NOTE TO USERS

Page(s) not included in the original manuscript are unavailable from the author or university. The manuscript was microfilmed as received.

41

This reproduction is the best copy available.

UMI

IMPLICIT NAVIER-STOKES COMPUTATIONS OF UNSTEADY FLOWS USING SUBITERATION METHODS

by

STAN DE RANGO

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

© by Stan De Rango, January 31, 1996

**National Library
of Canada**

**Acquisitions and
Bibliographic Services**

**395 Wellington Street
Ottawa ON K1A 0N4
Canada**

**Bibliothèque nationale
du Canada**

**Acquisitions et
services bibliographiques**

**395, rue Wellington
Ottawa ON K1A 0N4
Canada**

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-51537-0

Canada

Abstract

A time-accurate, implicit numerical method for the two-dimensional compressible thin-layer Navier-Stokes equations is described. Subiteration techniques are used to solve the non-linear problem arising at each time step. The subiteration methods have been incorporated in the well-known implicit Navier-Stokes flow solver ARC2D, developed at the NASA Ames Research Center. Two different subiteration techniques are used. They employ similar methodologies used to solve steady-state problems such as approximate factorization, diagonalization, and local time-stepping. One scheme uses a single time-step value while the other uses a dual time-stepping approach where one time-step governs the temporal accuracy of the code and the second time-step is chosen to accelerate convergence at each iteration. The subiteration schemes are second-order accurate in time and eliminate the linearization and factorization error characteristic of factored implicit time-marching methods. As a secondary benefit, the domain boundaries and wake cuts are treated implicitly without the need to change the current explicit boundary routines. Full coupling of the mean flow and the turbulence model is also obtained. Furthermore, increasing the order of temporal accuracy of the subiteration scheme is quite simple, a task which is very complex for implicit solvers using linearized, approximately-factored forms of the discretized equations.

Two modifications are made to the subiteration schemes to improve the efficiency of the code. The combined second and fourth-difference dissipation, which requires the inversion of a set of scalar pentadiagonal equations, is altered to use a modified second-difference dissipation scheme in the implicit operator, leading to a scalar tridiagonal form. The second modification consists of freezing the Jacobian matrices during the subiterations. The implementation of these modifications into either of the subiteration schemes is quite simple and can produce between 10% to 50% savings in computational work.

The flows studied exhibit self-induced unsteadiness. Laminar flow about a circular cylinder and over the NACA 0012 airfoil at low Reynolds number is computed. For the latter case, the subiteration schemes can use a time-step 20 times larger than a second-order, linearized, approximately-factored solver to obtain Strouhal numbers within 1% of the converged values, requiring less than $\frac{1}{23}$ of the cpu time when using the modified schemes.

Turbulent flow about an airfoil under buffet conditions is also considered. Turbulence is modeled using the Baldwin-Lomax and Spalart-Allmaras models. The Spalart-Allmaras model proved to be more robust than the Baldwin-Lomax model, but suffered from poor convergence characteristics in the subiteration process. Using the Baldwin-Lomax model, the subiteration schemes can be up to 38 times more efficient than the basic block pentadiagonal scheme for turbulent cases.

Acknowledgments

I would like to thank my supervisor, Professor D.W. Zingg, whose guidance and tutelage greatly shaped my knowledge and understanding of my research and studies. I would also like to extend my gratitude and appreciation to the members of the Computational Fluid Dynamics Group at UTIAS who seldom complained about my excessive use of CPU time and made my stay at UTIAS both interesting and enjoyable. I would also especially like to thank Philippe Godin, for having coded up the Spalart-Allmaras turbulence model and then generously allowing me to use it.

Funding from de Havilland Inc. and a University of Toronto Fellowship for one term is gratefully acknowledged.

To all my family members, I would like to express my gratitude for their ongoing support throughout my educational endeavors and I share my achievements with them.

Contents

List of Figures	v
List of Tables	vii
1 Introduction	1
1.1 Applications	1
1.2 Review of Numerical Methods for Unsteady Computations	2
1.2.1 Basic Equations	2
1.2.2 Computational Methods	3
1.3 Objective and Outline	6
2 Numerical Technique for Solving the Thin-Layer Navier-Stokes Equations	7
2.1 Governing Equations	7
2.2 Time Marching and Subiteration Algorithms	9
2.2.1 Single Time-step Subiteration Scheme	12
2.2.2 Dual Time-step Subiteration Scheme	13
2.3 Modified Subiteration Methods	14
2.4 Turbulence Model Implementation	16
3 Results and Discussion	17
3.1 Laminar Flow Over Circular Cylinder	17
3.2 Laminar Flow Over NACA 0012 Airfoil	19
3.3 Turbulent Flow Over NACA 0012 Airfoil	29
3.3.1 Baldwin-Lomax Turbulence Model	30
3.3.2 Spalart-Allmaras Turbulence Model	33
4 Conclusions	36
References	38
A Dual Time Stepping Scheme	41

List of Figures

3.1	Effect of subiterations and time-step on Strouhal number for circular cylinder. ($\mathcal{M} = 0.1, \mathcal{Re} = 120$)	18
3.2	Strouhal number vs. cpu work required per period using SI scheme for circular cylinder. ($\mathcal{M} = 0.1, \mathcal{Re} = 120$)	20
3.3	Strouhal number vs. cpu work required per period using SI scheme for circular cylinder. ($\mathcal{M} = 0.1, \mathcal{Re} = 120$)	20
3.4	Snapshot of enthalpy contours of unsteady flowfield for laminar flow over NACA 0012 airfoil. ($\mathcal{M} = 0.2, \mathcal{Re} = 800, \alpha = 20^\circ$)	21
3.5	169x49 C-mesh used for laminar flow over NACA 0012 airfoil.	22
3.6	Effect of subiterations and time-step on Strouhal number for NACA 0012 airfoil. ($\mathcal{M} = 0.2, \mathcal{Re} = 800, \alpha = 20^\circ$)	23
3.7	Strouhal number vs. cpu work required per period using SI scheme for NACA 0012 airfoil. ($\mathcal{M} = 0.2, \mathcal{Re} = 800, \alpha = 20^\circ$)	24
3.8	Strouhal number vs. cpu work required per period using SI scheme for NACA 0012 airfoil. ($\mathcal{M} = 0.2, \mathcal{Re} = 800, \alpha = 20^\circ$)	24
3.9	Coefficient of lift vs. nondimensional time using SI scheme for NACA 0012 airfoil. ($\mathcal{M} = 0.2, \mathcal{Re} = 800, \alpha = 20^\circ$)	25
3.10	Average coefficient of lift over one period vs. time-step using SI scheme for NACA 0012 airfoil. ($\mathcal{M} = 0.2, \mathcal{Re} = 800, \alpha = 20^\circ$)	26
3.11	Average coefficient of lift vs. cpu work per period using SI scheme for NACA 0012 airfoil. ($\mathcal{M} = 0.2, \mathcal{Re} = 800, \alpha = 20^\circ$)	27
3.12	Average coefficient of lift vs. cpu work per period using SI scheme for NACA 0012 airfoil. ($\mathcal{M} = 0.2, \mathcal{Re} = 800, \alpha = 20^\circ$)	27
3.13	Strouhal number vs. cpu work per period using SI, FES and SCTRI schemes for NACA 0012 airfoil. ($\mathcal{M} = 0.2, \mathcal{Re} = 800, \alpha = 20^\circ$)	29
3.14	Snapshot of density contours of unsteady flowfield for turbulent flow over NACA 0012 airfoil. ($\mathcal{M} = 0.7, \mathcal{Re} = 9 \times 10^6, \alpha = 6^\circ$)	30
3.15	Comparison of computed coefficient of lift vs. nondimensional time using BPD and SI schemes with Baldwin-Lomax turbulence model for the NACA 0012 airfoil. ($\mathcal{M} = 0.7, \mathcal{Re} = 9 \times 10^6, \alpha = 6^\circ$)	31
3.16	Comparison of computed coefficient of lift vs. nondimensional time using BPD, SI and FES schemes with Baldwin-Lomax turbulence model for the NACA 0012 airfoil. ($\mathcal{M} = 0.7, \mathcal{Re} = 9 \times 10^6, \alpha = 6^\circ$)	32

3.17 Comparison of computed coefficient of lift vs. nondimensional time using BPD, SI and dual time-stepping schemes with Spalart-Allmaras turbulence model for NACA 0012 airfoil. ($\mathcal{M} = 0.7$, $\mathcal{R}e = 9 \times 10^6$, $\alpha = 6^\circ$)	35
---	----

List of Tables

3.1 CPU savings when using FES scheme over the SI scheme.	28
---	----

Nomenclature

$\hat{A}, \hat{B}$	flux Jacobians
C_L	lift coefficient
$\hat{D}$	function of flux vectors
D	diameter of cylinder
$D_\xi^{(2)}$	second-order dissipation operator
$D_\xi^{(4)}$	fourth-order dissipation operator
$\hat{E}, \hat{F}$	flux vectors
G	nonlinear function of solution vector
I	identity matrix
J	Jacobian
$\hat{M}$	viscous flux Jacobian
$\hat{N}$	product of eigenvector matrices
$\hat{N}_d$	product of eigenvector matrices for dual time-stepping scheme
$\hat{Q}$	solution vector
$\hat{R}$	vector of residuals
$\hat{S}$	viscous flux vector
S	dual time-step constant
St	Strouhal number
T	eigenvector matrix
U, V	contravariant velocities
$\mathcal{M}$	Mach number
Re	Reynolds number
a	speed of sound
a_∞	free stream speed of sound
c	airfoil chord
p	pressure
t	nondimensional time
τ	pseudo-nondimensional time
Δt	time step
$\Delta \tau$	pseudo-time step
u, v	Cartesian velocities
x, y	physical coordinates
Δ	forward finite-difference operator
∇	backward finite-difference operator
∇	gradient operator
Λ	eigenvalue matrix
Υ	pressure term
α	angle of attack
δ_ξ	centred difference operator

δ_η	centred difference operator
ρ	density
ξ, η	computational coordinates
σ	spectral radius

Chapter 1

Introduction

Until high speed computers were developed, aircraft designers relied heavily on costly empirical modeling techniques, such as wind tunnel testing, to provide necessary design feedback. The modern designer, however, has a multitude of relatively inexpensive design tools at his or her disposal. The rapid advances in computer technology have prompted similar advances in computational methods to the extent that computers are now routinely used to complement the wind tunnel in the development of new aerospace vehicles. Traditionally, most of the effort in computational fluid dynamics (CFD) has been applied to steady or quasi-steady applications. The next generation of challenges in CFD, however, will be in the area of unsteady calculations.

1.1 Applications

There are many facets of our everyday lives which involve unsteady fluid phenomena. Weak shock waves produced by the whip of a towel, the very fact that we can hear each other speak, and traveling ripples of water when a pebble is dropped into a pool are examples of such cases around us everyday. The physical mechanisms behind these examples are well understood today and can be explained by simple mathematical models. But what of the following cases?

- Chemically reacting flow in an aeropropulsion system which is a rapidly changing in time.

- Complicated airflow over the wing of a transatlantic airliner as it encounters strong turbulence.
- The complex interaction of ocean waves against the hull of an aircraft carrier or of its rapidly moving propellers.
- The vortex shedding on the upper surface of an aircraft wing at high angles of attack.

These are all extremely complicated flows. The mathematical models which govern the physical interaction of the fluid particles, namely the Navier-Stokes equations, in general cannot be solved analytically. They must be solved numerically.

The occurrence of flutter within the flight envelope of an aircraft usually leads to structural failure and loss of the vehicle [1]. Aircraft service life can be significantly degraded by unforeseen dynamic loadings, such as buffet, and predictive capability for such off-design point loadings is essential for proper structural design and the determination of safe flight envelopes for all aerospace vehicles. Hence, there is a need to develop efficient, accurate numerical methods for use in unsteady applications.

1.2 Review of Numerical Methods for Unsteady Computations

1.2.1 Basic Equations

Computational methods have been pursued at a number of differing levels of physical approximation to the flow equations. Fluid dynamic flow models available for transonic unsteady aerodynamics computation include the nonlinear potential equation (both Transonic Small Disturbance and Full-Potential equations), the Euler equations and the Navier-Stokes (NS) equations (both full and thin-layer NS equations).

The nonlinear potential and Euler equations are inviscid models. The full-potential equation models the flow as a compressible irrotational fluid and is appropriate for transonic flows with weak shocks. The Euler equations model the flow as

a compressible, rotational fluid and can be used for flows with strong shock waves. Viscous flow capability may be added to the potential and Euler equations by means of interacted boundary-layer models. The NS equations are capable of representing mathematically the physical phenomena encountered in transonic flows, including mixed subsonic-supersonic flow, shock waves, boundary layers, and separation. A slight simplification of the NS equations is the thin-layer approximation. The unsteady thin-layer NS equations are the set of equations used in this research effort. In this approximation the viscous terms in the streamwise direction are neglected. The rationale behind this approximation is that the streamwise diffusion terms are small compared to the normal diffusion terms. Secondly, because of limitations on computer storage it is not possible to provide sufficient grid points to adequately resolve the viscous terms in all directions, only those in a thin layer close to the surface.

1.2.2 Computational Methods

Many unsteady problems of practical importance in aeronautical applications of computational fluid dynamics exhibit some degree of stiffness from both physical and numerical sources. Efficient solution of such problems requires implicit time-marching methods. These involve the solution of a nonlinear problem, in time, at each time step.

To enhance their efficiency, practical use, or tractability on conventional computers, implicit solvers often use linearized, approximately factored forms of the original equations. Although these simplifications retain second-order accuracy in time, they can introduce significant errors, thereby reducing the time step needed to achieve a given level of accuracy. Therefore, linearizing and factoring decrease the cost per time step but increase the total number of time steps required. Examples of implicit finite difference Navier-Stokes solvers are the methods of Briley and McDonald [2] and Beam and Warming [3].

Navier-Stokes flow solvers often use non-time-accurate techniques to converge rapidly to a steady state. A well proven convergence acceleration technique for a

steady explicit solver is the highly efficient multigrid time stepping technique, originally developed by Jameson [4]. A recent trend in unsteady calculations is the use of a “steady” (non-time-accurate) solver to solve the non-linear problem at each time-step. The residual vector can be modified to obtain a fully implicit time-accurate discretization of the NS equations as the residual tends to zero. The coupled set of implicit non-linear equations are solved iteratively, driving the residual to a predetermined tolerance at each time step. In this thesis, the use of an iterative method at each time step is referred to as a subiteration method. Explicit or implicit solvers can be used. Thus one can apply Newton’s method, a multigrid method, or an implicit method with convergence acceleration techniques associated with being non-time-accurate, such as diagonalization, local time-stepping, and local preconditioning.

The nonlinear problem to be solved within an implicit time-marching method has one important difference from a typical steady-state problem. The difference between the initial guess and the converged solution is much smaller than would normally be encountered in a steady problem. As a result, the number of subiterations required is much smaller than would be required for a steady computation. Thus the cost of each time step is increased, but since linearization and factorization errors are eliminated, the time step size can be increased while retaining a given level of accuracy. Examples of some early work using subiteration techniques to solve sets of non-linear equations at each time step are Chakravarthy and Osher [5] and Rai and Chakravarthy [6]. In these two cases the iterative methods were used in steady-state applications. It was not until recently that these methods were modified for use in unsteady applications.

In terms of CPU expense, a subiteration method is only worthwhile if the increased step size more than outweighs the increased cost per time step. However, there are secondary advantages. For example, implicit treatment of domain boundaries, wake cuts, and block boundaries (in a multi-block context) is easily achieved. Full coupling of the mean flow and the turbulence model is also obtained without linearizing the coupled system. Another advantage is the versatility of the subiteration schemes in terms of the order of the time-marching methods which can be used. One

can use an implicit time-discretization of arbitrary order with negligible impact on the required cpu work. Only the discretization of the time derivative in the residual vector is altered to obtain time-discretizations of higher order. Increasing the order of time-accuracy of a linearized, approximately factored implicit NS solver, without the use of subiterations, is a complex and demanding undertaking.

There are basically two categories for subiteration methods. The first category consists of those methods which only use a single time step value. An example is the subiteration method developed by Pulliam [7]. Pulliam describes how this method can be implemented in an implicit flow solver using a variety of convergence acceleration techniques. The second category consists of dual time-stepping algorithms where two separate time step values are used. One time step is used to govern the temporal accuracy of the scheme while the other time step is chosen appropriately to accelerate convergence of the subiteration process. Some examples are Belov et al. [8] and Alonso et al. [9]. The algorithms presented in these papers use a five stage explicit Runge-Kutta scheme with multigrid and finite volume spatial discretization. Other examples using implicit solvers are Shuen et al. [10], Arnone et al. [11], Venkateswaran and Merkle [12], and Rumsey et al. [13].

References [10] and [12] illustrate the use of local preconditioning with the dual time stepping approach. Rumsey et al. [13] uses both an explicit and implicit solver for the dual time-stepping scheme. This reference also includes results using the method of Pulliam [7]. Pulliam and Rumsey et al. demonstrate the improvement in efficiency of their unsteady NS solver when using subiteration schemes. However, they compare their results for the subiteration schemes to the diagonalized form (see reference [14]) of the NS equations, which is only first-order accurate in time. The work presented in this thesis compares the performance and accuracy of the subiteration schemes to a second-order accurate approximately factored solver, which is much more efficient than the first-order method.

1.3 Objective and Outline

The purpose of this thesis is to develop improved subiteration methods and to demonstrate their efficiency. The flows studied exhibit self-induced unsteadiness and are vortex dominated. The subiteration methods have been incorporated in the well-known implicit Navier-Stokes flow solver ARC2D [15]. Turbulence is modeled using the Baldwin-Lomax and Spalart-Allmaras models. Two subiteration methods are considered, that presented by Pulliam [7], and the dual-time-stepping method of Venkateswaran and Merkle [12]. After describing the basic methods, modifications which have been made are presented. Finally, results are presented demonstrating the efficiency of the basic subiteration methods and the modified methods relative to the linearized factored approach.

Chapter 2

Numerical Technique for Solving the Thin-Layer Navier-Stokes Equations

In this chapter the solution method for solving the thin-layer Navier-Stokes equations is presented. The governing equations are presented in section 2.1. The implicit approximately-factored method of Beam and Warming is presented in section 2.2 along with the diagonal form. The method of Beam and Warming is a well-established implicit technique for the solution of the compressible Navier-Stokes equations and is used in the NASA Ames airfoil code ARC2D. The two subiteration methods, which have been implemented in ARC2D, are discussed in sections 2.2.1 and 2.2.2. The modifications which were made to the subiteration methods are described in section 2.3.

2.1 Governing Equations

The governing equations are the two-dimensional thin-layer Navier-Stokes equations, written in generalized coordinates as

$$\partial_t \hat{Q} + \partial_\xi \hat{E} + \partial_\eta \hat{F} = \mathcal{R}e^{-1} \partial_\eta \hat{S} \quad (2.1)$$

where

$$\hat{Q} = J^{-1} \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ e \end{bmatrix} \quad (2.2)$$

The flux terms are

$$\hat{E} = J^{-1} \begin{bmatrix} \rho U \\ \rho U u + \xi_x p \\ \rho U v + \xi_y p \\ (e + p)U - \xi_t p \end{bmatrix}, \quad (2.3)$$

$$\hat{F} = J^{-1} \begin{bmatrix} \rho V \\ \rho V u + \eta_x p \\ \rho V v + \eta_y p \\ (e + p)V - \eta_t p \end{bmatrix} \quad (2.4)$$

with

$$U = \xi_t + \xi_x u + \xi_y v, \quad V = \eta_t + \eta_x u + \eta_y v \quad (2.5)$$

the contravariant velocities. The variable J represents the metric Jacobian of the transformation:

$$J^{-1} = (x_\xi y_\eta - x_\eta y_\xi) \quad (2.6)$$

The viscous terms are

$$\hat{S} = J^{-1} \begin{bmatrix} 0 \\ \eta_x m_1 + \eta_y m_2 \\ \eta_x m_2 + \eta_y m_3 \\ \eta_x (u m_1 + v m_3 + m_4) \\ + \eta_y (u m_2 + v m_3 + m_5) \end{bmatrix} \quad (2.7)$$

with

$$\begin{aligned}
m_1 &= \mu(4\eta_x u_\eta - 2\eta_y v_\eta)/3 \\
m_2 &= \mu(\eta_y u_\eta + \eta_x v_\eta) \\
m_3 &= \mu(-2\eta_x u_\eta + 4\eta_y v_\eta)/3 \\
m_4 &= \mu \mathcal{P} r^{-1} (\gamma - 1)^{-1} \eta_x \partial_\eta (a^2) \\
m_5 &= \mu \mathcal{P} r^{-1} (\gamma - 1)^{-1} \eta_y \partial_\eta (a^2)
\end{aligned} \tag{2.8}$$

Pressure is related to the conservative flow variables, $\hat{Q}$, by the equation of state for a perfect gas:

$$p = (\gamma - 1) \left(e - \frac{1}{2} \rho (u^2 + v^2) \right) \tag{2.9}$$

2.2 Time Marching and Subiteration Algorithms

The second-order backwards time-marching method is generally preferred over the trapezoidal method for the class of problems considered here. In delta form, the time marching algorithm takes the following form,

$$\Delta \hat{Q}^n = \frac{2}{3} \Delta t \left\{ \partial_t (\Delta \hat{Q}^n) + \partial_t \hat{Q}^n \right\} + \frac{1}{3} \Delta \hat{Q}^{n-1} \tag{2.10}$$

with $\Delta \hat{Q}^n = \hat{Q}^{n+1} - \hat{Q}^n$ and $\hat{Q}^n = \hat{Q}(n\Delta t)$. Equation 2.10 can also be written in the following fashion:

$$\frac{3\hat{Q}^{n+1} - 4\hat{Q}^n + \hat{Q}^{n-1}}{2\Delta t} + \tag{2.11}$$

$$\begin{aligned}
[\partial_\xi \hat{E}(\hat{Q}^{n+1}) + \partial_\eta \hat{F}(\hat{Q}^{n+1}) - \mathcal{R} e^{-1} \partial_\eta \hat{S}^{n+1}] &= G(\hat{Q}^{n+1}) \\
&= 0
\end{aligned}$$

which is the nonlinear relation we wish to satisfy at each time-step.

Applying equation 2.10 to the two-dimensional Navier Stokes equations and linearizing, one arrives at the following,

$$[I + \alpha \partial_\xi \hat{A}^n + \alpha \partial_\eta \hat{B}^n - \Delta t \mathcal{R} e^{-1} \partial_\eta J^{-1} \hat{M}^n] \Delta \hat{Q}^n = \hat{R}^n \quad (2.12)$$

where

$$\begin{aligned} \hat{R}^n = & -\alpha [\partial_\xi \hat{E}(\hat{Q}^n) + \partial_\eta \hat{F}(\hat{Q}^n) - \mathcal{R} e^{-1} \partial_\eta \hat{S}^n] \\ & + \frac{1}{3} \Delta \hat{Q}^{n-1}, \end{aligned} \quad (2.13)$$

and

$$\alpha = \frac{2}{3} \Delta t \quad (2.14)$$

The matrices $\hat{A}$, $\hat{B}$, and $\hat{M}$ are the flux Jacobians, defined by

$$\hat{A} = \frac{\partial \hat{E}}{\partial \hat{Q}}, \quad \hat{B} = \frac{\partial \hat{F}}{\partial \hat{Q}}, \quad \hat{M} = \frac{\partial \hat{S}}{\partial \hat{Q}}, \quad (2.15)$$

In general, equation 2.12 is prohibitively time consuming to solve directly. Various approximations can be made to the implicit operator in order to reduce the required computational time. Approximate factorization of equation 2.11, along with centered spatial differences, leads to the following form,

$$[I + \alpha \partial_\xi \hat{A}^n][I + \alpha \partial_\eta \hat{B}^n - \Delta t \mathcal{R} e^{-1} \partial_\eta J^{-1} \hat{M}^n] \Delta \hat{Q}^n = \hat{R}^n \quad (2.16)$$

Equation 2.16 is second-order accurate in time and conservative. With combined second and fourth-order implicit nonlinear dissipation, it requires the solution of a block pentadiagonal system of equations. Hence, this scheme is referred to as the BPD scheme from this point onward. It is the benchmark against which the subiteration schemes will be compared with respect to their accuracy and efficiency for unsteady applications.

The diagonal form of equation 2.16 is given by:

$$T_\xi[I + \alpha\partial_\xi\Lambda_\xi]\hat{N}[I + \alpha\partial_\eta\Lambda_\eta]T_\eta^{-1}\Delta\hat{Q}^n = \hat{R}^n \quad (2.17)$$

where

$$\hat{N} = T_\xi^{-1}T_\eta \quad (2.18)$$

The matrices Λ_ξ and Λ_η are diagonal matrices whose elements are the eigenvalues of the flux Jacobians. The matrix T_ξ has the eigenvectors of $\hat{A}$ as columns and T_η has the eigenvectors of $\hat{B}$ as columns. The viscous flux Jacobian $\hat{M}$ cannot be diagonalized with the flux Jacobian $\hat{B}$ so it has been dropped from the left hand side of equation 2.17. However, a term approximating the viscous eigenvalues is added to the diagonal, as described by Pulliam [15].

Artificial dissipation must be added to centered difference schemes to maintain stability. The combined second-fourth order non-linear dissipation scheme of Jameson et al. [16] is used. The fourth order dissipation provides the necessary stability and the second order dissipation prevents overshoot at shocks. Written as difference operators the second and fourth order dissipation terms are

$$\begin{aligned} D_\xi^{(2)} &= \nabla_\xi(\sigma J^{-1}\epsilon^{(2)})_{j+\frac{1}{2},k} \Delta_\xi J_{j,k} \\ D_\eta^{(2)} &= \nabla_\eta(\sigma J^{-1}\epsilon^{(2)})_{j,k+\frac{1}{2}} \Delta_\eta J_{j,k} \end{aligned} \quad (2.19)$$

$$\begin{aligned} D_\xi^{(4)} &= \nabla_\xi(\sigma J^{-1}\epsilon^{(4)})_{j+\frac{1}{2},k} \Delta_\xi \nabla_\xi \Delta_\xi J_{j,k} \\ D_\eta^{(4)} &= \nabla_\eta(\sigma J^{-1}\epsilon^{(4)})_{j,k+\frac{1}{2}} \Delta_\eta \nabla_\eta \Delta_\eta J_{j,k} \end{aligned} \quad (2.20)$$

The dissipation coefficients are given by

$$\begin{aligned} \epsilon_{j,k}^{(2)} &= \kappa_2 \Delta t \max(\Upsilon_{j+1,k}, \Upsilon_{j,k}, \Upsilon_{j-1,k}) \\ \epsilon_{j,k}^{(4)} &= \max(0, \kappa_4 \Delta t - \epsilon_{j,k}^{(2)}) \end{aligned} \quad (2.21)$$

where the pressure term Υ is defined to be

$$\Upsilon_{j,k} = \frac{|p_{j+1,k} - 2p_{j,k} + p_{j-1,k}|}{|p_{j+1,k} + 2p_{j,k} + p_{j-1,k}|} \quad (2.22)$$

and σ is the spectral radius scaling of the flux Jacobian matrices $\hat{A}$ and $\hat{B}$.

$$\sigma_{j,k} = \begin{cases} |U| + a\sqrt{\xi_x^2 + \xi_y^2} & , \quad \xi \text{ sweep} \\ |V| + a\sqrt{\eta_x^2 + \eta_y^2} & , \quad \eta \text{ sweep} \end{cases} \quad (2.23)$$

Pulliam [17] showed that the best rate of convergence for the Euler equations is achieved when matched artificial dissipation operators are included both implicitly and explicitly. Equation 2.17 with the dissipation operators included is

$$\begin{aligned} T_\xi \left[I + \alpha\delta_\xi \Lambda_\xi - \alpha D_\xi^{(2)} + \alpha D_\xi^{(4)} \right] \hat{N} \left[I + \alpha\delta_\eta \Lambda_\eta - \alpha D_\eta^{(2)} + \alpha D_\eta^{(4)} \right] T_\eta^{-1} \Delta \hat{Q}^n \\ = \hat{R}^n + \alpha(D_\xi^{(2)} - D_\xi^{(4)} + D_\eta^{(2)} - D_\eta^{(4)})\hat{Q} \end{aligned} \quad (2.24)$$

With combined second and fourth-order implicit nonlinear dissipation, the diagonal algorithm requires the solution of scalar pentadiagonal systems of equations rather than block pentadiagonal systems and has proven to be 3-4 times faster than the BPD scheme. Unfortunately, the diagonal algorithm is only first-order accurate in time and non-conservative. Hence it is not suitable for unsteady transonic flows. However, the diagonal form is ideally suited for use in a subiteration method.

2.2.1 Single Time-step Subiteration Scheme

The first subiteration scheme follows that developed by Pulliam [7]. The first step is to rewrite equation 2.10 substituting a new iterative index $p+1$ for $n+1$ terms. For simplicity, ignore the viscous terms in equation 2.1, and let

$$\hat{D}(\hat{Q}) = \partial_\xi \hat{E} + \partial_\eta \hat{F}. \quad (2.25)$$

Then by replacing $\partial_t \hat{Q}$ with $-\hat{D}(\hat{Q})$, ignoring higher order terms, and subtracting $\hat{Q}^p$ from both sides, the basic subiteration scheme (SI) becomes

$$\left[I + \frac{2}{3} \Delta t \hat{D}'(\hat{Q}^p) \right] \Delta \hat{Q}^p = -\frac{2}{3} \Delta t \hat{D}(\hat{Q}^p) - \left[\hat{Q}^p - \frac{4}{3} \hat{Q}^n + \frac{1}{3} \hat{Q}^{n-1} \right] \quad (2.26)$$

with $\hat{D}'(\hat{Q}^p) = \partial \hat{D} / \partial \hat{Q}$. In diagonal form, the subiteration scheme may be written as follows:

$$T_\xi [I + \frac{2}{3} \Delta t \partial_\xi \Lambda_\xi] \hat{N} [I + \frac{2}{3} \Delta t \partial_\eta \Lambda_\eta] T_\eta^{-1} \Delta \hat{Q}^p = -\frac{2}{3} \Delta t \hat{D}(\hat{Q}^p) - [\hat{Q}^p - \frac{4}{3} \hat{Q}^n + \frac{1}{3} \hat{Q}^{n-1}] \quad (2.27)$$

If the subiteration process converges, it will yield second-order time-accuracy, independent of the choice of $\hat{D}'(\hat{Q}^p)$. The n and $n-1$ terms are evaluated from previous time levels. After iterating p times the solution at time level $n+1$ will be taken from the most recent solution $\hat{Q}^{p+1}$. The resulting numerical scheme, at convergence gives

$$\frac{3\hat{Q}^{n+1} - 4\hat{Q}^n + \hat{Q}^{n-1}}{2\Delta t} + \hat{D}(\hat{Q}^{n+1}) = 0 \quad (2.28)$$

which is equivalent to equation 2.11.

2.2.2 Dual Time-step Subiteration Scheme

The second subiteration scheme is similar to the strategy employed by Venkateswaran and Merkle [12], and Rumsey et al. [13]. This scheme is the dual-time approach, wherein a ‘pseudo’ time-derivative is introduced in order to solve the nonlinear problem given by equation 2.11, as follows:

$$\frac{\partial \hat{Q}}{\partial \tau} = -G(\hat{Q}) \quad (2.29)$$

Applying implicit Euler time-marching to equation 2.29 gives

$$\frac{\hat{Q}^{p+1} - \hat{Q}^p}{\Delta \tau} = -\frac{3\hat{Q}^{p+1} - 4\hat{Q}^n + \hat{Q}^{n-1}}{2\Delta t} - \hat{D}(\hat{Q}^{p+1}) \quad (2.30)$$

where $\Delta \tau$ is the pseudo-time-step size and we have used equation 2.25 again. Thus the dual time-stepping scheme, in diagonal form, may be written as follows:

$$T_\xi [S + \Delta \tau \partial_\xi \Lambda_\xi] \hat{N}_d [S + \Delta \tau \partial_\eta \Lambda_\eta] T_\eta^{-1} \Delta \hat{Q}^p = -\Delta \tau G(\hat{Q}^p) \quad (2.31)$$

where

$$S = \left\{ \frac{3}{2\Delta t} + \frac{3\Delta\tau}{2\Delta t} \right\} I, \quad \hat{N}_d = T_\xi^{-1} S^{-1} T_\eta \quad (2.32)$$

For a detailed derivation of this scheme, the reader is referred to Appendix A. Once again, upon convergence of the inner iterations, equation 2.11 is satisfied.

2.3 Modified Subiteration Methods

The BPD scheme, represented by equation 2.16, is second-order accurate in time. It is the benchmark used here for unsteady calculations. The SI and dual time-stepping schemes eliminate linearization and factorization error. The subiteration schemes also allow the boundaries and turbulence models to be handled in a fully implicit fashion. The boundaries and turbulence values are updated at every subiteration. At the end of the subiteration process, the time lagged nature of the explicit boundary schemes in relation to the implicit update of the interior has effectively been eliminated. Similarly, the turbulence values are fully coupled to the current mean flow conditions.

Factorization and diagonalization are two examples of the approximations which can be made to the implicit operator in steady state applications. The subiteration techniques described above solve a non-linear problem at each time-step, much like a steady state solver. However, the subiteration problem has certain differences from the steady state problems usually encountered in CFD. Since we are advancing the mean flow field in time by Δt , and Δt is relatively small for unsteady calculations, the mean flow field is not changing by very much between time-steps. Hence, it generally requires only a few subiterations to drive the subiteration process to a pre-determined residual tolerance at each time-step. Consequently, inexact linearization of the dissipation model and other approximations that are used in the implicit operator, which would hinder convergence to steady state, do not affect the convergence of the subiterations in the same manner.

One can choose any implicit operator as long as the subiteration process converges. We have made the following two additional approximations to reduce the

CPU expense of the subiterations: 1) Second-difference dissipation is used in the implicit operator, leading to a scalar tridiagonal form, 2) The Jacobian matrices are frozen during the subiterations. The implementation of these modifications into either of the subiteration schemes is quite simple and can produce between 10% to 50% savings in computational work.

ARC2D uses the combined second and fourth order non-linear dissipation scheme of Jameson et al. [16]. The fourth order dissipation uses a 5 point stencil which forces the solution of equation 2.27 and equation 2.31 to require scalar pentadiagonal inversions. The dissipation terms, written as difference operators, are given by equations 2.19 and 2.20. The dissipation scheme on the implicit side is modified to reduce the scalar pentadiagonal to a scalar tridiagonal matrix. The original non-linear dissipation scheme is still used on the explicit, or right hand side of equation 2.27 and equation 2.31. The second order coefficients on the implicit side are replaced by a value equal to the second order coefficients on the explicit side plus twice the fourth order coefficients. The fourth order coefficients are subsequently set to zero, eliminating the use of the 5 point stencil:

$$\begin{aligned} D_{implicit}^{(2)} &= D_{explicit}^{(2)} + 2D_{explicit}^{(4)} \\ D_{implicit}^{(4)} &= 0 \end{aligned} \tag{2.33}$$

This scheme will be referred to as the scalar tridiagonal subiteration scheme (SCTRI) from this point onward.

The computational effort for inverting a banded matrix is scaled approximately by the square of the bandwidth. Hence a pentadiagonal system will take roughly 25/9 more computational effort to invert than a tridiagonal system. The effect of inexact linearization of the dissipation terms on the convergence of the subiteration scheme is investigated in the next chapter.

Inversion of the banded matrix constitutes a significant portion of the computational effort in solving for $\Delta\hat{Q}$. To avoid repetitive inversions of a matrix which does not change very much during the subiteration process, the eigenvalues of the

flux Jacobians are frozen during the subiterations at each time-step. That is, an LU decomposition is used to decompose and store $[S + \Delta\tau\partial_\xi\Lambda_\xi]$ in equation 2.31, for instance, after the first iteration at each time-step. The same is done for the η sweep. The eigenvector matrices denoted by T_ξ and T_η , however, are not frozen. The savings in computational effort can be rather significant and depends on the number of subiterations to be performed. There is computational overhead to contend with in decomposing and storing large amounts of data. The initial overhead becomes increasingly less significant as the number of subiterations are increased. The convergence rates and computational savings of this scheme, which we designate the frozen eigenvalue subiteration scheme (FES), will be discussed further in the next section.

2.4 Turbulence Model Implementation

The algebraic Baldwin-Lomax [18] and one-equation Spalart-Allmaras [19] turbulence models are described in detail in the given references. Both models are decoupled from the Navier-Stokes equations. The resulting partial-differential equation, in the Spalart-Allmaras model, is solved implicitly using an approximate factorization method. As described in reference [19], the turbulence model is first-order accurate in time and uses a subiteration technique to reduce factorization error. The subiteration technique used, however, differs from those described in the previous section. Both the single time-step and dual time-step subiteration schemes have been implemented in the Spalart-Allmaras turbulence model. These subiteration schemes improve the order of accuracy to second-order in time. All three subiteration schemes are compared for efficiency, accuracy and their effect on the overall convergence rate in the subiteration process.

Chapter 3

Results and Discussion

In this chapter, the subiteration schemes are evaluated. In order to assess the tradeoffs involved in using subiteration schemes, simulation of low Reynolds number laminar flows around a cylinder and airfoil are considered. Turbulent results are presented for the NACA 0012 airfoil under buffet conditions. All turbulent results presented use either the Baldwin-Lomax or the Spalart-Allmaras turbulence model.

3.1 Laminar Flow Over Circular Cylinder

For a Reynolds number in the range of $40 < Re < 5000$, laminar flow about a cylinder exhibits a double row of alternating vortices behind the cylinder, the upper row turning clockwise and the lower row turning counterclockwise. This phenomenon is formally referred to as a *von Karman vortex street*. The Strouhal number is generally used to characterize the flow. It is defined as $St = (nD^2)/(\nu Re)$, where n represents the frequency of the shed vortices, D the diameter of the cylinder and ν the kinematic viscosity. Figure 3.1 shows results comparing the computed Strouhal number vs. time-step for the SI scheme using a range of 1 to 5 subiterations. The grid used is a 140×90 O-mesh with minimum spacing at the body of 0.001 diameters and a far-field extent of 16 diameters. The full Navier Stokes equations are used since we are dealing with a bluff-body flow.

Figure 3.1: Effect of subiterations and time-step on Strouhal number for circular cylinder. ($\mathcal{M} = 0.1$, $Re = 120$)

For the given grid, both the SI and dual time-stepping schemes converge to the same St of approximately 0.1831. There is a large variation in the predicted St with decreasing Δt for the first order diagonal algorithm, where adequate results are only found for $\Delta t \leq 0.05$ if a Strouhal number to within 1% of the converged value is desired. The BPD scheme was not used in this case due to the lack of a periodic block-pentadiagonal solver. The efficiency and time-accuracy of the BPD scheme will be investigated in the next case study. It should be noted that the time-steps quoted and the time scale on the axis are nondimensionalized time values. The real time values are scaled by $\frac{a_\infty}{D}$.

By increasing the number of subiterations, one can use a larger timestep to achieve the same relative accuracy in the computed Strouhal number. For the two-subiteration case (a total of three sweeps of the mean flow solver, i.e., the main iteration and the two subiterations), sufficiently accurate results are obtained at $\Delta t \approx 0.5$. The time step used is a factor of ten larger than the one used by the diagonal algorithm for the same level of accuracy.

As the number of subiterations increases, the computational work per iteration also increases. It would be advantageous if this increase in work is more than offset by the larger time-step that one is able to use. Figure 3.2 shows the computed St using the SI scheme compared to the required cpu work per period, where 1 work unit is equivalent to 100 sweeps of the diagonal algorithm.

The SI scheme clearly demonstrates its efficiency and accuracy when compared to the diagonal form. Each subiteration reduces the residual by 1 to 3 orders of magnitude. Figure 3.3 is a close-up view of the previous figure. It is desirable to determine the optimum number of subiterations resulting in the least amount of cpu work for a given level of accuracy. Due to its non-asymptotic nature, the SI scheme using only 1 subiteration is not considered the optimum scheme despite its clear advantage in efficiency. Consequently, 2 subiterations is the optimum number of subiterations for this particular case. For St within 1% of the converged value the SI with 2 subiterations gives an overall gain of a factor of 3.3 in computational speed over the diagonal form alone. ARC2D is optimized for compressible transonic flows. As we move towards that realm, the convergence in the subiteration schemes should improve, consequently improving its efficiency as well. The results for the dual time-stepping scheme are not shown since the results are extremely similar to those of the SI scheme.

3.2 Laminar Flow Over NACA 0012 Airfoil

Laminar flow over airfoils at low Reynolds numbers and high angles of attack can also produce vortex shedding. In this section, laminar flow over the NACA 0012 airfoil is computed at a Mach number of $\mathcal{M} = 0.2$, $Re = 800$ based on chord length, and an angle of attack of 20° . At higher Reynolds numbers, the results actually exhibit a progression from single period vortex shedding to chaotic unsteadiness [20]. Figure 3.4 depicts the flowfield at the moment a vortex is about to shed off the trailing edge of the airfoil and subsequently convected downstream.

Figure 3.2: Strouhal number vs. cpu work required per period using SI scheme for circular cylinder. ($\mathcal{M} = 0.1$, $Re = 120$)

Figure 3.3: Strouhal number vs. cpu work required per period using SI scheme for circular cylinder. ($\mathcal{M} = 0.1$, $Re = 120$)

Figure 3.4: Snapshot of enthalpy contours of unsteady flowfield for laminar flow over NACA 0012 airfoil. ($M = 0.2$, $Re = 800$, $\alpha = 20^\circ$)

A 169x49 C-mesh was used for this case similar to the one used by Pulliam [20]. There are 50 points on the upper and lower surfaces of the airfoil and 35 in the wake. The normal spacing at the body is 0.01 with a far-field grid extent of 12 chords. The clustering along the surface consists of surface spacing of 0.005 at the leading edge and 0.001 at the trailing edge. Figure 3.5 shows some views of the grid.

For the given grid, all schemes converge to the same St number, based on chord, of approximately 0.5192. Results from the BPD scheme have been included in Figure 3.6 where the computed St is plotted versus the time-step used. The smallest time-step used for the BPD scheme is 5×10^{-4} . As in the previous case, increasing the number of subiterations allows a larger time-step to be used to obtain a given level of accuracy.

Figure 3.7 shows the computed St versus the relative cpu work required per period. One work unit is equivalent to 200 sweeps of the diagonal algorithm. For this case the optimum number of subiterations is not as evident as it is for the flow over the circular cylinder. Figure 3.7 does clearly illustrate, however, that for a given level of cpu work, regardless of the number of subiterations used, the SI scheme is

Figure 3.5: 169x49 C-mesh used for laminar flow over NACA 0012 airfoil.

Figure 3.6: Effect of subiterations and time-step on Strouhal number for NACA 0012 airfoil. ($\mathcal{M} = 0.2$, $Re = 800$, $\alpha = 20^\circ$)

considerably more efficient and accurate than the BPD scheme. To obtain a St within 1% of the converged value, the SI scheme using either 3 or 4 subiterations is 16 times more efficient than the BPD scheme. As the desired level of accuracy decreases, the benefits of using the subiteration schemes also diminish. To obtain a St within 5% of the converged value, the SI scheme is 6 times more efficient than the BPD scheme. This is roughly true regardless of the number of subiterations used. Note, however, that due to the large slopes in the curves for the SI scheme, little cpu work would be forfeited for the improved accuracy of 1% over 5% error.

Figure 3.8 is a close up view of the results showing the St in a range of 2% of the converged value. The results using 3 and 4 subiterations are extremely similar in terms of the computed St for a given level of cpu work. The run using 4 subiterations marginally outperformed the case using 3 subiterations in terms of efficiency. The case using 3 subiterations, however, is deemed to be the optimum number of subiterations.

The reason for this choice lies in the fact that the computed frequency of the unsteady flow does not fully represent the accuracy of the numerical scheme. Even if the subiteration process converges, the numerical error involved in the discretization

Figure 3.7: Strouhal number vs. cpu work required per period using SI scheme for NACA 0012 airfoil. ($\mathcal{M} = 0.2$, $Re = 800$, $\alpha = 20^\circ$)

Figure 3.8: Strouhal number vs. cpu work required per period using SI scheme for NACA 0012 airfoil. ($\mathcal{M} = 0.2$, $Re = 800$, $\alpha = 20^\circ$)

of the time derivative is still proportional to $(\Delta t)^2$. Hence, although one is able to use a larger time-step using 4 subiterations to obtain the same frequency as computed using 3 subiterations, the local accuracy of the result is affected. Figure 3.9 illustrates this trend. The SI scheme using 4 subiterations and $\Delta t = 0.06$ requires 17% less cpu work than the SI scheme using 3 subiterations with $\Delta t = 0.04$ yet both compute the same St of 0.5183. Examining Figure 3.9 closely, however, one finds that the case using 4 subiterations and $\Delta t = 0.06$ has a significant local error in terms of maximum, minimum, and average C_L over one period compared to the 3 subiteration case. Using 4 subiterations with $\Delta t = 0.08$ results in $St = 0.5176$, within $\frac{1}{2}\%$ of the converged value, with a savings in cpu work of 37% over the 3 subiteration case yet

Figure 3.9: Coefficient of lift vs. nondimensional time using SI scheme for NACA 0012 airfoil. ($M = 0.2$, $Re = 800$, $\alpha = 20^\circ$)

suffers from larger local error as shown in Figure 3.9. Figure 3.10 further illustrates the degradation of local accuracy as the time-step gets larger. The average C_L value over one period increases as the time-step increases, deviating from the converged

value. Hence, if two different combinations of time-step and number of subiterations used require approximately the same amount of cpu work, then the combination using the smallest time-step should be used. It may be possible to use larger time-steps if the discretization of the time derivative is of a higher order. The faster combinations of time-step and number of subiterations could then be used. One would only incur a slight penalty in memory storage to store the added solution vectors.

If greater than 5% error in St is acceptable, then 2 subiterations, as opposed to 3 subiterations, becomes the optimum number for the SI scheme. At that level of accuracy the difference in cpu work becomes negligible between the two schemes. In light of the discussion above, 2 subiterations is the optimum choice.

Figure 3.10: Average coefficient of lift over one period vs. time-step using SI scheme for NACA 0012 airfoil. ($\mathcal{M} = 0.2$, $Re = 800$, $\alpha = 20^\circ$)

Figure 3.11 shows the average C_L values versus the cpu work required per period. The SI scheme is clearly superior to the BPD scheme, independent of the number of subiterations used. For a given level of cpu work, the SI scheme achieves significantly better local accuracy than the BPD scheme. Figure 3.12 is a close up view of the results.

Figure 3.11: Average coefficient of lift vs. cpu work per period using SI scheme for NACA 0012 airfoil. ($\mathcal{M} = 0.2$, $Re = 800$, $\alpha = 20^\circ$)

Figure 3.12: Average coefficient of lift vs. cpu work per period using SI scheme for NACA 0012 airfoil. ($\mathcal{M} = 0.2$, $Re = 800$, $\alpha = 20^\circ$)

Number of Subiterations	Savings over SI scheme
2	23%
3	27%
4	30%
5	30%

Table 3.1: CPU savings when using FES scheme over the SI scheme.

The SCTRI scheme using the modified dissipation stencil has proven to be numerically stable and its effect on the convergence rate of the subiteration schemes is minimal. The subroutines used to invert the pentadiagonal diagonal systems in ARC2D constitute 18% of the total computational effort for one mean flow iteration. The new dissipation scheme is able to produce an overall savings of approximately 10%. Both the SCTRI and FES schemes produce extremely similar results as the SI scheme. The savings in computational effort realized when using the FES scheme depends on the number of subiterations performed.

Table 3.1 tabulates the cpu savings obtained when using the FES scheme over the SI scheme with respect to the number of subiterations used. Up to 30% of the cpu work can be saved when using 4 subiterations. Figure 3.13 compares all three schemes in terms of St versus cpu work per period. To obtain St within 1% of the converged value, the FES scheme using 3 subiterations is 23 times more efficient than the BPD scheme. The time-step used by the FES scheme is 20 times greater than that used by the BPD scheme. To obtain St within 5% of the converged value, the FES scheme is 8 times more efficient than the BPD scheme. The SCTRI and FES schemes can be combined to give additional savings. This was not done in this study. It requires an LU decomposition routine for a scalar tridiagonal system of equations.

It should be noted that the subiteration schemes are able to handle transients much more efficiently than the BPD scheme. For unsteady problems, initial condition dependent transients may take a long time to relax and time steps for tracking unsteady features may be restricted [7]. There can be considerable savings in cpu time over the BPD scheme as the subiteration schemes reach periodic steady-state much more quickly.

Figure 3.13: Strouhal number vs. cpu work per period using SI, FES and SCTRI schemes for NACA 0012 airfoil. ($\mathcal{M} = 0.2$, $Re = 800$, $\alpha = 20^\circ$)

The dual time-stepping scheme, with and without the modifications, produced results within plotting accuracy of the results for the SI scheme and hence are not shown. The dual time-stepping scheme does have one advantage over the SI scheme. It is possible to use preconditioning techniques, as described in references [10] and [12], to help improve convergence characteristics in the subiteration process. Preconditioning was not used in this study.

3.3 Turbulent Flow Over NACA 0012 Airfoil

Turbulent flow over a NACA 0012 airfoil is computed at $\mathcal{M} = 0.7$, $Re = 9$ million based on chord length, and an angle of attack of 6° . At these conditions, the airfoil experiences self-induced periodic shock oscillations. The unsteady shock interaction with the boundary layer periodically generates vortices just aft of the shock. These vortices are subsequently convected downstream. Figure 3.14 depicts the flowfield with one vortex approaching the trailing edge and the other in the process of being

convected downstream. Note that separation of the flow begins at the location of the shock, where it interacts with the boundary layer.

Figure 3.14: Snapshot of density contours of unsteady flowfield for turbulent flow over NACA 0012 airfoil. ($\mathcal{M} = 0.7$, $Re = 9 \times 10^6$, $\alpha = 6^\circ$)

3.3.1 Baldwin-Lomax Turbulence Model

All cases involving the Baldwin-Lomax model were run on a 385×65 C-mesh with a farfield extent of 25 chord lengths. There are 320 points on the airfoil surface and 33 in the wake. The minimum normal spacing at the wall is 10^{-5} chord lengths.

Figure 3.15 compares four particular runs. Two runs used the BPD scheme. The other two used the SI scheme with 2 and 4 subiterations, respectively. Using a time-step of $\Delta t = 2.5 \times 10^{-3}$, two subiterations are sufficient to reduce the density residual to less than 10^{-8} . Instead of restricting the number of subiterations to a specific number, one could simply set a residual tolerance level. For the comparisons we make here, it is more convenient to compare cpu requirements if the number of subiterations remains fixed. At 2 subiterations, the SI, dual time-stepping and BPD

schemes require approximately the same amount of cpu time for each mean flow iteration. Hence, from Figure 3.15, one notes that for the same computational effort per mean flow iteration, the SI scheme produces results which are very close to those produced by BPD scheme using a time-step which is 16 times smaller. The BPD

Figure 3.15: Comparison of computed coefficient of lift vs. nondimensional time using BPD and SI schemes with Baldwin-Lomax turbulence model for the NACA 0012 airfoil. ($\mathcal{M} = 0.7$, $Re = 9 \times 10^6$, $\alpha = 6^\circ$)

scheme using a time-step of $\Delta t = 2.5 \times 10^{-3}$ produces rather poor results. Using a time-step of $\Delta t = 5 \times 10^{-3}$ and four subiterations, the SI scheme is able to achieve accurate results when compared with the BPD results which use a time step that is 32 times smaller. The computational work per mean flow iteration increases by a factor of approximately 1.7 for a gain in computational speed of a factor of 19. Results using the dual time-stepping scheme are not shown since they are extremely similar to those of the SI scheme in both accuracy and efficiency.

As in the laminar case, the modified dissipation scheme has also proven to be numerically stable for turbulent flows. Its effect on the convergence rate of the subiteration schemes is minimal. The new dissipation scheme is able to produce an overall savings of approximately 10%. The computed results using the modified dissipation scheme are extremely similar to that of the SI scheme and are therefore not shown.

Figure 3.16 compares the SI scheme with the frozen eigenvalue subiteration scheme. Both schemes used 4 subiterations. The FES scheme required 50% less cpu time than the SI scheme using the same timestep and the same number of subiterations. The savings in computational effort depend on the number of subiterations performed. Although the results compare well, there is a noticeable reduction in accuracy when using the FES scheme. Further study is required to assess these tradeoffs.

Figure 3.16: Comparison of computed coefficient of lift vs. nondimensional time using BPD, SI and FES schemes with Baldwin-Lomax turbulence model for the NACA 0012 airfoil. ($M = 0.7$, $Re = 9 \times 10^6$, $\alpha = 6^\circ$)

The FES scheme provides a savings of 50% in cpu work, over the SI scheme, for turbulent flow and only 30% for laminar flow. The reason lies in the flow solver used by ARC2D. For the laminar case involving both the cylinder and airfoil, the wake cut boundary was solved explicitly. For the unsteady turbulent cases investigated here, however, the Baldwin-Lomax turbulence model exhibited instabilities in the wake cut region if it was solved explicitly. This limited the time-step to values one to two orders of magnitude smaller than those shown in the previous two figures. When solving the wake cut explicitly, the subiteration schemes were able to use much larger time-steps than the BPD scheme but the upper time-step limit was on the order of 10^{-3} . ARC2D is capable of solving the wake cut implicitly for C-mesh topologies. It does so by shuffling all of its vectors and matrices in such a manner as to solve the flowfield in two separate blocks, the airfoil block and the wake block respectively. Although such a procedure is computationally expensive, it does make the existing code more robust and allows for a much larger time-step to be used. When using the FES scheme and solving the wake cut boundary implicitly, the shuffling is only performed once per iteration and the values stored using LU decomposition. Hence, considerable cpu work is saved by not having to perform the shuffling for each subiteration. The result is an added 20% savings in cpu time making the FES scheme 2 times faster than the SI scheme using 4 subiterations.

3.3.2 Spalart-Allmaras Turbulence Model

The one-equation Spalart-Allmaras turbulence model produces significantly different results when compared to the algebraic Baldwin-Lomax model. The reason for this is not yet known. The results for two periods are shown in Figure 3.17, where the SI and dual time-stepping schemes used only 2 subiterations. This case was run on a 151x50 C-mesh topology with 112 points on the body, 20 points in the wake and a grid extent of 18 chordlengths. Although this grid is relatively coarse compared to the one used with the Baldwin-Lomax turbulence model, the same qualitative results were obtained with finer grids. Both the SI and the dual time-stepping schemes do not perform well when using the Spalart-Allmaras model. They converge much more

slowly in the subiteration process than when using the Baldwin-Lomax model. The third and subsequent subiterations are ineffective at reducing the residual any further; hence only 2 subiterations were used.

This phenomenon may be attributed to the fact that the Spalart-Allmaras turbulence model is only first-order accurate in time, as originally implemented in reference [19]. The original implementation uses an approximately-factored form of the discretized partial-differential equation with implicit Euler time-marching, which is first-order accurate. It utilizes a subiteration scheme to reduce factorization error. This subiteration scheme does not affect the order of accuracy of the time marching. The model was altered to use a second-order backwards time-marching method like the one used by the mean flow solver in the BPD scheme. This degraded the convergence of the SI and dual time-stepping schemes for the mean flow even further. The SI and dual time-stepping schemes were then implemented in the turbulence model. The original subiteration scheme was removed since the SI and dual time-stepping schemes also reduce factorization error. The convergence of the subiteration process in the turbulence model is extremely poor, requiring hundreds of subiterations to reduce the residual by 3 to 4 orders of magnitude. The convergence of the subiteration process of the mean flow solver does not fare any better, requiring dozens of subiterations to reduce the residual to the order of 10^{-8} , a task often accomplished in one or two subiterations when using the Baldwin-Lomax model.

Detailed analysis of the components contributing to the residual vector (the explicit side of the discretized equations) revealed that the correction terms, which increase the order of time-accuracy to second-order are much larger than the contributions from the turbulence model itself. Hence, the turbulent quantities were swamped out by these corrections, hindering the convergence of the subiteration scheme within the turbulence model. One possible solution to this problem, although not attempted here, is to time-march the turbulence model with smaller time-steps for every one mean flow subiteration. By marching the turbulence values by small increments in time, the correction terms themselves will be small and may not dominate the computation.

Figure 3.17: Comparison of computed coefficient of lift vs. nondimensional time using BPD, SI and dual time-stepping schemes with Spalart-Allmaras turbulence model for NACA 0012 airfoil. ($\mathcal{M} = 0.7$, $Re = 9 \times 10^6$, $\alpha = 6^\circ$)

Despite the difficulties encountered when using the Spalart-Allmaras model, it did offer one distinct advantage over the Baldwin-Lomax model. The Spalart-Allmaras model is more robust and has a larger upper stability limit, in terms of the time-step which can be used. If the wake cut boundary is solved explicitly, as it is for the results shown in Figure 3.17, the difference in the upper time-step limit between the two models can be one to two orders of magnitude, depending on the grid used. The Spalart-Allmaras model is able to use a larger time-step with the wake cut boundary solved explicitly than the Baldwin-Lomax model with the wake cut solved implicitly. Hence, if the problems described above can be resolved, then the Spalart-Allmaras turbulence model appears to be better suited for unsteady applications which require extremely long run times due to the small time-steps which must be used. Furthermore, it is quite likely that the Spalart-Allmaras results are much more accurate than the Baldwin-Lomax results.

Chapter 4

Conclusions

Two different subiteration methods have been examined for use in unsteady Navier-Stokes applications. They apply a method normally used to solve steady-state problems to the non-linear problem arising at each time step of an implicit time-marching method. The subiteration schemes eliminate errors resulting from linearization and factorization. Furthermore, implicit treatment of domain boundaries and full coupling of the mean flow and the turbulence model is also obtained.

Modifications were implemented in both schemes to reduce the cpu expense of the subiterations. Modifications to the dissipation scheme in the implicit operator lead to a 10% savings in cpu work with a negligible effect on the convergence rate. Freezing of the eigenvalues of the flux Jacobians in the subiterations alleviated the need to invert a large matrix in each subiteration. The result is a 30-50% savings in computational work while still achieving the same relative accuracy as the original subiteration scheme.

The two subiteration methods were applied to both laminar and turbulent cases. The Baldwin-Lomax and the Spalart-Allmaras models were used to model the turbulence. The results indicate that the increased step size which is able to be used with the subiteration methods while retaining a given level of accuracy more than offsets the increase in the cost of each time-step.

In a specific laminar case, the SI and dual time-stepping scheme is 23 times more efficient than the BPD scheme. The subiteration schemes were able to use a

time-step 20 times larger than the one used by the BPD scheme to obtain a Strouhal number within 1% of the converged value. This indicates that linearization and factorization errors contribute significantly to the overall temporal error in the BPD method. The dual time-stepping and SI schemes produced similar results in all test cases and required roughly the same amount of cpu time. In a specific turbulent case, the subiteration schemes produced savings in cpu time up to a factor of 38 relative to the BPD scheme for equivalent accuracy.

The use of different turbulence models produced significantly different results for the unsteady turbulent flow over an airfoil. Further investigation is necessary to determine the accuracy of either result. The transport equation used by the Spalart-Allmaras model, however, appears to be more robust and better suited for unsteady applications.

References

- [1] Edwards, J.W. and Thomas, J.L., "Computational Methods for Unsteady Transonic Flows," in *Unsteady Transonic Aerodynamics* (D. Nixon, ed.), vol. 120, ch. 5, p. 212, American Institute of Aeronautics and Astronautics, Inc., Washington, D.C., 1989.
- [2] Briley, W. R. and McDonald, H., "Solution of the Three-Dimensional Navier-Stokes Equations by an Implicit Technique," in *Lecture Notes in Physics*, vol. 35, pp. 105–110, Proc. Fourth Int. Conf. Num. Methods Fluid Dyn., Springer-Verlag, New York, 1972.
- [3] Beam, R.M. and Warming, R.F., "An Implicit Factored Scheme for the Compressible Navier-Stokes Equations," *AIAA Journal*, vol. 26, no. 4, pp. 393–402, 1978.
- [4] Jameson, A., "Solution of the Euler Equations by a Multigrid Method," *Applied Mathematics and Computation*, vol. 13, pp. 327–356, 1983.
- [5] Chakravarthy, S.R. and Osher, S., "Numerical Experiments with the Osher Upwind Scheme for the Euler Equations," *AIAA Journal*, vol. 21, no. 9, pp. 1241–1248, 1983.
- [6] Rai, M.M. and Chakravarthy, S.R., "An Implicit Form for the Osher Upwind Scheme," *AIAA Journal*, vol. 24, pp. 735–743, May 1986.
- [7] Pulliam, T.H., "Time Accuracy and the Use of Implicit Methods." AIAA Paper 93-3360-CP, July 1993.

- [8] A. Belov, L. Martinelli, and A. Jameson, "A New Implicit Algorithm with Multigrid for Unsteady Incompressible Flow Calculation." AIAA Paper 95-0049, January 1995.
- [9] Alonso, J. and Martinelli, L. and Jameson, A., "Multigrid Unsteady Navier-Stokes calculations with Aeroelastic Applications." AIAA Paper 95-0048, January 1995.
- [10] J.-S. Shuen, K.-H. Chen, and Y. Choi, "A Time-Accurate Algorithm for Chemical Non-Equilibrium Viscous Flows at All Speeds." AIAA Paper 92-3639, July 1992.
- [11] Arnone, A., Liou, M.-S. and Povinelli, L.A., "Integration of Navier-Stokes Equations Using Dual Time Stepping and a Multigrid Method," *AIAA Journal*, vol. 33, pp. 985-990, June 1995.
- [12] Venkateswaran, S. and Merkle, C.L., "Dual Time Stepping and Preconditioning for Unsteady Computations." AIAA Paper 95-0078, January 1995.
- [13] Rumsey, C.L., Sanetrik, M.D., Biedron, R.T., Melson, N.D., Parlette, E.B., "Efficiency and Accuracy of Time-Accurate Turbulent Navier-Stokes Computations." AIAA Paper 95-1835-CP, July 1995.
- [14] Pulliam, T.H. and Chaussee, D.S, "A Diagonal Form of an Implicit Approximate Factorization Algorithm," *Journal of Computational Physics*, vol. 39, pp. 347-363, 1981.
- [15] Pulliam, T. H., "Efficient Solution Methods for the Navier-Stokes Equations." Lecture Notes For The Von Karman Institute For Fluid Dynamics Lecture Series: *Numerical Techniques For Viscous Flow Computation In Turbomachinery Bladings*, Jan. 1986.
- [16] Jameson, A., Schmidt, W. and Turkel, E., "Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping." AIAA Paper 81-1259, June 1981.

- [17] Pulliam, T.H., "Artificial Dissipation Models for the Euler Equations," *AIAA Journal*, vol. 24, no. 12, pp. 1931-1940, 1986.
- [18] Baldwin, B. and Lomax, H., "Thin-Layer Approximation and Algebraic Model for Separated Turbulent Flows." AIAA Paper 78-257, January 1978.
- [19] Spalart, P. R. and Allmaras, S. R., "A One-Equation Turbulence Model for Aerodynamic Flows." AIAA Paper 92-0439, January 1992.
- [20] Pulliam, T.H., "Low Reynolds Number Numerical Solutions of Chaotic Flows." AIAA Paper 89-0123, January 1989.

NOTE TO USERS

Page not included in the original manuscript are unavailable from the author or university. The manuscript was microfilmed as received.

41

This reproduction is the best copy available.

UMI

Substitution of equation A.4 into equation A.3 results in the following:

$$\hat{Q}^{p+1} = \hat{Q}^p - \Delta\tau \left(I + \frac{2}{3}\Delta t \frac{\partial \hat{D}^p}{\partial \hat{Q}^p} \right) \Delta \hat{Q}^p - \Delta\tau G(\hat{Q}^p) \quad (\text{A.5})$$

Rearranging equation A.5 and simplifying gives

$$\begin{aligned} \left[I + \Delta\tau I + \frac{2}{3}\Delta t \Delta\tau \hat{D}'(\hat{Q}^p) \right] \Delta \hat{Q}^p &= -\Delta\tau G(\hat{Q}^p) \\ \left[\frac{3}{2\Delta t} I + \frac{3\Delta\tau}{2\Delta t} I + \Delta\tau \hat{D}'(\hat{Q}^p) \right] \Delta \hat{Q}^p &= -\frac{3\Delta\tau}{2\Delta t} G(\hat{Q}^p) \\ \left[S + \Delta\tau \hat{D}'(\hat{Q}^p) \right] \Delta \hat{Q}^p &= -\frac{3\Delta\tau}{2\Delta t} \left(\hat{Q}^p - \mathcal{N} + \frac{2}{3}\Delta t \hat{D}(\hat{Q}^p) \right) \\ \left[S + \Delta\tau \partial_\xi \hat{A}^p + \Delta\tau \partial_\eta \hat{B}^p \right] \Delta \hat{Q}^p &= -\Delta\tau \left[\frac{3\hat{Q}^p - 4\hat{Q}^n + \hat{Q}^{n-1}}{2\Delta t} + \hat{D}(\hat{Q}^p) \right] \end{aligned} \quad (\text{A.6})$$

where

$$S = \left\{ \frac{3}{2\Delta t} + \frac{3\Delta\tau}{2\Delta t} \right\} I \quad (\text{A.7})$$

Thus the dual time-stepping scheme, in diagonal form, may be written as follows:

$$T_\xi [S + \Delta\tau \partial_\xi \Lambda_\xi] \hat{N}_d [S + \Delta\tau \partial_\eta \Lambda_\eta] T_\eta^{-1} \Delta \hat{Q}^p = -\Delta\tau G(\hat{Q}^p) \quad (\text{A.8})$$

where

$$\hat{N}_d = T_\xi^{-1} S^{-1} T_\eta \quad (\text{A.9})$$

After iterating p times the solution at time level $n+1$ will be taken from the most recent solution $\hat{Q}^{p+1}$. Upon convergence of the inner iterations, equation 2.11 is satisfied.