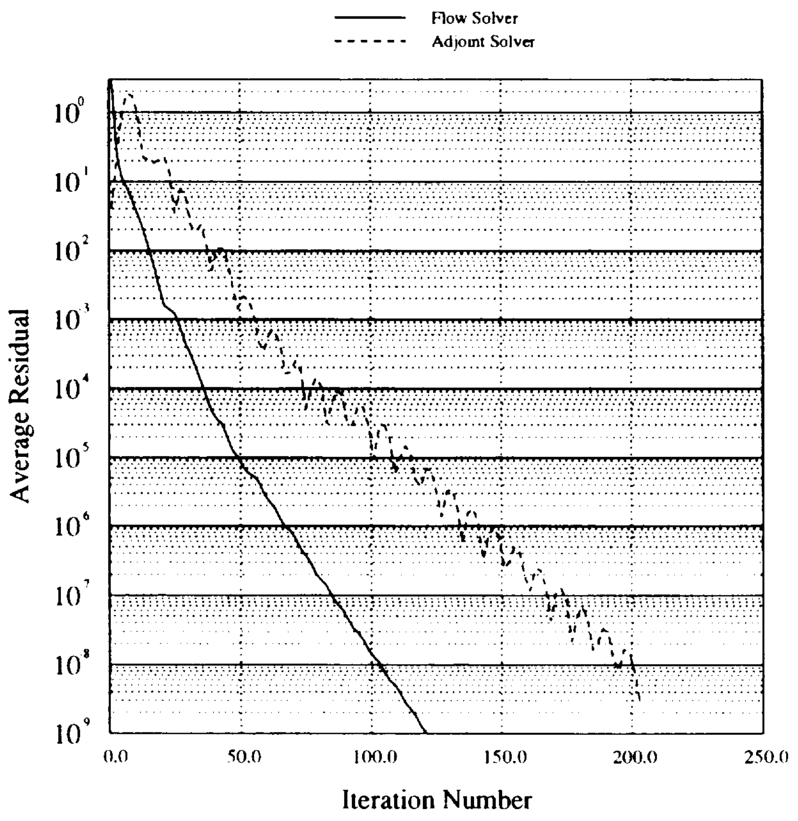


Figure 9.1: *Convergence Rates in Terms of Average Residual for FLO82 and ADJ82. Mesh = 193x33. 4 Mesh Multigrid Cycle.*

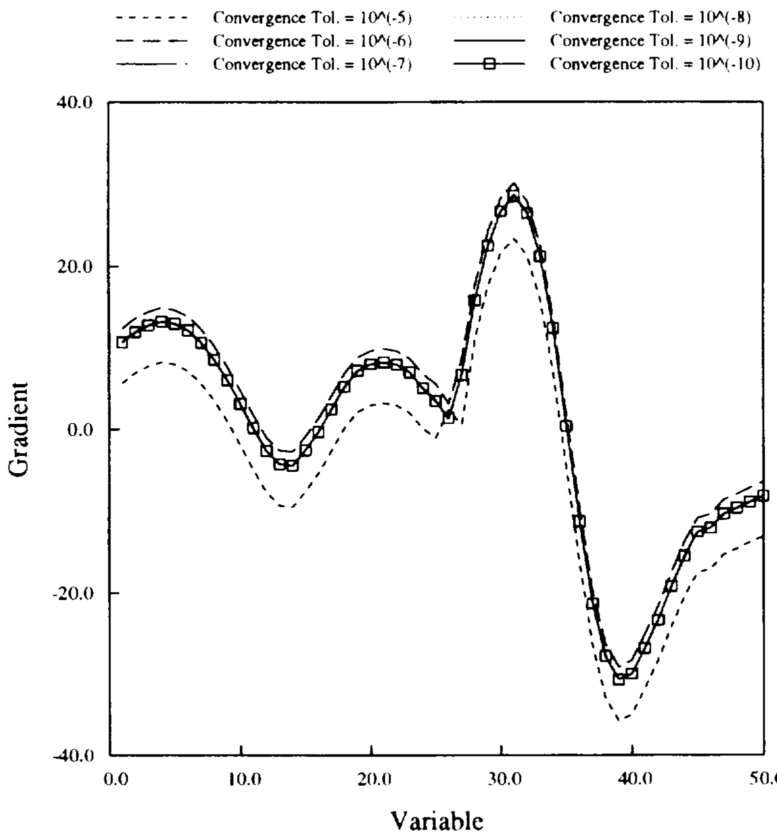


Figure 9.2: *Sensitivity of Finite Difference Method to Flow Solver Convergence Design Variables Defined in Table 1. Test Case 2.*
Stepsize = 0.0001. Variable Scaling = 0.01.

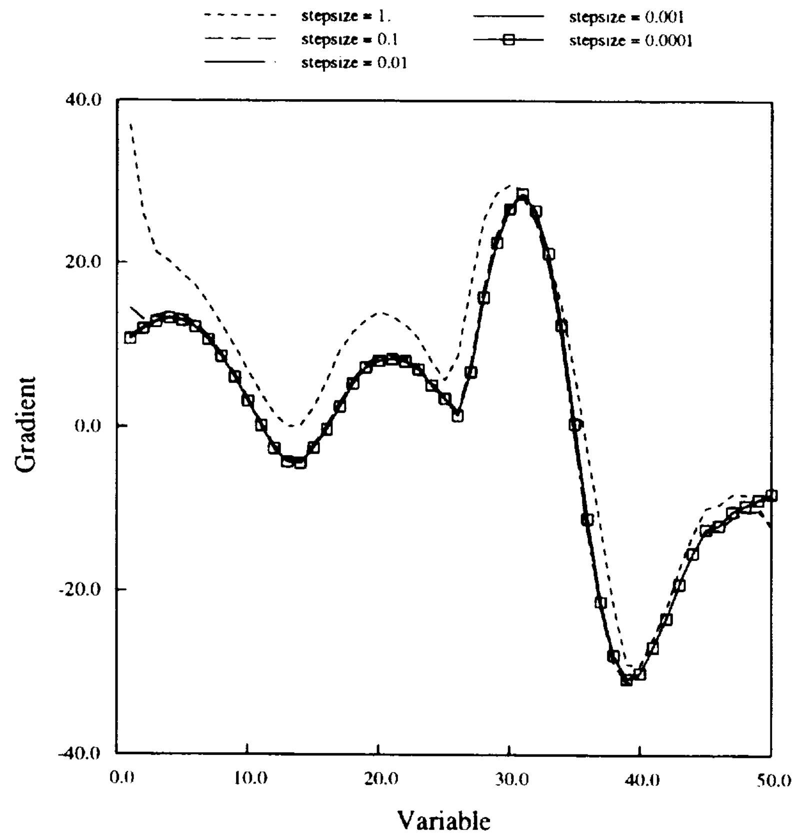


Figure 9.3: *Sensitivity of Finite Difference Method to Stepsize Design Variables Defined in Table 1. Test Case 2. Flow Solver Average Residual* = 10^{-9}. *Variable Scaling = 0.01.*

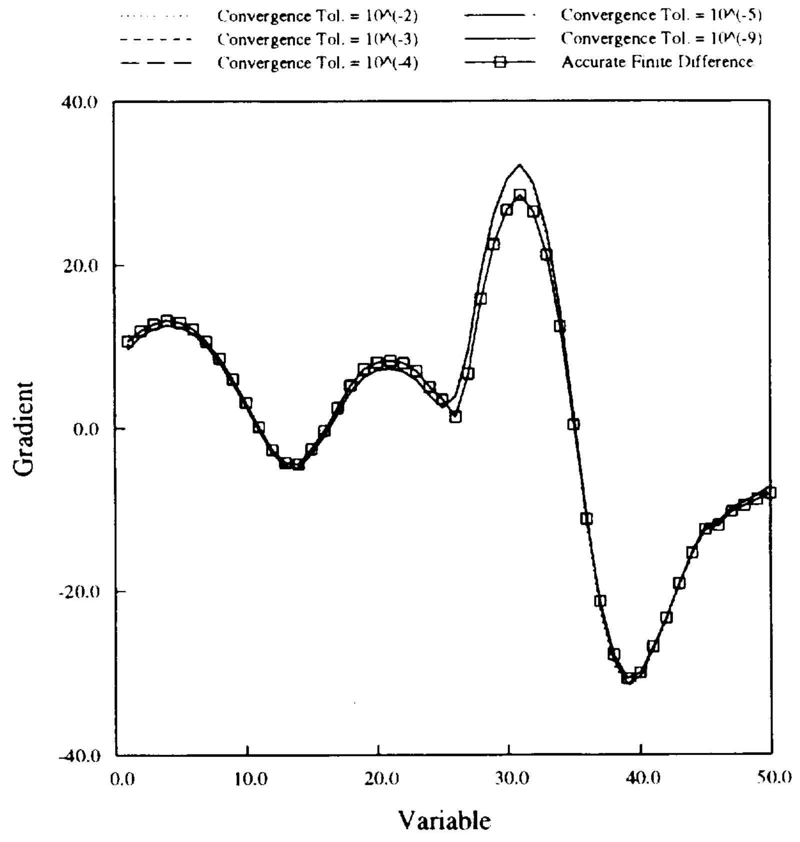


Figure 9.4: *Sensitivity of Adjoint Method to Flow Solver Convergence Design Variables Defined in Table 1. Test Case 2. Stepsize = 0.0001. Variable Scaling = 0.01.*

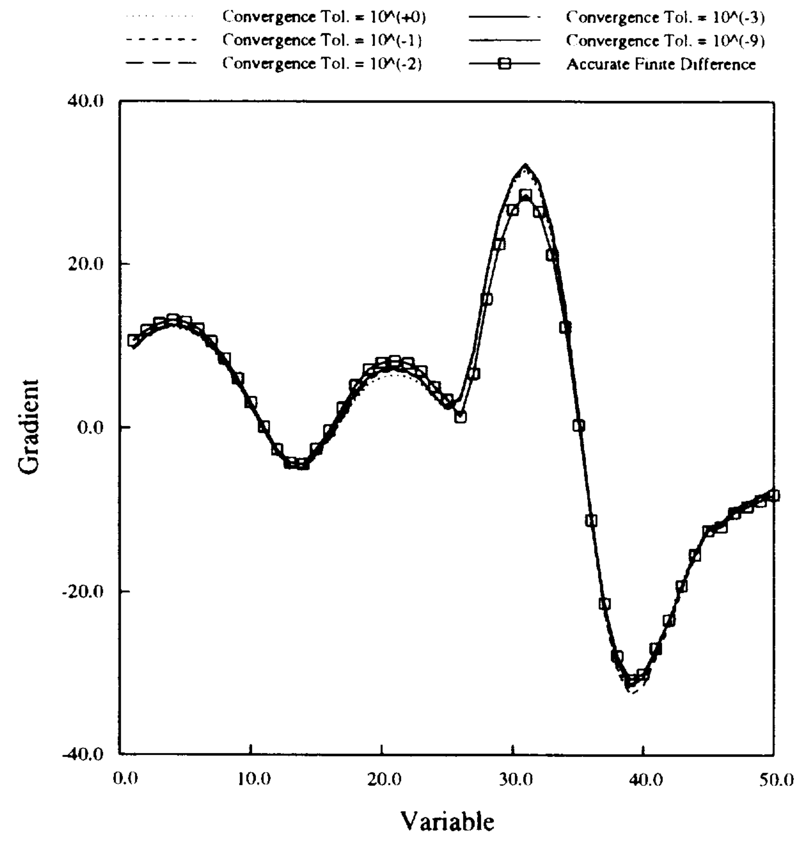


***Figure 9.5:** Sensitivity of Adjoint Method to Adjoint Solver Convergence Design Variables Defined in Table 1. Test Case 2.*
Stepsize = 0.0001. Variable Scaling = 0.01.

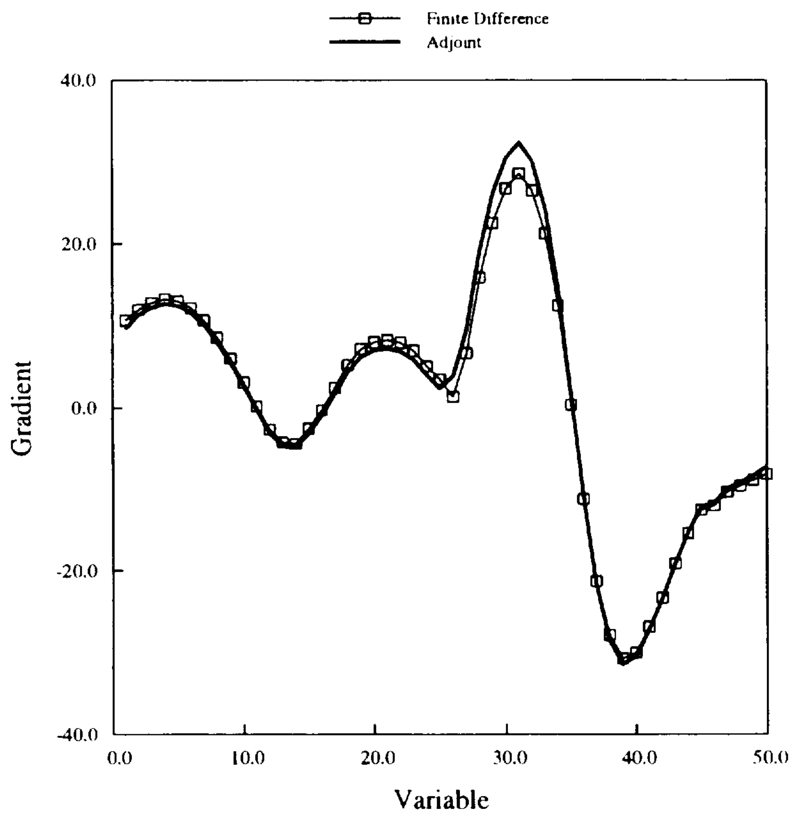


***Figure 9.6:** Comparison of Finite Difference and Adjoint Gradients Design Variables Defined in Table 1. Test Case 2. Euler Formulation. Flow Solver Average Residual* = 10^{-9}.
Adjoint Solver Average Residual = 10^{-9}.
Stepsize = 0.0001. Variable Scaling = 0.01.

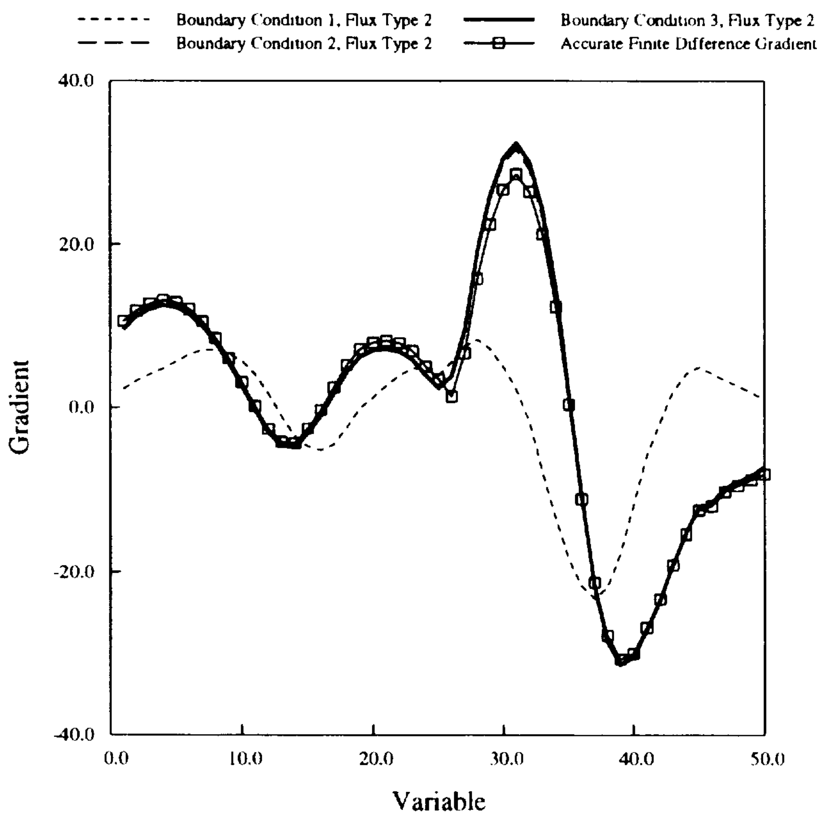


Figure 9.7: *Comparison of Adjoint Gradients with Different Wall Boundary Conditions.*
Design Variables Defined in Table 1. Test Case 2. Euler Formulation.
Flow Solver Average Residual = 10^{-9}.
Adjoint Solver Average Residual = 10^{-9}.
Stepsize = 0.0001. Variable Scaling = 0.01.

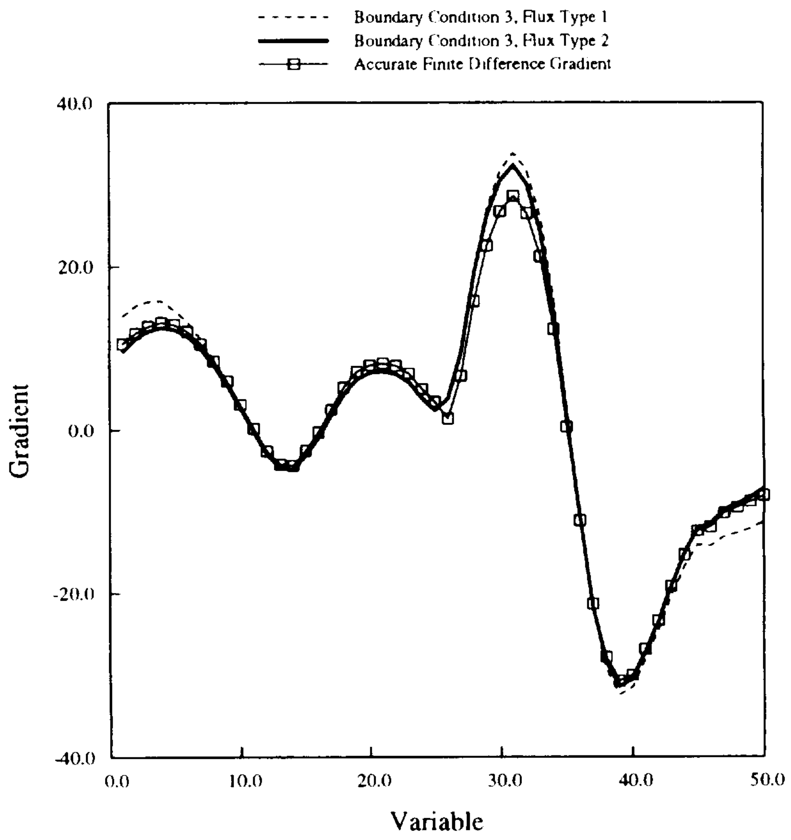
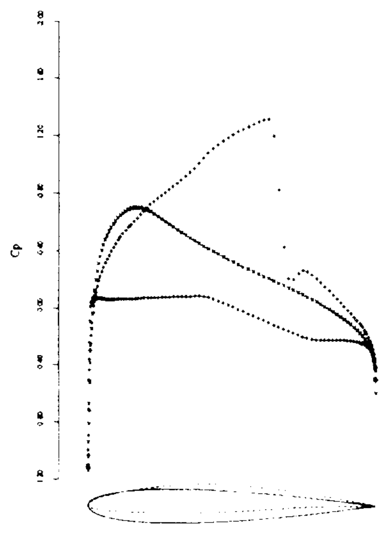
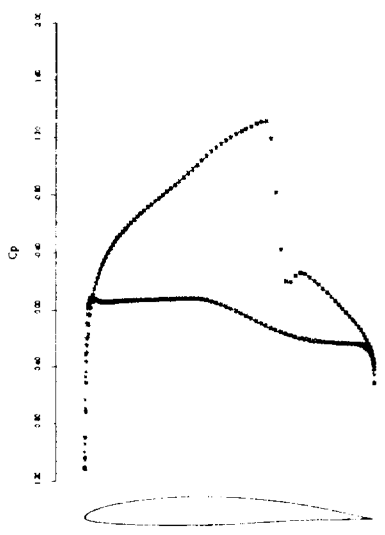


***Figure 9.8:** Comparison of Adjoint Gradients for Different Flux Routines Design Variables Defined in Table 1. Test Case 2. Euler Formulation.*
Flow Solver Average Residual = 10^{-9}.
Adjoint Solver Average Residual = 10^{-9}.
Stepsize = 0.0001. Variable Scaling = 0.01.

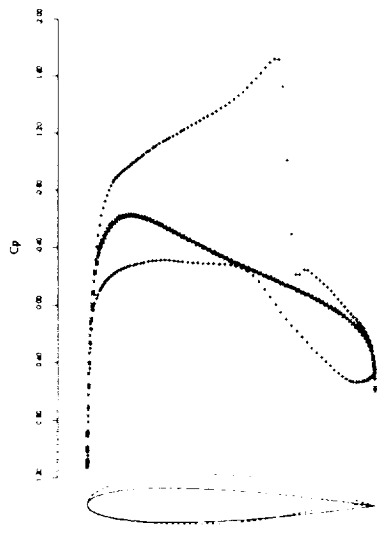


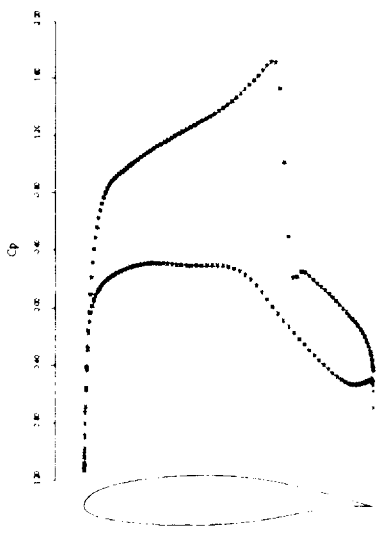


***9.9a:** Initial Condition,* I = 112.205
C_l = 0.0000, C_d = 0.0003, α = 0.0°

***9.9b:** 55 Design Iterations,* I = 0.040
C_l = 0.6476, C_d = 0.0076, α = 0.0°

***Figure 9.9:** Case 1:* M = 0.735, *Fixed Alpha Mode. Inverse Design, 50 Hicks-Henne Design Variables.*
—, × *Initial Airfoil: NACA 0012.*
- - -, + *Target* C_p*: NACA 64A410,* M = 0.75, α = 0.0°.

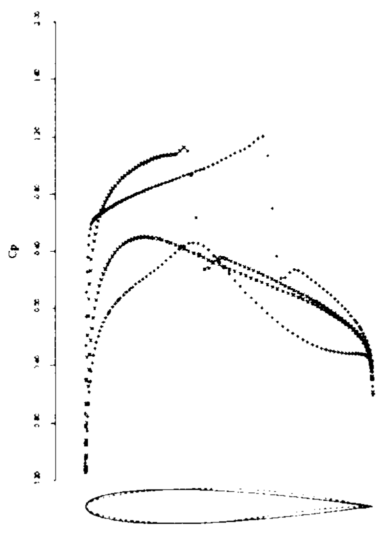




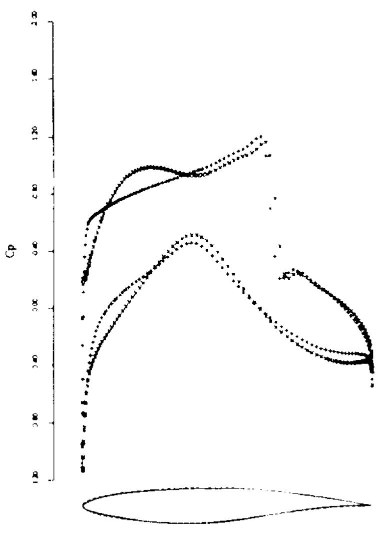
***9.10a:** Initial Condition,* I = 144.161
C_l = 0.0031, C_d = 0.0002, α = 0.0°

***9.10b:** 45 Design Iterations,* I = 0.025
C_l = 0.8132, C_d = 0.0157, α = 0.0°

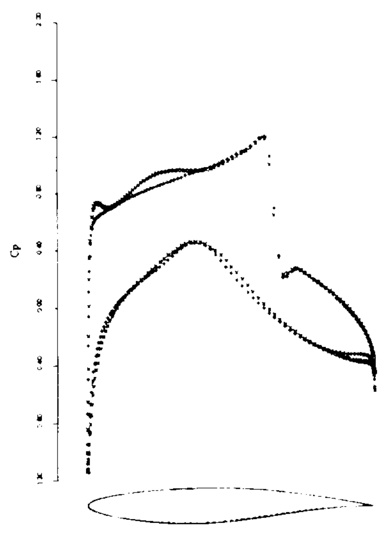
***Figure 9.10:** Case 2:* M = 0.70 *Fixed Alpha Mode. Inverse Design, 50 Hicks-Henne Design Variables.*
—, × *Initial Airfoil: NACA 0012.*
- - -, + *Target* C_p*: GAW 72,* M = 0.70, α = 0.0°.



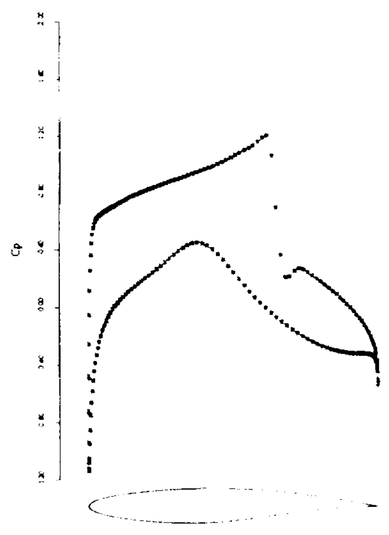
***9.11a:** Initial Condition, I = 68.556*
C_l = 0.2261, C_d = 0.0028, α = 1.0°



***9.11b:** 5 Design Iterations, I = 5.611*
C_l = 0.6549, C_d = 0.0036, α = 1.0°

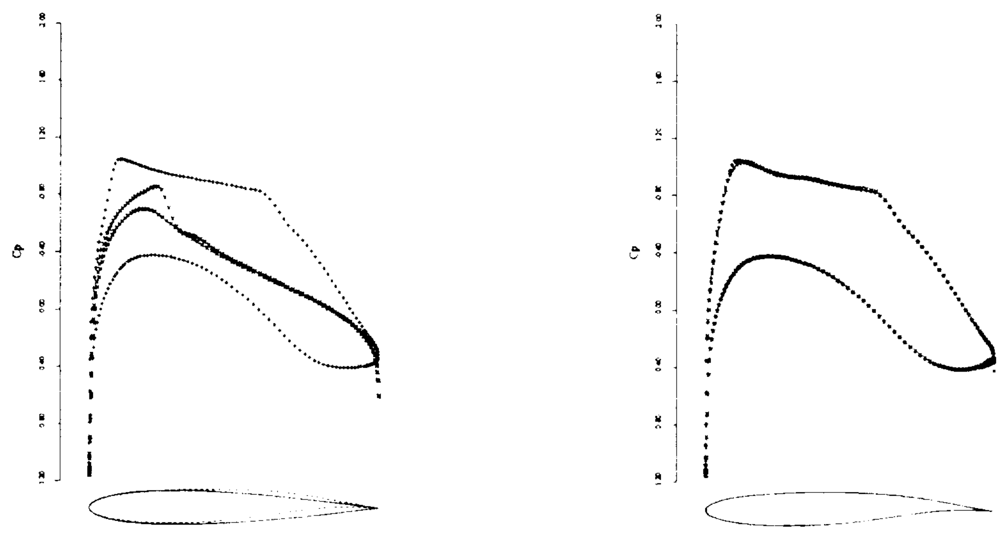


***9.11c:** 10 Design Iterations, I = 0.840*
C_l = 0.6189, C_d = 0.0035, α = 1.0°



***9.11d:** 35 Design Iterations, I = 0.008*
C_l = 0.6028, C_d = 0.0045, α = 1.0°

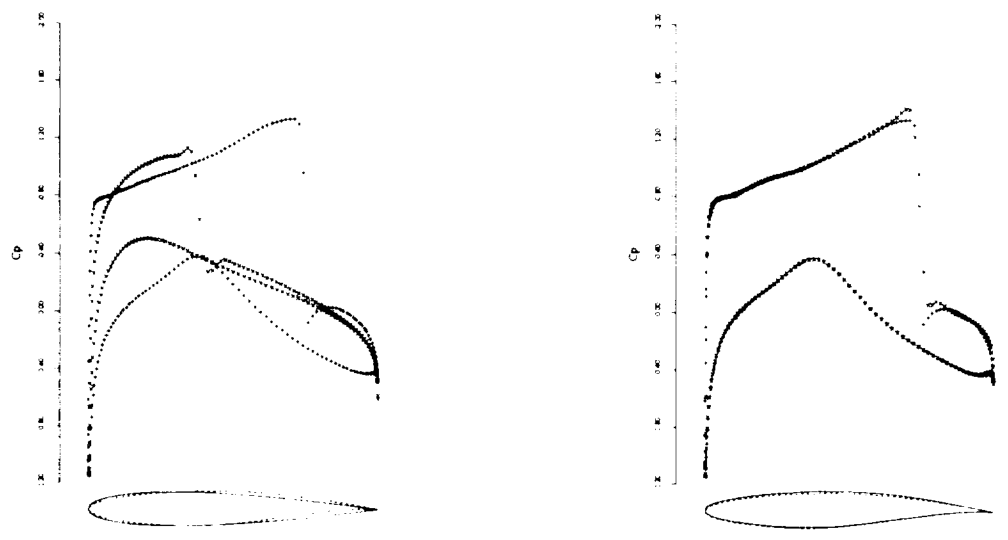
***Figure 9.11:** Case 3: M = 0.75, Fixed Alpha Mode. Inverse Design, 50 Hicks-Henne Design Variables.*
—, × *Initial Airfoil: NACA 0012.*
- - -, + *Target C_p: RAE 2822, M = 0.75, α = 1.0°.*



9.12a: *Initial Condition,* $I = 63.147$
$C_l = 0.0412$, $C_d = 0.0004$, $\alpha = 0.175°$

9.12b: *40 Design Iterations,* $I = 0.065$
$C_l = 0.6292$, $C_d = 0.0004$, $\alpha = 0.175°$

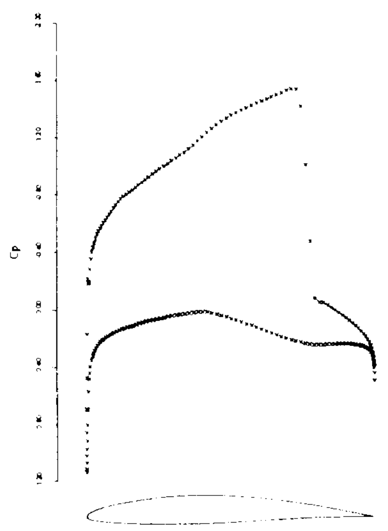
Figure 9.12: *Case 4:* $M = 0.75$, *Fixed Alpha Mode. Inverse Design, 54 Hicks-Henne Design Variables.*
—, × *Initial Airfoil: NACA 0012.*
- - -, + *Target* C_p*: Korn Airfoil,* $M = 0.75$, $\alpha = 0.175°$.



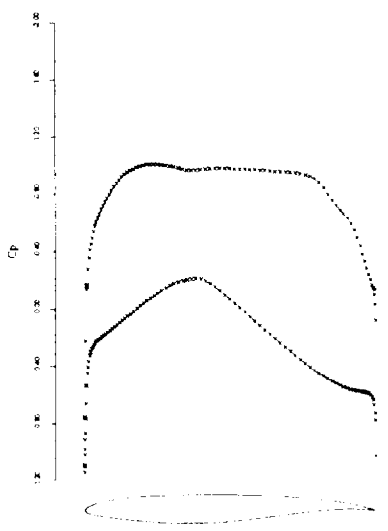
9.13a: *Initial Condition,* $I = 128.320$
$C_l = 0.2261$, $C_d = 0.0028$, $\alpha = 1.0°$

9.13b: *40 Design Iterations,* $I = 1.097$
$C_l = 0.8135$, $C_d = 0.0166$, $\alpha = 1.0°$

Figure 9.13: *Case 5:* $M = 0.75$, *Fixed Alpha Mode. Inverse Design, 50 Hicks-Henne Design Variables.*
—, × *Initial Airfoil: NACA 0012.*
- - -, + *Target* C_p*: RAE 2822 (Potential Flow Cp),* $M = 0.75$, $\alpha = 1.0°$.

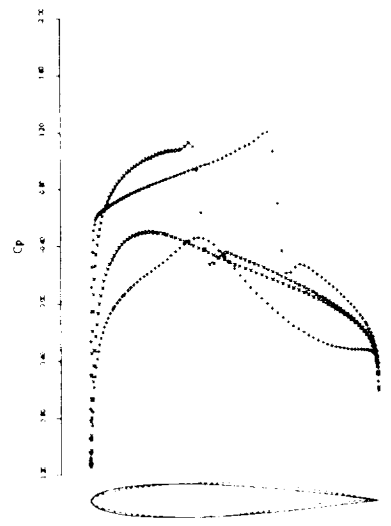


9.14a: *Initial Condition,* I = 20.809
C_l = 1.0000, C_d = 0.0416, α = 0.518°

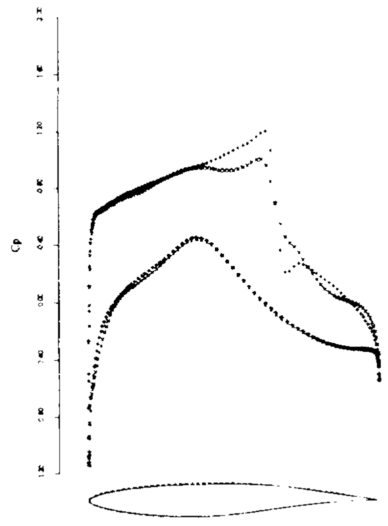


9.14b: *13 Design Iterations,* I = 0.029
C_l = 1.0000, C_d = 0.0001, α = −0.787°

Figure 9.14: *Case 6:* M = 0.75, *Fixed Lift Mode. Drag Minimization Design, 18 Hicks-Henne Design Variables.*
—, × *Initial Airfoil: NACA 64A410.*

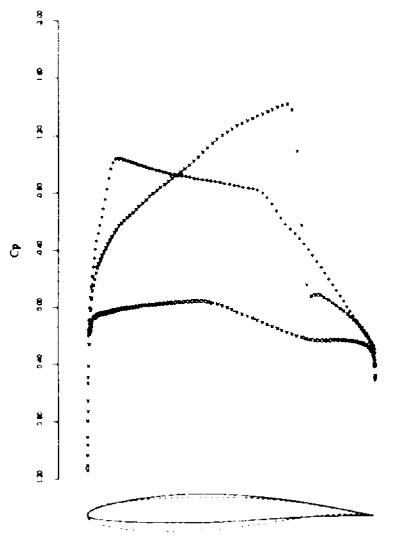


9.15a: *Initial Condition,* I = 79.881
C_l = 0.2261, C_d = 0.0028, α = 1.0°

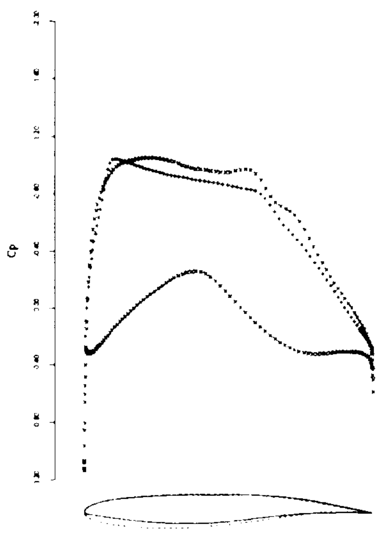


9.15b: *19 Design Iterations,* I = 5.446
C_l = 0.5937, C_d = 0.0007, α = 1.0°

Figure 9.15: *Case 7:* M = 0.75, *Fixed Alpha Mode. Drag Minimization + Inverse Design, 50 Hicks-Henne Design Variables.*
—, × *Initial Airfoil: NACA 0012.*
- - -, + *Target* C_p*: RAE 2822,* M = 0.75, α = 1.0°.

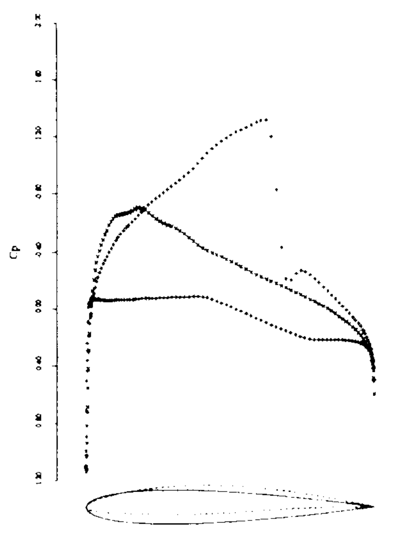


***9.16a:** Initial Condition,* I = 52.314
C_l = 0.8000, C_d = 0.0230, α = 0.708°

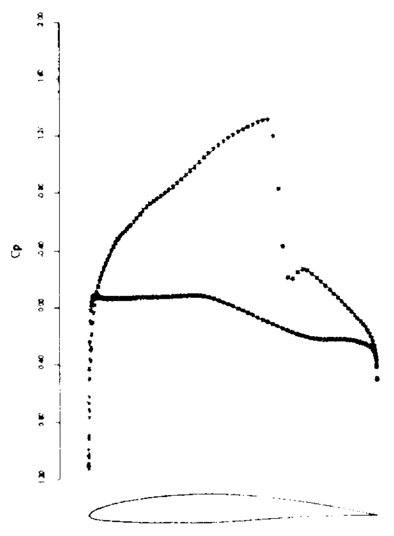


***9.16b:** 20 Design Iterations,* I = 3.312
C_l = 0.8000, C_d = 0.0003, α = 0.405°

***Figure 9.16:** Case 8:* M = 0.75, *Fixed Lift Mode. Drag Minimization + Inverse Design, 24 Hicks-Henne Design Variables.*
—, *Initial Airfoil: NACA 64A410.*
- - -, + *Target* C_p*: Korn,* M = 0.75, α = 0.174°.

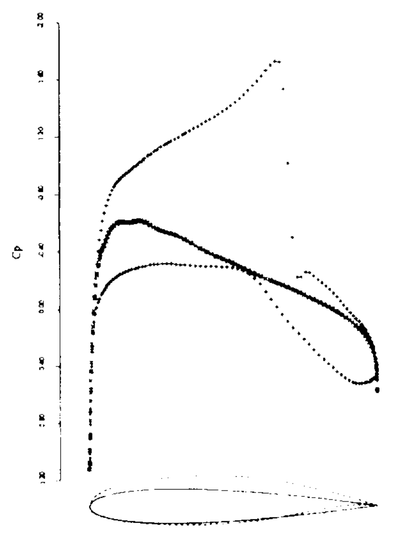


9.17a: *Initial Condition,* $I = 112.443$
$C_l = 0.0000$, $C_d = 0.0003$, $\alpha = 0.0°$

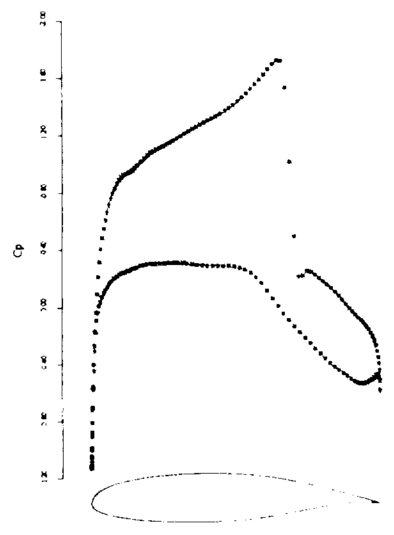


9.17b: *40 Design Iterations,* $I = 0.043$
$C_l = 0.6485$, $C_d = 0.0073$, $\alpha = 0.0°$

Figure 9.17: *Case 9:* $M = 0.735$, *Fixed Alpha Mode. Inverse Design, 31 B-Spline Design Variables.*
—, × *Initial Airfoil: NACA 0012.*
- - -, + *Target* C_p*: NACA 64A410,* $M = 0.75$.

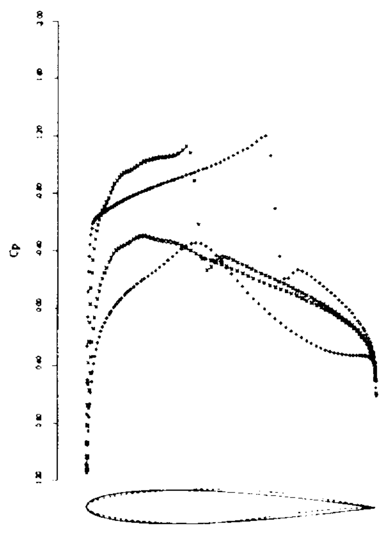


9.18a: *Initial Condition,* $I = 144.250$
$C_l = 0.0027$, $C_d = 0.0002$, $\alpha = 0.0°$

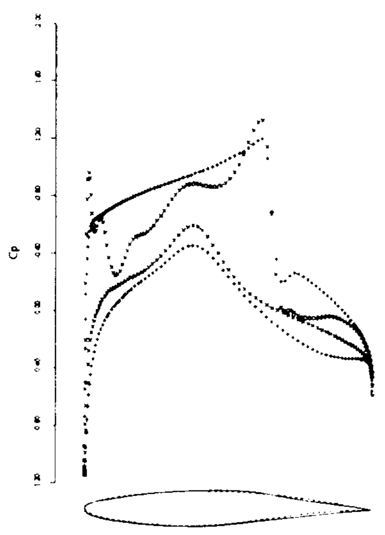


9.18b: *45 Design Iterations,* $I = 0.015$
$C_l = 0.8119$, $C_d = 0.0160$, $\alpha = 0.0°$

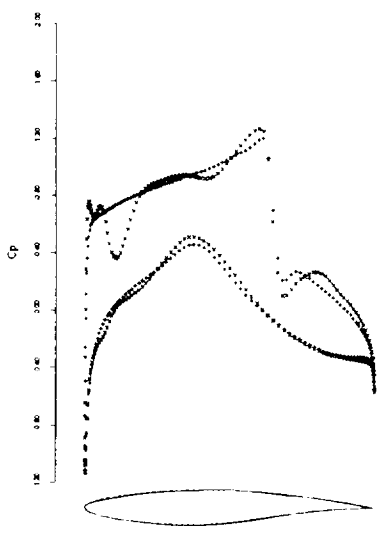
Figure 9.18: *Case 10:* $M = 0.70$ *Fixed Alpha Mode. Inverse Design, 31 B-Spline Design Variables.*
—, × *Initial Airfoil: NACA 0012.*
- - -, + *Target* C_p*: GAW 72,* $M = 0.70$.



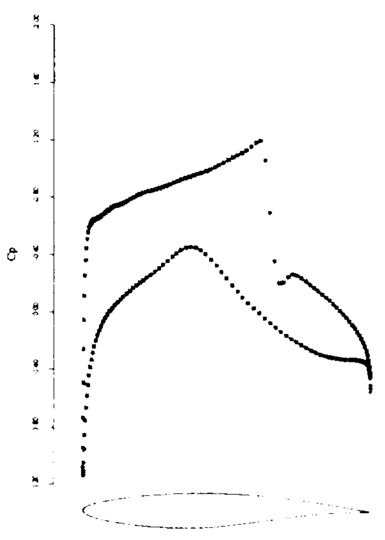
9.19a: *Initial Condition,* $I = 68.943$
$C_l = 0.2261$, $C_d = 0.0027$, $\alpha = 1.0°$



9.19b: *5 Design Iterations,* $I = 16.762$
$C_l = 0.3616$, $C_d = 0.0075$, $\alpha = 1.0°$

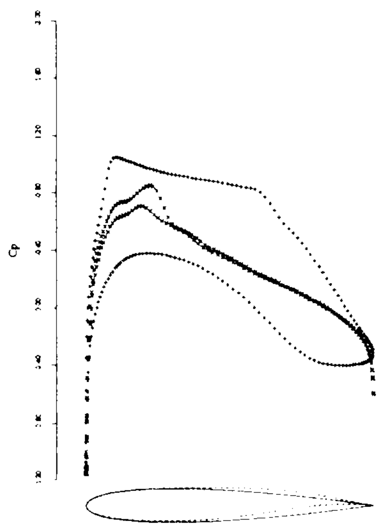


9.19c: *10 Design Iterations,* $I = 4.211$
$C_l = 0.5927$, $C_d = 0.0072$, $\alpha = 1.0°$

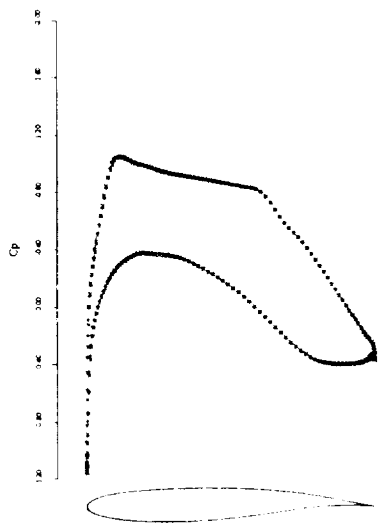


9.19d: *35 Design Iterations,* $I = 0.004$
$C_l = 0.6033$, $C_d = 0.0046$, $\alpha = 1.0°$

Figure 9.19: *Case 11:* $M = 0.75$, *Fixed Alpha Mode. Inverse Design, 31 B-Spline Design Variables.*
—, × *Initial Airfoil: NACA 0012.*
- - -, + *Target* C_p: *RAE 2822,* $M = 0.75$.



***9.20a:** Initial Condition, I = 63.043*
C_l = 0.0412, C_d = 0.0004, α = 0.175°



***9.20b:** 40 Design Iterations, I = 0.044*
C_l = 0.6302, C_d = 0.0004, α = 0.175°

***Figure 9.20:** Case 12: M = 0.75, **Fixed Alpha Mode.** Inverse Design, 31 B-Spline Design Variables.*
—, × *Initial Airfoil: NACA 0012.*
- - -, + *Target C_p: Korn Airfoil, M = 0.75.*

Chapter 10

CONCLUSIONS AND FUTURE WORK

This work has presented two new aerodynamic shape design methods. Both methods addressed the problem of transonic airfoil design subject to inviscid governing equations. The first examined the use of the potential flow equation while the second employed the Euler equations. The design problems for both methods were developed as general formulations such that any likely aerodynamic figure of merit may be treated within the framework of gradient based numerical optimization. The computationally expensive finite difference gradients often used by such methods were replaced here with gradients calculated via adjoint solvers. This idea which was originally proposed by Jameson [54, 55] for transonic flows was extended here such that constraints with respect to geometric complexity have been removed. This was accomplished by replacing the reliance of the previous method on an analytic mapping with a general finite volume formulation combined with a grid perturbation method. The research also introduced the use of both alternative design space parameterizations and an alternative design space search strategy. Two different parameterizations in the form of Hicks-Henne functions and B-spline control points were tested. The design space search strategy was an unconstrained quasi-Newton method which has been frequently used for finite difference optimization problems. In addition, various discretization procedures for the adjoint systems that result from the application of continuous sensitivity analysis were presented.

10.1 Computational Savings Via Adjoint Formulations

The main idea of using control theory in aerodynamic design is to reduce the computational requirements of the gradient calculation required for the design methods from $(\bar{n})(f_{cost})$ to $2(f_{cost})$, where $\bar{n}$ is the number of design variables, and f_{cost} is the approximate computational cost of a single flow analysis. In the examples that have been presented in this research, a reasonable estimate of adequately parameterizing the design space for airfoils was 50 degrees of freedom (50 design variables). A simple estimate for the amount of savings achievable in the calculation of the gradient is given below:

$$\begin{aligned}\text{Flow Analysis} &= 1\ f_{cost}\\ \text{Finite Difference Gradient Calculation} &= 51\ f_{cost}\\ \text{Adjoint Based Gradient Calculation} &= 2\ f_{cost}\end{aligned}$$

The actual savings that were seen in the Euler equations were:

$$\begin{aligned}\text{Flow Analysis} &= 5.8 \text{ CPU seconds}\\ \text{Finite Difference Gradient Calculation} &= 59.3 \text{ CPU seconds}\\ \text{Adjoint Based Gradient Calculation} &= 8.2 \text{ CPU seconds}\end{aligned}$$

The reason that a factor of about 25 reduction in the CPU time for the gradient calculation, as might be expected, was not obtained results from the fact that the finite difference gradient was significantly cheaper than predicted by the simple estimate. The use of restart solution data to accelerate the convergence of the nearby solutions needed in the finite difference gradient is mostly responsible for this difference between the estimated and actual costs. Nevertheless, it is still seen that a large reduction in CPU time to calculate the gradient is realized by the use of the adjoint method.

The computational costs of the entire design problem is summarized below.

$$\text{Actual Flow Analysis (Initial Point)} = 5.8 \text{ CPU seconds}$$

Actual Adjoint Based Gradient Calculation (Initial Point)	=	8.2 CPU seconds
Actual 35 Design Iteration Using the Adjoint Method	=	665.0 CPU seconds

The actual computational cost for 35 iterations was dominated (~85%) by the 2-5 flow solutions which were performed within each line search. While the entire design code was not run using the finite difference gradient, a reasonable estimate of the CPU time required for such a run is 2500 seconds. Implying that roughly a factor of 4 reduction in CPU time for the overall design procedure was achieved by employing adjoint gradients.

If three-dimensional problems are to be considered it is clear that the computational savings would be even greater due to the necessarily greater number of design variables. The actual computational savings of the adjoint methods are further enhanced by their reduced convergence requirements for the state and costate systems when compared with their finite difference counterparts. The cost savings realized by the adjoint method when it is actually employed within a design algorithm will depend on the number of additional flow calculations that may be needed in univariate searches. Such extra flow calculations incur the same additional computational costs regardless of whether the adjoint approach is used or not. Thus these extra costs will have a greater relative impact on the total CPU time of adjoint-based methods since they may make up a large portion of the computational costs outside the gradient calculation.

10.2 CONTINUOUS VS. DISCRETE SENSITIVITY ANALYSIS

The new methods developed in this thesis both implement continuous sensitivity methods. While these methods produce gradients that may not exactly correspond to those obtained via finite differences they must produce gradients that are consistent with the solution to the continuous problem within the accuracy of the discretization. Equivalently, it may be stated that the gradients obtained, in the limit of convergence and step size for the finite difference method, will always contain errors with respect to the gradient of the continuous system simply due to the discrete flow field approximations. In contrast, if a method is developed using full discrete sensitivity analysis it must necessarily produce gradients, in the limit of convergence and step size, that exactly matches those obtained by finite differences. Thus, discrete sensitivity methods produce gradients that are consistent with the discrete representation of the flow equations regardless of specific discretization errors that may be present in the flow equations. On the other hand, continuous sensitivity analysis produces gradients that are directly consistent with the original continuous system depend on the refinement of the mesh. And unless the discretization of the continuous adjoint system is done to mimic that for the transpose of the discrete flow equations these gradients may differ from finite difference gradients by an amount consistent with the difference in discretizations between the two systems. In the limit of mesh refinement both the continuous and the discrete methods as well as the finite difference method should produce identical gradients. In this research a preliminary study of the importance of these various discretization options for the adjoint system was studied. It appears that the continuous sensitivity method produces gradients which are accurate

10.3 Optimization Strategy

The second aspect that deserves a significant research effort in future works is the trade-off between the accuracy required of the flow and adjoint convergence and the optimization algorithm. This trade-off will have consequences for determining the type of optimization algorithm that is used and the manner in which constraints are enforced. This body of work did not attempt to resolve this open issue since it was felt that such an effort would have resulted in a research project that would extend well beyond the scope of a single Ph.D. thesis. In this work an unconstrained quasi-Newton design method using BFGS updates to the approximate Hessian was used. Since this method uses rank two updates to build the approximate Hessian, reasonable accuracy in both the gradient direction and achieving the minimum along the line search direction were necessary. This typically required 2-5 flow analyses in the line searches using the successive quadratic interpolation technique. This choice of the optimization algorithm often proved ideal in finite difference applications in which unconstrained methods sufficed. However, with the development of adjoint methods to obtain cheap gradients for CFD problems, BFGS quasi-Newton methods may no longer be the wise choice. For one, the cost of assuring very accurate gradients and and somewhat accurate line search minimums, which are necessary to assure accurate rank two updates, may be overkill since a constant step along a very inexpensive and yet reasonably accurate gradient should insure modest progress toward a minimum during the initial steps of the design process. To illustrate the idea, instead of highly converging both the flow and the adjoint systems and then constructing a highly accurate gradient that is passed to a quasi-Newton method, a tightly coupled approach takes a few steps in the flow solution, a few steps in the adjoint solution, calculates an approximate gradient, takes a constant preset step in the gradient direction, and continues. This is exactly the approach that both Ta'asan et al. and Jameson have independently explored in conjunction with adjoint methods.

BFGS quasi-Newton methods may also become problematic for an entirely different reason when adjoint-based three-dimensional design problems are to be considered. In such problems it is naturally attractive to exploit the cheap gradient information by parameterizing the design space with hundreds if not thousands of design variables. However, quasi-Newton methods become more cumbersome as the number of design variables become truly large due to the necessity of performing matrix operations. Furthermore, it may take considerably longer to develop a Hessian with sufficient rank, bringing into question the usefulness of the early approximate Hessian information. Unfortunately the simple, tightly coupled steepest descent method previously outlined may also not be the ideal solution. For one, it is well know that steepest descent methods can have very poor convergence performance on even slightly non-linear problems. Also, an appropriate constant step size in the initial stages of the design may become poorly scaled as the optimum is reached. A possible alternative would be a low memory or reduced quasi-Newton method that retains only the last few updates. Simple rank one updates may be used such that the minimum along each successive search direction need not be reached. The method would not require storage above that of a limited number of vectors used to reconstruct the approximate Hessian at each design iteration. Furthermore only the last updates which should remain locally pertinent within the quadratic approximation would be retained. Such a method should be more tolerant of changes in the Hessian as large design spaces are explored with many iterations. As in the case of the method first employed by Jameson, a limited memory quasi-Newton method could be used without converging either the flow solver or the adjoint solver since any errors in the resulting gradients and hence the Hessian updates would only be retained in the approximate Hessian for a few design iterations. This is in contrast to the traditional quasi-Newton methods which are permanently contaminated by any gradient inaccuracies. Other design algorithm alternatives should also be explored in conjunction with the use of adjoint methods, one possibility being a full Newton method with the solution of the adjoint equation used to assist in the construction of the full Hessian at each iteration as well as the gradient (see Appendix A).

10.4 GEOMETRIC GENERALITY AND DESIGN PARAMETERIZATIONS

This research focused on demonstrating the viability of using adjoint-based design methods in the context of traditional optimization strategies and realized, to first order, the computational benefits outlined above. It was shown that the basic approach could be applied to not only different governing differential equations, but also different parameterizations of the design space such as Hicks-Henne functions or B-spline control points. Unlike Jameson's previous efforts, here, no specific mappings from physical to computational space were required here. Instead, subsequent meshes, required to determine the manner in which variations of the design variable affect the variation of the cost function, were created via an analytic mesh perturbation algorithm. Essentially, this algorithm defined an algebraic relationship for determining how changes in the mesh point locations at the surface propagate into the flow field domain. The only necessary information needed by such an algorithm is a "high quality" initial mesh and the changes that are desired at the bounding surfaces. This approach, in theory, will allow any mesh topology and/or structure to be treated without restriction to a particular mapping. Further, savings in the design methods were obtained by reducing the integrals spanning the entire flow fields to integrals acting only on the bounding surfaces. This reduction was possible through the use of the known analytic relationship between the surface points and the mesh as a whole.

The work here also replaces Jameson's original parameterization of the surface control (the actual mesh points) with smooth analytic functions such that the transient as well as final airfoil designs remained smooth. This allowed the design method to proceed without the use of implicit smoothing of the gradients. The two different design space parameterizations that were tested (Hicks-Henne functions and B-spline control points) both showed promise and versatility. However, much further work in this area will be required.

10.5 FUTURE WORK

Numerous open issues still remain for future research. Among these are the following:

- Continuous vs. Discrete Sensitivity Analysis
- The Level of Coupling Between the Flow, Adjoint, and Design Systems
- The Choice of the Optimization Algorithm
- The Choice of Parameterization of the Design Space
- The Manner in which Constraints are Built into the Design Method

This work represents just the initial demonstration cases for a new and rapidly developing technology. Three research papers have been presented by the author based on implementations that have been developed in this dissertation [92, 65, 94]. However, due to the pace of research in this area, none of these have appeared yet in archival journals. The importance of this research is clear since it opens the way for efficient three-dimensional automatic design algorithms. Unlike many previous design approaches that have been developed for two-dimensional problems but could not easily be extended to three dimensions, the work here extends directly to three dimensions. Furthermore, the use of mesh metrics and not a mapping Jacobian implies that the new techniques are not dependent on any mesh type. Demonstrations of the extension to three-dimensional design problems with general and even multiblock meshes have very recently been achieved by the author and his collaborators [93, 95, 23, 96]. These later efforts were not presented here simply to keep the focus of the research on the more technical aspects of adjoint-based design as opposed to its practical extensions. Finally, it is noted that even unstructured meshes may be treated under the same framework presented in this thesis.

Appendix A

SECOND ORDER METHODS

Generally, a design optimization procedure benefits when more information pertaining to the nature of the design space is available. For the gradient-based methods introduced in *Chapter 3*, the more terms of the Taylor series expansion that are retained, the greater the accuracy of the approximation. However, as discussed before, obtaining higher-order information is even more computationally expensive than obtaining gradient information. Thus, this Appendix presents a technique by which the full Hessian can be obtained at a significant reduction in computational cost over that for a purely finite difference method. The development resembles the use of control theory to obtain cheap first derivatives.

A.1 REVIEW OF FIRST ORDER FORMULATION

The cost function and the flow field constraints can be written in index notation as

$$I = I(w_i, \mathcal{X}_i, \mathcal{F}_k),$$

$$R_j(w_i, \mathcal{X}_i, \mathcal{F}_k) = 0.$$

Here, i and j represent the domain mesh points and k represents the surface mesh points. Where Einstein's summation notation is implied, the first variations become

$$\delta I = \frac{\partial I}{\partial w_i}\delta w_i + \frac{\partial I}{\partial \mathcal{X}_i}\delta \mathcal{X}_i + \frac{\partial I}{\partial \mathcal{F}_k}\delta \mathcal{F}_k, \tag{A.1}$$

$$\delta R_j = \left[\frac{\partial R_j}{\partial w_i}\right]\delta w_i + \left[\frac{\partial R_j}{\partial \mathcal{X}_i}\right]\delta \mathcal{X}_i + \left[\frac{\partial R_j}{\partial \mathcal{F}_k}\right]\delta \mathcal{F}_k = 0. \tag{A.2}$$

where the sparseness of the matrices $\frac{\partial R_j}{\partial w_i}$ and $\frac{\partial R_j}{\partial \mathcal{X}_i}$ depend upon the stencil of support in the discrete scheme. By satisfying the adjoint equation,

$$\left[\frac{\partial R_i}{\partial w_j}\right]\psi_i = \frac{\partial I}{\partial w_j}, \tag{A.3}$$

the first variation in the cost function may be written as

$$\delta I = \frac{\partial I}{\partial \mathcal{X}_i}\delta\mathcal{X}_i + \frac{\partial I}{\partial \mathcal{F}_k}\delta\mathcal{F}_k - \psi_j\left(+\left[\frac{\partial R_j}{\partial \mathcal{X}_i}\right]\delta\mathcal{X}_i + \left[\frac{\partial R_j}{\partial \mathcal{F}_k}\right]\delta\mathcal{F}_k\right). \tag{A.4}$$

A.2 Second Order Formulation

The first variations may be written as

$$\delta I = \delta I(w_i, \mathcal{X}_i, \mathcal{F}_k, \delta w_i, \delta\mathcal{X}_i, \delta\mathcal{F}_k)$$

$$\delta R_j = \delta R_j(w_i, \mathcal{X}_i, \mathcal{F}_k, \delta w_i, \delta\mathcal{X}_i, \delta\mathcal{F}_k) = 0.$$

It is therefore possible to express using the chain rule the second variations as

$$\begin{aligned}\delta^2 I = \quad & \frac{\partial(\delta I)}{\partial w_i}\delta w_i + \frac{\partial(\delta I)}{\partial \mathcal{X}_i}\delta\mathcal{X}_i + \frac{\partial(\delta I)}{\partial \mathcal{F}_k}\delta\mathcal{F}_k \\ + \quad & \frac{\partial(\delta I)}{\partial(\delta w_i)}\delta^2 w_i + \frac{\partial(\delta I)}{\partial(\delta\mathcal{X}_i)}\delta^2\mathcal{X}_i + \frac{\partial(\delta I)}{\partial(\delta\mathcal{F}_k)}\delta^2\mathcal{F}_k \end{aligned} \tag{A.5}$$

$$\begin{aligned}\delta^2 R_j = 0 = \quad & \left[\frac{\partial(\delta R_j)}{\partial w_i}\right]\delta w_i + \left[\frac{\partial(\delta R_j)}{\partial \mathcal{X}_i}\right]\delta\mathcal{X}_i + \left[\frac{\partial(\delta R_j)}{\partial \mathcal{F}_k}\right]\delta\mathcal{F}_k \\ + \quad & \left[\frac{\partial(\delta R_j)}{\partial(\delta w_i)}\right]\delta^2 w_i + \left[\frac{\partial(\delta R_j)}{\partial(\delta\mathcal{X}_i)}\right]\delta^2\mathcal{X}_i + \left[\frac{\partial(\delta R_j)}{\partial(\delta\mathcal{F}_k)}\right]\delta^2\mathcal{F}_k \end{aligned} \tag{A.6}$$

Using expressions (A.1) and (A.2) we have

$$\frac{\partial(\delta I)}{\partial(\delta w_i)} = \frac{\partial I}{\partial w_i}, \qquad \frac{\partial(\delta R_j)}{\partial(\delta w_i)} = \frac{\partial R_j}{\partial w_i},$$

$$\frac{\partial(\delta I)}{\partial(\delta\mathcal{X}_i)} = \frac{\partial I}{\partial \mathcal{X}_i}, \qquad \frac{\partial(\delta R_j)}{\partial(\delta\mathcal{X}_i)} = \frac{\partial R_j}{\partial \mathcal{X}_i},$$

$$\frac{\partial(\delta I)}{\partial(\delta\mathcal{F}_k)} = \frac{\partial I}{\partial \mathcal{F}_k}, \qquad \frac{\partial(\delta R_j)}{\partial(\delta\mathcal{F}_k)} = \frac{\partial R_j}{\partial \mathcal{F}_k},$$

$$\frac{\partial(\delta I)}{\partial w_i} = \frac{\partial^2 I}{\partial w_l \partial w_i}\delta w_l + \frac{\partial^2 I}{\partial \mathcal{X}_l \partial w_i}\delta\mathcal{X}_l + \frac{\partial^2 I}{\partial \mathcal{F}_m \partial w_i}\delta\mathcal{F}_m,$$

$$
\begin{aligned}
\frac{\partial(\delta I)}{\partial \mathcal{X}_i} &= \frac{\partial^2 I}{\partial w_l \partial \mathcal{X}_i}\delta w_l + \frac{\partial^2 I}{\partial \mathcal{X}_l \partial \mathcal{X}_i}\delta \mathcal{X}_l + \frac{\partial^2 I}{\partial \mathcal{F}_m \partial \mathcal{X}_i}\delta \mathcal{F}_m,\\
\frac{\partial(\delta I)}{\partial \mathcal{F}_k} &= \frac{\partial^2 I}{\partial w_l \partial \mathcal{F}_k}\delta w_l + \frac{\partial^2 I}{\partial \mathcal{X}_l \partial \mathcal{F}_k}\delta \mathcal{X}_l + \frac{\partial^2 I}{\partial \mathcal{F}_m \partial \mathcal{F}_k}\delta \mathcal{F}_m,\\
\frac{\partial(\delta R_j)}{\partial w_i} &= \frac{\partial^2 R_j}{\partial w_l \partial w_i}\delta w_l + \frac{\partial^2 R_j}{\partial \mathcal{X}_l \partial w_i}\delta \mathcal{X}_l + \frac{\partial^2 R_j}{\partial \mathcal{F}_m \partial w_i}\delta \mathcal{F}_m,\\
\frac{\partial(\delta R_j)}{\partial \mathcal{X}_i} &= \frac{\partial^2 R_j}{\partial w_l \partial \mathcal{X}_i}\delta w_l + \frac{\partial^2 R_j}{\partial \mathcal{X}_l \partial \mathcal{X}_i}\delta \mathcal{X}_l + \frac{\partial^2 R_j}{\partial \mathcal{F}_m \partial \mathcal{X}_i}\delta \mathcal{F}_m,\\
\frac{\partial(\delta R_j)}{\partial \mathcal{F}_k} &= \frac{\partial^2 R_j}{\partial w_l \partial \mathcal{F}_k}\delta w_l + \frac{\partial^2 R_j}{\partial \mathcal{X}_l \partial \mathcal{F}_k}\delta \mathcal{X}_l + \frac{\partial^2 R_j}{\partial \mathcal{F}_m \partial \mathcal{F}_k}\delta \mathcal{F}_m,
\end{aligned}
\tag{A.7}
$$

where l also represents the points in the domain and m represents the surface points. By combining (A.5) and (A.6) with the co-state variable ψ_j and substituting (A.7), $\delta^2 I$ can be written as,

$$
\begin{aligned}
\delta^2 I = \quad & \frac{\partial^2 I}{\partial w_l \partial w_i}\delta w_l \delta w_i + \frac{\partial^2 I}{\partial \mathcal{X}_l \partial \mathcal{X}_i}\delta \mathcal{X}_l \delta \mathcal{X}_i + \frac{\partial^2 I}{\partial \mathcal{F}_m \partial \mathcal{F}_k}\delta \mathcal{F}_m \delta \mathcal{F}_k\\
& +2\frac{\partial^2 I}{\partial \mathcal{X}_l \partial w_i}\delta \mathcal{X}_l \delta w_i + 2\frac{\partial^2 I}{\partial \mathcal{F}_m \partial w_i}\delta \mathcal{F}_m \delta w_i + 2\frac{\partial^2 I}{\partial \mathcal{F}_m \partial \mathcal{X}_i}\delta \mathcal{F}_m \delta \mathcal{X}_i\\
& +\frac{\partial I}{\partial \mathcal{X}_i}\delta^2 \mathcal{X}_i + \frac{\partial I}{\partial \mathcal{F}_k}\delta^2 \mathcal{F}_k\\
& -\psi_j\left[\frac{\partial^2 R_j}{\partial w_l \partial w_i}\delta w_l \delta w_i + \frac{\partial^2 R_j}{\partial \mathcal{X}_l \partial \mathcal{X}_i}\delta \mathcal{X}_l \delta \mathcal{X}_i + \frac{\partial^2 R_j}{\partial \mathcal{F}_m \partial \mathcal{F}_k}\delta \mathcal{F}_m \delta \mathcal{F}_k\right]\\
& -\psi_j\left[2\frac{\partial^2 R_j}{\partial \mathcal{X}_l \partial w_i}\delta \mathcal{X}_l \delta w_i + 2\frac{\partial^2 R_j}{\partial \mathcal{F}_m \partial w_i}\delta \mathcal{F}_m \delta w_i + 2\frac{\partial^2 R_j}{\partial \mathcal{F}_m \partial \mathcal{X}_i}\delta \mathcal{F}_m \delta \mathcal{X}_i\right]\\
& -\psi_j\left[+\frac{\partial R_j}{\partial \mathcal{X}_i}\delta^2 \mathcal{X}_i + \frac{\partial R_j}{\partial \mathcal{F}_k}\delta^2 \mathcal{F}_k\right].
\end{aligned}
\tag{A.8}
$$

This daunting expression still has 16 terms that involve the variations δw, $\delta \mathcal{X}$, $\delta \mathcal{F}$, $\delta^2 \mathcal{X}$, and $\delta^2 \mathcal{F}$. The only variation which has been eliminated by using ψ is $\delta^2 w$. However, if the grid generation and surface perturbation are analytic, all the terms with the exception of δw should be easy to obtain. In this case, δw can be obtained from the direct expression,

$$
\frac{\partial R_j}{\partial w_i}\delta w_i = -\frac{\partial R_j}{\partial \mathcal{X}_i}\delta \mathcal{X}_i - \frac{\partial R_j}{\partial \mathcal{F}_k}\delta \mathcal{F}_k.
$$

Therefore the term δw must be calculated for each right hand side depending on each design variable. This is equivalent in cost to a finite difference gradient evaluation. Thus the cost of evaluating $\delta^2 I$ is $n + 1$ flow calculations plus one adjoint calculation. An alternative design method is then possible that is not much more computationally expensive than a finite difference gradient method, but promises faster convergence to the minimum through the use of the full and up to date Hessian. The implementation of such a method is beyond the scope of this research and it remains an open issue whether such methods would be advantageous. When compared with using control theory for just the the gradient information, the addition of n flow calculations per design iteration may not be warranted for obtaining the full Hessian information at each iteration. If the number of design variables required to parameterize the design space adequately becomes large, not only does the cost of n flow calculations become prohibitive but the cost of solving and storing the Newton problem with a full rank Hessian also becomes an issue. For that matter, if the number of design variables becomes very large ($O(10^3)$ or greater) even the cost of quasi-Newton updates and direction calculations may force the use of a much more simplistic optimization process such as steepest descent.

An additional concern of this full Newton design method is the practical evaluation of the third order tensors $\frac{\partial^2 R_j}{\partial w_l \partial w_i}$, $\frac{\partial^2 R_j}{\partial X_l \partial X_i}$, $\frac{\partial^2 R_j}{\partial X_l \partial w_i}$. The gradient evaluation only required knowledge of the standard sparse Jacobian matrices, $\frac{\partial R_j}{\partial w_i}$ and $\frac{\partial R_j}{\partial X_i}$, that are readily available in many flow solution methodologies.

One of the fortunate aspects of the higher-order approach is that once an adjoint solver is available, no additional difficult iterative solution procedure need be constructed since the same adjoint problem applies whether the desired goal is 1st or 2nd order information. In practice, an implementation of such a method is somewhat simplified from equation (A.8) since many terms drop out. The following section develops a specific application of the second order design algorithm proposed here.

A.3 APPLICATION TO THE EULER EQUATIONS

In order to see how a second order design algorithm would work on a specific problem we shall once again consider the design of airfoils subject to the inviscid Euler equations. The cost function chosen as an illustration of the technique will be the inverse design problem at a fixed angle of attack.

Repeating the development in *Chapter 5* the cost function may be written as

$$I = \frac{1}{2}\oint_C (p - p_d)^2\, ds = \frac{1}{2}\oint_C (p - p_d)^2 \left(\frac{ds}{d\xi}\right) d\xi,$$

while its first variation is

$$\delta I = \oint_C (p - p_d)\,\delta p \left(\frac{ds}{d\xi}\right) d\xi + \frac{1}{2}\oint_C (p - p_d)^2\, \delta\left(\frac{ds}{d\xi}\right) d\xi. \qquad \text{(A.9)}$$

To develop a second order approach it is necessary to obtain the second variation of the cost function as well:

$$\begin{aligned}\delta^2 I = &\quad \oint_C \delta p \delta p \left(\frac{ds}{d\xi}\right) d\xi + 2\oint_C (p - p_d)\,\delta p\, \delta\left(\frac{ds}{d\xi}\right) d\xi \\ &+ \frac{1}{2}\oint_C (p - p_d)^2\, \delta^2\left(\frac{ds}{d\xi}\right) d\xi + \oint_C (p - p_d)\,\delta^2 p \left(\frac{ds}{d\xi}\right) d\xi. \end{aligned} \qquad \text{(A.10)}$$

Recall from *Chapter 5* that the governing equation was given as

$$\frac{\partial(\delta F)}{\partial\xi} + \frac{\partial(\delta G)}{\partial\eta} = 0 \quad \text{in } D, \qquad \text{(A.11)}$$

while its first variation was

$$\begin{aligned}\delta F &= J\frac{\partial\xi}{\partial x}\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\delta\mathbf{w} + J\frac{\partial\xi}{\partial y}\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\delta\mathbf{w} + \delta\left(J\frac{\partial\xi}{\partial x}\right)\mathbf{f} + \delta\left(J\frac{\partial\xi}{\partial y}\right)\mathbf{g}\\ \delta G &= J\frac{\partial\eta}{\partial x}\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\delta\mathbf{w} + J\frac{\partial\eta}{\partial y}\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\delta\mathbf{w} + \delta\left(J\frac{\partial\eta}{\partial x}\right)\mathbf{f} + \delta\left(J\frac{\partial\eta}{\partial y}\right)\mathbf{g}.\end{aligned}$$

Now just as in the case of the cost function, the second variation of (A.11) is also needed:

$$\frac{\partial\left(\delta^2 F\right)}{\partial\xi} + \frac{\partial\left(\delta^2 G\right)}{\partial\eta} = 0 = \delta^2 R \quad \text{in } D, \qquad \text{(A.12)}$$

where

$$\begin{aligned}\delta^2 F &= J\frac{\partial\xi}{\partial x}\frac{\partial^2 \mathbf{f}}{\partial \mathbf{w}^2}\delta\mathbf{w}\delta\mathbf{w} + J\frac{\partial\xi}{\partial y}\frac{\partial^2 \mathbf{g}}{\partial \mathbf{w}^2}\delta\mathbf{w}\delta\mathbf{w} + 2\delta\left(J\frac{\partial\xi}{\partial x}\right)\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\delta\mathbf{w} + 2\delta\left(J\frac{\partial\xi}{\partial y}\right)\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\delta\mathbf{w}\\ &+ J\frac{\partial\xi}{\partial x}\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\delta^2\mathbf{w} + J\frac{\partial\xi}{\partial y}\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\delta^2\mathbf{w} + \delta^2\left(J\frac{\partial\xi}{\partial x}\right)\mathbf{f} + \delta^2\left(J\frac{\partial\xi}{\partial y}\right)\mathbf{g},\end{aligned}$$

$$\begin{aligned}\delta^2 G &= J\frac{\partial\eta}{\partial x}\frac{\partial^2\mathbf{f}}{\partial\mathbf{w}^2}\delta\mathbf{w}\delta\mathbf{w} + J\frac{\partial\eta}{\partial y}\frac{\partial^2\mathbf{g}}{\partial\mathbf{w}^2}\delta\mathbf{w}\delta\mathbf{w} + 2\delta\left(J\frac{\partial\eta}{\partial x}\right)\frac{\partial\mathbf{f}}{\partial\mathbf{w}}\delta\mathbf{w} + 2\delta\left(J\frac{\partial\eta}{\partial y}\right)\frac{\partial\mathbf{g}}{\partial\mathbf{w}}\delta\mathbf{w}\\ &+ J\frac{\partial\eta}{\partial x}\frac{\partial\mathbf{f}}{\partial\mathbf{w}}\delta^2\mathbf{w} + J\frac{\partial\eta}{\partial y}\frac{\partial\mathbf{g}}{\partial\mathbf{w}}\delta^2\mathbf{w} + \delta^2\left(J\frac{\partial\eta}{\partial x}\right)\mathbf{f} + \delta^2\left(J\frac{\partial\eta}{\partial y}\right)\mathbf{g}.\end{aligned}$$

It is noted that (A.12) is set equal to zero since all variations of R must necessarily be zero. Equivalently stated, the residual must be forced to zero everywhere within the design space. It was this fact that allowed us to use the first variation of R as a constraint to eliminate $\delta\mathbf{w}$ from the definition of the first variation of the cost function.

After multiplying (A.12) by a still arbitrary adjoint vector and integrating over the domain we have

$$\int_D \psi^T\left(\frac{\partial\delta^2 F}{\partial\xi} + \frac{\partial\delta^2 G}{\partial\eta}\right)d\xi d\eta = 0,$$

which after integration by parts may be written as

$$\begin{aligned}\int_D\left(\frac{\partial\psi^T}{\partial\xi}\delta^2 F + \frac{\partial\psi^T}{\partial\eta}\delta^2 G\right)d\xi d\eta = &\oint_B\left(n_\xi\psi^T\delta^2 F + n_\eta\psi^T\delta^2 G\right)d\xi\\ + &\oint_C\left(n_\xi\psi^T\delta^2 F + n_\eta\psi^T\delta^2 G\right)d\xi.\end{aligned}$$

Just as in the body of the thesis, an O-mesh is implied with the integrals along the cut line cancelling. On the profile, where $n_\xi = 0$ and $n_\eta = -1$, and the boundary conditions for the Euler equations are given by

$$G = J\begin{bmatrix}0\\ \frac{\partial\eta}{\partial x}p\\ \frac{\partial\eta}{\partial y}p\\ 0\end{bmatrix},$$

the second variation may be given by,

$$\delta^2 G = J \begin{bmatrix} 0 \\ \frac{\partial \eta}{\partial x}\delta^2 p \\ \frac{\partial \eta}{\partial y}\delta^2 p \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 2\delta\left(J\frac{\partial \eta}{\partial x}\right)\delta p \\ 2\delta\left(J\frac{\partial \eta}{\partial y}\right)\delta p \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \delta^2\left(J\frac{\partial \eta}{\partial x}\right)p \\ \delta^2\left(J\frac{\partial \eta}{\partial y}\right)p \\ 0 \end{bmatrix}. \tag{A.13}$$

Then by choosing $\psi = 0$ at the outer boundary where $\bar{n}_\xi = 0$ and $\bar{n}_\eta = 1$, and forcing ψ to satisfy our original adjoint expression, we have

$$\frac{\partial \psi}{\partial t} - C_1^T\frac{\partial \psi}{\partial \xi} - C_2^T\frac{\partial \psi}{\partial \eta} = 0 \quad \text{in } D, \tag{A.14}$$

as well as the boundary condition

$$J\left(\psi_2\frac{\partial \eta}{\partial x} + \psi_3\frac{\partial \eta}{\partial y}\right) = -(p - p_d)\frac{ds}{d\xi} \quad \text{on } C. \tag{A.15}$$

The final form of the second variation of the cost function can be written without a dependency on $\delta^2\mathbf{w}$ as

$$\begin{aligned} \delta^2 I \;=\; & \oint_C \delta p\delta p\left(\frac{ds}{d\xi}\right)d\xi + 2\oint_C (p - p_d)\delta p\left(\frac{ds}{d\xi}\right)d\xi + \frac{1}{2}\oint_C (p - p_d)^2\delta^2\left(\frac{ds}{d\xi}\right)d\xi \\ + & \int_D\left(\frac{\partial \psi^T}{\partial \xi}\tilde{\delta}^2 F + \frac{\partial \psi^T}{\partial \eta}\tilde{\delta}^2 G\right)d\xi d\eta \\ + & \oint_B\left\{\psi_2\left[\delta^2\left(J\frac{\partial \eta}{\partial x}\right)p + 2\delta\left(J\frac{\partial \eta}{\partial x}\right)\delta p\right] + \psi_3\left[\delta^2\left(J\frac{\partial \eta}{\partial y}\right)p + 2\delta\left(J\frac{\partial \eta}{\partial y}\right)\delta p\right]\right\}d\xi. \end{aligned} \tag{A.16}$$

The $\tilde{\delta}^2 F$ and $\tilde{\delta}^2 G$ terms are defined as

$$\begin{aligned} \tilde{\delta}^2 F \;=\; & J\frac{\partial \xi}{\partial x}\frac{\partial^2 \mathbf{f}}{\partial \mathbf{w}^2}\delta\mathbf{w}\delta\mathbf{w} + J\frac{\partial \xi}{\partial y}\frac{\partial^2 \mathbf{g}}{\partial \mathbf{w}^2}\delta\mathbf{w}\delta\mathbf{w} + 2\delta\left(J\frac{\partial \xi}{\partial x}\right)\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\delta\mathbf{w} + 2\delta\left(J\frac{\partial \xi}{\partial y}\right)\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\delta\mathbf{w} \\ + & \delta^2\left(J\frac{\partial \xi}{\partial x}\right)\mathbf{f} + \delta^2\left(J\frac{\partial \xi}{\partial y}\right)\mathbf{g}, \\ \tilde{\delta}^2 G \;=\; & J\frac{\partial \eta}{\partial x}\frac{\partial^2 \mathbf{f}}{\partial \mathbf{w}^2}\delta\mathbf{w}\delta\mathbf{w} + J\frac{\partial \eta}{\partial y}\frac{\partial^2 \mathbf{g}}{\partial \mathbf{w}^2}\delta\mathbf{w}\delta\mathbf{w} + 2\delta\left(J\frac{\partial \eta}{\partial x}\right)\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\delta\mathbf{w} + 2\delta\left(J\frac{\partial \eta}{\partial y}\right)\frac{\partial \mathbf{g}}{\partial \mathbf{w}}\delta\mathbf{w} \\ + & \delta^2\left(J\frac{\partial \eta}{\partial x}\right)\mathbf{f} + \delta^2\left(J\frac{\partial \eta}{\partial y}\right)\mathbf{g}. \end{aligned}$$

It is noteworthy that even though $\delta^2 \mathrm{w}$ has been eliminated from (A.16) the equation still contains $\delta \mathrm{w}$, $\delta\left(J\frac{\partial \xi_i}{\partial x_j}\right)$, and $\delta^2\left(J\frac{\partial \xi_i}{\partial x_j}\right)$. The expression may be simplified by choosing a grid perturbation method in which $\delta^2\left(J\frac{\partial \xi_i}{\partial x_j}\right) = 0$. However, even with this assumption, evaluating (A.16) is no simple task.

Appendix B

TABLES OF DESIGN VARIABLES

Design Variable	Width Exponent	x Position	Scaling	Stepsize	Surface
1	3.50	0.975	0.01	0.0001	Lower
2	3.50	0.95	0.01	0.0001	Lower
3	3.50	0.925	0.01	0.0001	Lower
4	3.50	0.90	0.01	0.0001	Lower
5	3.50	0.85	0.01	0.0001	Lower
6	4.00	0.80	0.01	0.0001	Lower
7	4.00	0.75	0.01	0.0001	Lower
8	4.00	0.70	0.01	0.0001	Lower
9	4.00	0.65	0.01	0.0001	Lower
10	4.50	0.60	0.01	0.0001	Lower
11	4.50	0.55	0.01	0.0001	Lower
12	4.50	0.50	0.01	0.0001	Lower
13	4.50	0.45	0.01	0.0001	Lower
14	4.50	0.40	0.01	0.0001	Lower
15	4.00	0.35	0.01	0.0001	Lower
16	4.00	0.30	0.01	0.0001	Lower
17	4.00	0.25	0.01	0.0001	Lower
18	3.50	0.20	0.01	0.0001	Lower
19	3.50	0.15	0.01	0.0001	Lower
20	4.00	0.125	0.01	0.0001	Lower
21	4.00	0.10	0.01	0.0001	Lower
22	4.00	0.075	0.01	0.0001	Lower
23	4.00	0.05	0.01	0.0001	Lower
24	4.00	0.025	0.01	0.0001	Lower
25	4.00	0.0125	0.01	0.0001	Lower

Table B.1: *Design Variable Set 1: Lower Surface — 1-25*

Design Variable	Width Exponent	x Position	Scaling	Stepsize	Surface
26	4.00	0.0125	0.01	0.0001	Upper
27	4.00	0.025	0.01	0.0001	Upper
28	4.00	0.05	0.01	0.0001	Upper
29	4.00	0.075	0.01	0.0001	Upper
30	4.00	0.10	0.01	0.0001	Upper
31	4.00	0.125	0.01	0.0001	Upper
32	3.50	0.15	0.01	0.0001	Upper
33	3.50	0.20	0.01	0.0001	Upper
34	4.00	0.25	0.01	0.0001	Upper
35	4.00	0.30	0.01	0.0001	Upper
36	4.00	0.35	0.01	0.0001	Upper
37	4.50	0.40	0.01	0.0001	Upper
38	4.50	0.45	0.01	0.0001	Upper
39	4.50	0.50	0.01	0.0001	Upper
40	4.50	0.55	0.01	0.0001	Upper
41	4.50	0.60	0.01	0.0001	Upper
42	4.00	0.65	0.01	0.0001	Upper
43	4.00	0.70	0.01	0.0001	Upper
44	4.00	0.75	0.01	0.0001	Upper
45	4.00	0.80	0.01	0.0001	Upper
46	3.50	0.85	0.01	0.0001	Upper
47	3.50	0.90	0.01	0.0001	Upper
48	3.50	0.925	0.01	0.0001	Upper
49	3.50	0.95	0.01	0.0001	Upper
50	3.50	0.975	0.01	0.0001	Upper

Table B.2: *Design Variable Set 1: Upper Surface — 26-50*

Design Variable	Width Exponent	x Position	Scaling	Stepsize	Surface
1	3.50	0.975	0.01	0.0001	Lower
2	3.50	0.95	0.01	0.0001	Lower
3	3.50	0.925	0.01	0.0001	Lower
4	3.50	0.90	0.01	0.0001	Lower
5	3.50	0.85	0.01	0.0001	Lower
6	4.00	0.80	0.01	0.0001	Lower
7	4.00	0.75	0.01	0.0001	Lower
8	4.00	0.70	0.01	0.0001	Lower
9	4.00	0.65	0.01	0.0001	Lower
10	4.50	0.60	0.01	0.0001	Lower
11	4.50	0.55	0.01	0.0001	Lower
12	4.50	0.50	0.01	0.0001	Lower
13	4.50	0.45	0.01	0.0001	Lower
14	4.50	0.40	0.01	0.0001	Lower
15	4.00	0.35	0.01	0.0001	Lower
16	4.00	0.30	0.01	0.0001	Lower
17	4.00	0.25	0.01	0.0001	Lower
18	4.00	0.225	0.01	0.0001	Lower
19	4.00	0.20	0.01	0.0001	Lower
20	4.00	0.175	0.01	0.0001	Lower
21	4.00	0.15	0.01	0.0001	Lower
22	4.00	0.125	0.01	0.0001	Lower
23	4.00	0.10	0.01	0.0001	Lower
24	4.00	0.075	0.01	0.0001	Lower
25	4.00	0.05	0.01	0.0001	Lower
26	4.00	0.025	0.01	0.0001	Lower
27	4.00	0.0125	0.01	0.0001	Lower

Table B.3: *Design Variable Set 2: Lower Surface — 1-25*

Design Variable	Width Exponent	x Position	Scaling	Stepsize	Surface
28	4.00	0.0125	0.01	0.0001	Upper
29	4.00	0.025	0.01	0.0001	Upper
30	4.00	0.05	0.01	0.0001	Upper
31	4.00	0.075	0.01	0.0001	Upper
32	4.00	0.10	0.01	0.0001	Upper
33	4.00	0.125	0.01	0.0001	Upper
34	4.00	0.15	0.01	0.0001	Upper
35	4.00	0.175	0.01	0.0001	Upper
36	4.00	0.20	0.01	0.0001	Upper
37	4.00	0.225	0.01	0.0001	Upper
38	4.00	0.25	0.01	0.0001	Upper
39	4.00	0.30	0.01	0.0001	Upper
40	4.00	0.35	0.01	0.0001	Upper
41	4.50	0.40	0.01	0.0001	Upper
42	4.50	0.45	0.01	0.0001	Upper
43	4.50	0.50	0.01	0.0001	Upper
44	4.50	0.55	0.01	0.0001	Upper
45	4.50	0.60	0.01	0.0001	Upper
46	4.00	0.65	0.01	0.0001	Upper
47	4.00	0.70	0.01	0.0001	Upper
48	4.00	0.75	0.01	0.0001	Upper
49	4.00	0.80	0.01	0.0001	Upper
50	3.50	0.85	0.01	0.0001	Upper
51	3.50	0.90	0.01	0.0001	Upper
52	3.50	0.925	0.01	0.0001	Upper
53	3.50	0.95	0.01	0.0001	Upper
54	3.50	0.975	0.01	0.0001	Upper

Table B.4: *Design Variable Set 2: Upper Surface — 26-50*

Design Variable	Width Exponent	x Position	Scaling	Stepsize	Surface
1	3.50	0.10	0.01	0.0001	Upper and Lower
2	3.50	0.20	0.01	0.0001	Upper and Lower
3	4.00	0.30	0.01	0.0001	Upper and Lower
4	4.00	0.35	0.01	0.0001	Upper and Lower
5	4.50	0.40	0.01	0.0001	Upper and Lower
6	4.50	0.45	0.01	0.0001	Upper and Lower
7	4.50	0.50	0.01	0.0001	Upper and Lower
8	4.50	0.55	0.01	0.0001	Upper and Lower
9	4.50	0.60	0.01	0.0001	Upper and Lower
10	4.00	0.65	0.01	0.0001	Upper and Lower
11	4.00	0.70	0.01	0.0001	Upper and Lower
12	4.00	0.75	0.01	0.0001	Upper and Lower
13	4.00	0.80	0.01	0.0001	Upper and Lower
14	3.50	0.85	0.01	0.0001	Upper and Lower
15	3.50	0.90	0.01	0.0001	Upper and Lower
16	3.50	0.925	0.01	0.0001	Upper and Lower
17	3.50	0.95	0.01	0.0001	Upper and Lower
18	3.50	0.975	0.01	0.0001	Upper and Lower

Table B.5: *Design Variable Set 3: Camber — 1-18*

Design Variable	Width Exponent	x Position	Scaling	Stepsize	Surface
1	4.00	0.05	0.01	0.0001	Upper and Lower
2	4.00	0.10	0.01	0.0001	Upper and Lower
3	4.00	0.125	0.01	0.0001	Upper and Lower
4	4.00	0.15	0.01	0.0001	Upper and Lower
5	4.00	0.175	0.01	0.0001	Upper and Lower
6	4.00	0.20	0.01	0.0001	Upper and Lower
7	4.00	0.225	0.01	0.0001	Upper and Lower
8	4.00	0.25	0.01	0.0001	Upper and Lower
9	4.00	0.30	0.01	0.0001	Upper and Lower
10	4.00	0.35	0.01	0.0001	Upper and Lower
11	4.50	0.40	0.01	0.0001	Upper and Lower
12	4.50	0.45	0.01	0.0001	Upper and Lower
13	4.50	0.50	0.01	0.0001	Upper and Lower
14	4.50	0.55	0.01	0.0001	Upper and Lower
15	4.50	0.60	0.01	0.0001	Upper and Lower
16	4.00	0.65	0.01	0.0001	Upper and Lower
17	4.00	0.70	0.01	0.0001	Upper and Lower
18	4.00	0.75	0.01	0.0001	Upper and Lower
19	4.00	0.80	0.01	0.0001	Upper and Lower
20	3.50	0.85	0.01	0.0001	Upper and Lower
21	3.50	0.90	0.01	0.0001	Upper and Lower
22	3.50	0.925	0.01	0.0001	Upper and Lower
23	3.50	0.95	0.01	0.0001	Upper and Lower
24	3.50	0.975	0.01	0.0001	Upper and Lower

Table B.6: *Design Variable Set 4: Camber — 1-24*

Control Point	x Position	y Position	Active
1	0.00000000	0.00000000	No
2	0.00000000	0.00828974	Yes
3	0.01250000	0.02310806	Yes
4	0.05000000	0.03703213	Yes
5	0.10000000	0.04748770	Yes
6	0.15000000	0.05341722	Yes
7	0.20000000	0.05773245	Yes
8	0.30000000	0.06071101	Yes
9	0.40000000	0.05772764	Yes
10	0.50000000	0.05280310	Yes
11	0.60000000	0.04494146	Yes
12	0.70000000	0.03615675	Yes
13	0.80000000	0.02521739	Yes
14	0.85000000	0.01965614	Yes
15	0.90000000	0.01325599	Yes
16	0.95000000	0.00710581	Yes
17	0.97500000	0.00336899	Yes
18	1.00000000	0.00000000	No
19	0.00000000	0.00000000	No
20	0.00000000	-0.00828974	Equal to point 2
21	0.01250000	-0.02310806	Yes
22	0.05000000	-0.03703213	Yes
23	0.10000000	-0.04748770	Yes
24	0.15000000	-0.05341722	Yes
25	0.20000000	-0.05773245	Yes
26	0.30000000	-0.06071101	Yes
27	0.40000000	-0.05772764	Yes
28	0.50000000	-0.05280310	Yes
29	0.60000000	-0.04494146	Yes
30	0.70000000	-0.03615675	Yes
31	0.80000000	-0.02521739	Yes
32	0.85000000	-0.01965614	Yes
33	0.90000000	-0.01325599	Yes
34	0.95000000	-0.00710581	Yes
35	0.97500000	-0.00336899	Yes
36	1.00000000	0.00000000	No

Table B.7: *Design Variable Set 5: Upper and Lower Surfaces — 1-36*

Appendix C

SECOND FORM OF GREEN'S THEOREM

Given an operator of the form

$$L(\ ) = (C_1(\ )_\xi)_\xi + (C_2(\ )_\eta)_\xi + (C_2(\ )_\xi)_\eta + (C_3(\ )_\eta)_\eta$$

operating on a function ϕ, where C_{1-3} are frozen coefficients, the integral over a domain D,

$$\int_D (\psi L\phi)\, d\xi d\eta,$$

where ψ is another function, may be written after integration by parts as

$$\begin{aligned}\int_D (\psi L(\phi))\, d\xi d\eta &= -\int_D (\psi_\xi C_1 \phi_\xi + \psi_\xi C_2 \phi_\eta + \psi_\eta C_2 \phi_\xi + \psi_\eta C_3 \phi_\eta)\, d\xi d\eta \\ &\quad + \int_{\partial D} (\psi C_1 \phi_\xi + \psi C_2 \phi_\eta)\, d\eta + \int_{\partial D} (\psi C_2 \phi_\xi + \psi C_3 \phi_\eta)\, d\xi.\end{aligned}$$

∂D is on the boundaries of the domain involving ξ and η. Similarly we can write with the aid of integration by parts,

$$\begin{aligned}\int_D (\phi L(\psi))\, d\xi d\eta &= -\int_D (\phi_\xi C_1 \psi_\xi + \phi_\xi C_2 \psi_\eta + \phi_\eta C_2 \psi_\xi + \phi_\eta C_3 \psi_\eta)\, d\xi d\eta \\ &\quad + \int_{\partial D} (\phi C_1 \psi_\xi + \phi C_2 \psi_\eta)\, d\eta + \int_{\partial D} (\phi C_2 \psi_\xi + \phi C_3 \psi_\eta)\, d\xi.\end{aligned}$$

Thus, by subtracting one relation from the other, it is possible to develop the identity

$$\begin{aligned}\int_D (\psi L(\phi) - \phi L(\psi))\, d\xi d\eta &= +\int_{\partial D} (\psi C_1 \phi_\xi + \psi C_2 \phi_\eta)\, d\eta + \int_{\partial D} (\psi C_2 \phi_\xi + \psi C_3 \phi_\eta)\, d\xi \\ &\quad - \int_{\partial D} (\phi C_1 \psi_\xi + \phi C_2 \psi_\eta)\, d\eta - \int_{\partial D} (\phi C_2 \psi_\xi + \phi C_3 \psi_\eta)\, d\xi.\end{aligned}$$

Bibliography

[1] R. Aris. *Vectors, Tensors, and the Basic Equations of Fluid Mechanics*. Dover Publications, Inc., New York, 1989.

[2] F. Bauer, P. Garabedian, D. Korn, and A. Jameson. *Supercritical Wing Sections II*. Springer Verlag, New York, 1975.

[3] O. Baysal and M. E. Eleshaky. Aerodynamic design optimization using sensitivity analysis and computational fluid dynamics. *AIAA paper 91-0471*, 29th Aerospace Sciences Meeting, Reno, Nevada, January 1991.

[4] O. Baysal and M. E. Eleshaky. Aerodynamic sensitivity analysis methods for the compressible Euler equations. *Journal of Fluids Engineering*, 113(4):681–688, 1991.

[5] O. Baysal and M. E. Eleshaky. Aerodynamic design optimization using sensitivity analysis and computational fluid dynamics. *AIAA Journal*, 30(3):718–725, 1992.

[6] O. Baysal and M. E. Eleshaky. Airfoil shape optimization using sensitivity analysis on viscous flow equations. *Journal of Fluids Engineering*, 115:75–84, 1993.

[7] M. Bieterman, J. Bussoletti, C. Hilmes, F. Johnson, R. Melvin, and D. Young. An adaptive grid method for analysis of 3D aircraft configurations. *Computer Methods in Applied Mechanics and Engineering*, 101:225–249, 1992.

[8] C. Bischof, A. Carle, G. Corliss, A. Griewank, and P. Hovland. Generating derivative codes from Fortran programs. *Internal report MCS-P263-0991*, Computer Science Division, Argonne National Laboratory and Center of Research on Parallel Computation, Rice University, 1991.

[9] A. Buckley. A combined conjugate-gradient quasi-Newton minimization algorithm. *Math. Programming*, 15:200–210, 1978.

[10] G. W. Burgreen and O. Baysal. Aerodynamic shape optimization using preconditioned conjugate gradient methods. *AIAA paper 93-3322*, July 1993.

[11] G. W. Burgreen and O. Baysal. Three-dimensional aerodynamic shape optimization of wings using sensitivity analysis. *AIAA paper 94-0094*, 32nd Aerospace Sciences Meeting and Exhibit, Reno, Nevada, January 1994.

[12] J. Bussoletti, F. Johnson, M. Bieterman, R. Melvin, D. Young, and M. Drela. TRANAIR: solution adaptive CFD modeling for complex 3D configurations. In *1993 European Forum- Recent Developments and Applications in Aeronautical CFD, Royal Aeronautical Society*, pages 10.1–10.14, 1993.

[13] R. Campbell. An approach to constrained aerodynamic design with application to airfoils. *NASA Technical Paper 3260*, Langley Research Center, November 1992.

[14] R. Campbell. Efficient constrained design using Navier-Stokes codes. *AIAA paper 95-1808*, 13th Applied Aerodynamics Conference, San Diego, CA, June 1995.

[15] D. A. Caughey and A. Jameson. Numerical calculation of transonic flow about a wing-fuselage. *AIAA paper 77-677*, June 1977.

[16] Y. Crispin. Aircraft conceptual optimization using simulated evolution. *AIAA paper 94-0092*, 32nd Aerospace Sceinces Meeting and Exhibit, Reno, Nevada, Janaury 1994.

[17] M. A. Drela and M. B. Giles. Viscous inviscid analysis of transonic and low Reynolds number airfoils. *AIAA Journal*, 25(10):1347–1355, October 1987.

[18] M. E. Eleshaky and O. Baysal. Preconditioned domain decomposition scheme for three-dimensional aerodynamic sensitivity analysis. In *Proceedings of of the 12th AIAA computational fluid dynamics conference*, pages 1055–1056, July 1993.

[19] M. E. Eleshaky and O. Baysal. Shape optimization of a 3-D nacelle near a flat plate wing using multiblock sensitivity analysis. *AIAA paper 94-0160*, 32nd Aerospace Sciences Meeting and Exhibit, Reno, Nevada, January 1994.

[20] G. Farin. *Curves and Surfaces for Computer Aided Geometric Design: A Practical Guide*. Academic Press INC, San Diego CA, 1988.

[21] J. Fay. *On the Design of Airfoils in Transonic Flow Using the Euler Equations*. Ph.D. Dissertation, Princeton University 1683-T, 1985.

[22] K. Y. Fung, H. Sobieczky, and A. R. Seebass. Shock-free wing design. *AIAA paper 79-1557*, 1979.

[23] J. Gallman, J. Reuther, N. Pfeiffer, W. Forrest, and D. Bernstorf. Business jet wing design using aerodynamic shape optimization. *AIAA paper 96-0554*, 34th Aerospace Sciences Meeting and Exhibit, Reno, Nevada, January 1996.

[24] P. Garabedian and G. McFadden. Design of supercritical swept wings. In *Proceedings of 1980 Army Numerical Analysis and Computers Conference*, ARO Report 80-3, 1980.

[25] P. Garabedian and G. McFadden. Computational fluid dynamics of airfoils and wings. In *Proceedings of Symposium on Transonic, Shock, and Multidimensional Flows*, pages 1–16, Academic Press, New York, 1982.

[26] P. R. Garabedian and D. G. Korn. Numerical design of transonic airfoils. In B. Hubbard, editor, *Proceedings of SYNSPADE 1970*, pages 253–271, Academic Press, New York, 1971.

[27] M. Giles, M. Drela, and W. T. Thompkins. Newton solution of direct and inverse transonic Euler equations. In *Proceedings AIAA 7th Computational Fluid Dynamics Conference*, pages 394–402, Cininnati, Ohio, 1985. *AIAA paper 85-1530.*

[28] M. B. Giles and M. A. Drela. Two-dimensional transonic aerodynamic design method. *AIAA Journal*, 25(9):1199–1206, October 1987.

[29] P. Gill and W. Murray. Conjugate-gradient methods for large-scale nonlinear optimization. *SOL Tech. Report 79-15*, Systems Optimization Laboratory, Department of Operations Research, Stanford University, Stanford, California, 1979.

[30] P. Gill, W. Murray, and R. Pitfield. The implementation of two revised quasi-Newton algorithms for unconstrained optimization. *NAC 11*, National Physical Laboratory, Division of Numerical Analysis and Computing, 1972.

[31] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization*. Academic Press, 1981.

[32] L. L. Green, P. A. Newman, and K. J. Haigler. Sensitivity derivatives for advanced CFD algorithm and viscous modeling parameters via automatic differentiation. *AIAA paper 93-3321*, 11th AIAA Computational Fluid Dynamics Conference, Orlando, Florida, 1993.

[33] M. Hafez and D. Lovell. Entropy and vorticity corrections for transonic flows. *International Journal for Numerical Methods in Fluids*, 8:31–53, 1988. Also AIAA Paper 83–1926.

[34] A. G. Hansen. Generalized control volume analyses with application to the basic laws of mechanics and thermodynamics. *Bulletin of Mechanical Engineering Education*, 4:161–168, 1965.

[35] A. G. Hansen. *Fluid Mechanics*. John Wiley & Sons, New York, 1967.

[36] P. A. Henne. An inverse transonic wing design method. *AIAA paper 80-0330*, 1980.

[37] R. M. Hicks and P. A. Henne. Wing design by numerical optimization. *Journal of Aircraft*, 15:407–412, 1978.

[38] R. M. Hicks and P. A. Henne. Wing design by numerical optimization. *AIAA paper 79-0080*, 1979.

[39] R. M. Hicks, E. M. Murman, and G. N. Vanderplaats. An assessment of airfoil design by numerical optimization. *NASA TM X-3092*, Ames Research Center, Moffett Field, California, July 1974.

[40] J. Huan and V. Modi. Optimum design of minimum drag bodies in incompressible laminar flow using a control theory approach. *Inverse Problems in Engineering*, 1:1–25, 1994.

[41] J. Huan and V. Modi. Design of minimum drag bodies in incompressible laminar flow. Technical report, The Forum on CFD for Design and Optimization, (IMECE 95), San Francisco, California, November 1995.

[42] W. P. Huffman, R. G. Melvin, D. P. Young, F. T. Johnson, J. E. Bussoletti, M. B. Bieterman, and C. L. Hilmes. Practical design and optimization in computational fluid dynamics. *AIAA paper 93-3111*, AIAA 24th Fluid Dynamics Conference, Orlando, Florida, July 1993.

[43] A. C. Taylor III, G. W. Hou, and V. M. Korivi. Sensitivity analysis, approximate analysis, and design optimization for internal and external viscous flows. *AIAA paper 91-3083*, September 1991.

[44] A. Jameson. Iterative solution of transonic flows over airfoils and wings, including flows at Mach 1. *Communications on Pure and Applied Mathematics*, 27:283–309, 1974.

[45] A. Jameson. Numerical calculation of transonic flow past a swept wing by a finite volume method. In *Proceedings of the Third IFIP Conference on Computing Methods in Applied Science and Enginering*, December 1977.

[46] A. Jameson. Remarks on the calculation of transonic potential flow by a finite volume method. In *Proceedings of the Conference on Computational Methods in Fluid Dynamics, Inst. Math. and Applications*, January 1978.

[47] A. Jameson. Acceleration of transonic potential flow calculations on arbitrary meshes by the multiple grid method. *AIAA paper 79-1458*, Fourth AIAA Computational Fluid Dynamics Conference, Williamsburg, Virginia, July 1979.

[48] A. Jameson. Accelerated finite-volume calculation of transonic potential flows. In Rizzi and Viviand, editors, *Notes on Numerical Fluid Mechanics, Numerical Methods for the Computation of Inviscid, Transonic Flows with Shock Waves*, volume 3, Braunschweig, 1981. Vieweg and Sohn.

[49] A. Jameson. Steady state solution of the Euler equations for transonic flow. In *Proceedings of Symposium on Transonic, Shock, and Multidimensional Flows*, pages 30–37, Academic Press, New York, 1982.

[50] A. Jameson. Solution of the Euler equations for two dimensional transonic flow by a multigrid method. *Applied Mathematics and Computation*, 13:327–356, 1983.

[51] A. Jameson. Multigrid algorithms for compressible flow calculations. In W. Hackbusch and U. Trottenberg, editors, *Lecture Notes in Mathematics, Vol. 1228*, pages 166–201. Proceedings of the 2nd European Conference on Multigrid Methods, Cologne, 1985, Springer-Verlag, 1986.

[52] A. Jameson. A vertex based multigrid algorithm for three dimensional compressible flow calculations. In *the ASME Symposium on Numerical Methods for Compressible Flow*, Anaheim, 1986.

[53] A. Jameson. Successes and challenges in computational aerodynamics. *AIAA paper 87-1184-CP*, AIAA 8th Computational Fluid Dynamics Conference, Honolulu, Hawaii, 1987.

[54] A. Jameson. Aerodynamic design via control theory. *Journal of Scientific Computing*, 3:233–260, 1988.

[55] A. Jameson. Automatic design of transonic airfoils to reduce the shock induced pressure drag. In *Proceedings of the 31st Israel Annual Conference on Aviation and Aeronautics, Tel Aviv*, pages 5–17, February 1990.

[56] A. Jameson. Airfoils admitting non-unique solutions of the Euler equations. *AIAA paper 91-1625*, AIAA 22nd Fluid Dynamics, Plasmadynamics & Lasers Conference, Honolulu, Hawaii, June 1991.

[57] A. Jameson. Time dependent calculations using multigrid, with applications to unsteady flows past airfoils and wings. *AIAA paper 91-1596*, AIAA 10th Computational Fluid Dynamics Conference, Honolulu, Hawaii, June 1991.

[58] A. Jameson. Aerodynamic design methods. In *International Workshop on Solution Techniques for Large-Scale CFD Problems*, Montreal, September 1994. CERCA.

[59] A. Jameson. Optimum aerodynamic design via boundary control. In *AGARD-VKI Lecture Series, Optimum Design Methods in Aerodynamics*. von Karman Institute for Fluid Dynamics, 1994.

[60] A. Jameson and T. J. Baker. Solution of the Euler equations for complex configurations. *AIAA paper 83-1929*, AIAA 6th Computational Fluid Dynamics Conference, Danvers, MA, July 1983.

[61] A. Jameson and T. J. Baker. Multigrid solution of the Euler equations for aircraft configurations. *AIAA paper 84-0093*, AIAA 22th Aerospace Sciences Meeting, Reno, Nevada, January 1984.

[62] A. Jameson and T. J. Baker. Improvements to the aircraft Euler method. *AIAA paper 87-0452*, AIAA 25th Aerospace Sciences Meeting, Reno, Nevada, January 1987.

[63] A. Jameson, T. J. Baker, and N. P. Weatherill. Calculation of inviscid transonic flow over a complete aircraft. *AIAA paper 86-0103*, AIAA 24th Aerospace Sciences Meeting, Reno, Nevada, January 1986.

[64] A. Jameson and D. A. Caughey. Numerical calculation of transonic flow past a swept wing. *New York University ERDA report COO 3077-140*, May 1977.

[65] A. Jameson and J. Reuther. Control theory based airfoil design using the Euler equations. *AIAA paper 94-4272*, 5th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Panama City Beach, FL, September 1994.

[66] A. Jameson, W. Schmidt, and E. Turkel. Numerical solutions of the Euler equations by finite volume methods with Runge-Kutta time stepping schemes. *AIAA paper 81-1259*, January 1981.

[67] R. Kennelly. Improved method for transonic airfoil design-by-optimization. *AIAA paper 83-1864*, AIAA Applied Aerodynamics Conference, Danvers, Massachusetts, July 1983.

[68] V. M. Korivi, A. C. Taylor III, G. W. Hou, P. A. Newman, and H. E. Jones. Sensitivity derivatives for three-dimensional supersonic Euler code using incremental iterative strategy. In *Proceedings of the 12th AIAA computational fluid dynamics conference*, pages 1053–1054, July 1993.

[69] V. M. Korivi, A. C. Taylor III, and P. A. Newman. Aerodynamic optimization studies using a 3-D supersonic Euler code with efficient calculation of sensitivity derivatives. *AIAA paper 94-4270*, 5th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Panama City, Florida, September 1994.

[70] V. M. Korivi, A. C. Taylor III, P. A. Newman, G. W. Hou, and H. E. Jones. An incremental strategy for calculating consistent discrete CFD sensitivity derivatives. *NASA TM 104207*, Langley Research Center, Hampton, VA, February 1992.

[71] G. Kuruvila, S. Ta'asan, and M. D. Salas. Airfoil optimization by the one-shot method. In *AGARD-VKI Lecture Series, Optimum Design Methods in Aerodynamics*. von Karman Institute for Fluid Dynamics, 1994.

[72] J. M. Lacasse and O. Baysal. Design optimization of single- and two-element airfoils on multiblock grids. *AIAA paper 94-4273*, 5th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Panama City, Florida, September 1994.

[73] F. W. Lanchester. Personal letters of. *AIAA Journal*, June 1931.

[74] J. Lewis. *Inverse Aerodynmic Design Based on the Full Potential Equations Using Variational Calculus*. Maters Thesis, Washington University, 1994.

[75] J. Lewis and R. Agarwal. Airfoil design via control theory using the full-potential and Euler equations. Technical report, The Forum on CFD for Design and Optimization, (IMECE 95), San Francisco, California, November 1995.

[76] J. Lewis, G. Peters, and R. Agarwal. Airfoil design via control theory using the Euler equations. *ASME FED-Vol. 129*, Proceedings of the 1991 ASME Winter Annual Meeting, Atlanta, December 1991.

[77] M. J. Lighthill. A new method of two-dimensional aerodynamic design. *R & M 1111*, Aeronautical Research Council, 1945.

[78] J. L. Lions. *Optimal Control of Systems Governed by Partial Differential Equations*. Springer-Verlag, New York, 1971. Translated by S.K. Mitter.

[79] D. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Math. Programming*, 45:503–528, 1989.

[80] L. Martinelli. Calculation of viscous flows with a multigrid method. *Ph.D. Dissertation*, Mechanical and Aerospace Engineering Department, Princeton University, 1987.

[81] G. B. McFadden. An artificial viscosity method for the design of supercritical airfoils. *Internal report, and Ph.D. Thesis C00-3077-158*, New York University, 1979.

[82] M. M. Munk. General theory of thin wings sections. *NACA report 142*, 1922.

[83] P. A. Newman, G. W. Hou, H. E. Jones, A. C. Taylor III, and V. M. Korivi. Observations on computational methodologies for use in large-scale gradient-based, multidisciplinary design incorporating advanced CFD codes. *NASA TM 104206*, Langley Research Center, Hampton, VA, February 1992.

[84] Facilities Planning Office. Research facilities handbook. *NASA TM AHB 8801-1*, NASA Ames Research Center, Moffett Field, CA, August 1982.

[85] F. Penaranda and M. S. Freda. Aeronautical facilities catalogue, volume 1, wind tunnels. *NASA RP 1132*, NASA, Washington, DC, 1985.

[86] O. Pironneau. *Optimal Shape Design for Elliptic Systems*. Springer-Verlag, New York, 1984.

[87] O. Pironneau. Optimal shape design for aerodynamics. In *AGARD-VKI Lecture Series, Optimum Design Methods in Aerodynamics*. von Karman Institute for Fluid Dynamics, 1994.

[88] L. Prandtl. Applications of modern hydrodynamics to aeronautics. *NACA report TR-116*, 1922.

[89] P. Rai, L. R. Miranda, and A. R. Seebass. A cost-effective method of shock-free supercritical wing design. *AIAA paper 81-0383*, 1981.

[90] J. Reuther. *Practical Aerodynamic Optimization of Airfoils and Wings*. Masters Thesis, University of California Davis, Davis, California, 1991.

[91] J. Reuther, S. Cliff, R. Hicks, and C.P. van Dam. Practical design optimization of wing/body configurations using the Euler equations. *AIAA paper 92-2633*, 1992.

[92] J. Reuther and A. Jameson. Control theory based airfoil design for potential flow and a finite volume discretization. *AIAA paper 94-0499*, 32nd Aerospace Sciences Meeting and Exhibit, Reno, Nevada, January 1994.

[93] J. Reuther and A. Jameson. Aerodynamic shape optimization of wing and wing-body configurations using control theory. *AIAA paper 95-0123*, 33rd Aerospace Sciences Meeting and Exhibit, Reno, Nevada, January 1995.

[94] J. Reuther and A. Jameson. A comparison of design variables for control theory based airfoil optimization. Technical report, 6th International Symposium on Computational Fluid Dynamics, Lake Tahoe, Nevada, September 1995.

[95] J. Reuther and A. Jameson. Supersonic wing and wing-body shape optimization using an adjoint formulation. Technical report, The Forum on CFD for Design and Optimization, (IMECE 95), San Francisco, California, November 1995.

[96] J. Reuther, A. Jameson, J. Farmer, L. Martinelli, and D. Saunders. Aerodynamic shape optimization of complex aircraft configurations via an adjoint formulation. *AIAA paper 96-0094*, 34th Aerospace Sciences Meeting and Exhibit, Reno, Nevada, January 1996.

[97] J. Reuther, C. P. van Dam, and R. Hicks. Subsonic and transonic low-Reynolds-number airfoils with reduced pitching moments. *Journal of Aircraft*, 29:297–298, 1992.

[98] A. Rizzi and H. Viviand. Numerical methods for the computation of inviscid transonic flows with shock waves. In *Proc. GAMM Workshop*, Stockholm, 1979.

[99] Y. Saad and M. Schultz. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. SCi. Stat. Comput.*, 7:856–869, 1986.

[100] M. D. Salas, R. E. Melnik, and A. Jameson. A comparative study of the non-uniqueness problem of the potential equation. *AIAA paper 83-1888*, Proceedings of 6th AIAA Computational Fluid Dynamics Conference, Danvers, July 1983.

[101] D. Shanno and K. Phua. Remark on algorithm 500–a variable method subroutine for unconstrained nonlinear minimization. *ACM Trans. Math. Software*, 6:618–622, 1980.

[102] L. L. Sherman, A. C. Taylor III, L. L. Green, G. W. Hou, P. A. Newman, and V. M. Korivi. First- and second-order aerodynamic sensitivity derivatives via automatic differentiation with incremental iterative methods. *AIAA paper 94-4262*, 5th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Panama City, Florida, September 1994.

[103] G. R. Shubin. Obtaining cheap optimization gradients from computational aerodynamics codes. *Internal paper AMS-TR-164*, Boeing Computer Services, June 1991.

[104] G. R. Shubin and P. D. Frank. A comparison of the implicit gradient approach and the variational approach to aerodynamic design optimization. *internal paper AMS-TR-164*, Boeing Computer Services, April 1991.

[105] H. Sobieczky. Related analytical, analog and numerical methods in transonic airfoil design. *AIAA paper 79-1556*, 1979.

[106] H. Sobieczky, K. Y. Fung, and A. R. Seebass. A new method for designing shock-free transonic configurations. *AIAA paper 78-1114*, 1978.

[107] S. Ta'asan, G. Kuruvila, and M. D. Salas. Aerodynamic design and optimization in one shot. *AIAA paper 92-0025*, 30th Aerospace Sciences Meeting and Exhibit, Reno, Nevada, January 1992.

[108] A. M. Thomas, R. E. Smith, and S. N. Tiwari. Rational B-spline and PDE surfaces with unstructured grids for aerospace vehicle design. *AIAA paper 94-0419*, 32nd Aerospace Sciences Meeting and Exhibit, Reno, Nevada, January 1994.

[109] P. A. Thompson. *Compressible-Fluid Flow*. The Maple Press Company, 1984.

[110] T. L. Tranen. A rapid computer aided transonic airfoil design method. *AIAA paper 74-501*, 1974.

[111] G. Volpe. Two-element airfoil analysis and design: An inverse method. *AIAA paper 78-1226*, 1978.

[112] G. Volpe and R. E. Melnik. The role of constraints in the inverse design problem for transonic airfoils. *AIAA paper 81-1233*, 1981.

[113] G. Volpe and R. E. Melnik. The design of transonic aerofoils by a well posed inverse method. *International Journal of Numerical Methods in Engineering*, 22:341–361, 1986.

[114] W. Wright and O. Wright. *The Papers of Wilbur and Orville Wright*, volume I and II. Arno Press, 1972.

[115] D. P. Young, W. P. Huffman, R. G. Melvin, M. B. Bieterman, C. L. Hilmes, and F. T. Johnson. Inexactness and global convergence in design optimization. *AIAA paper 94-4286*, 5th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Panama City, Florida, September 1994.