Fourth-Order Implicit Runge-Kutta Time Marching Using A Newton-Krylov Algorithm

by

Sammy Isono

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

Abstract

Fourth-Order Implicit Runge-Kutta Time Marching Using A Newton-Krylov Algorithm

Sammy Isono
Master of Applied Science
Graduate Department of Aerospace Science and Engineering
University of Toronto
2003

Two time-marching schemes are investigated for accuracy and efficiency in solving the unsteady Navier-Stokes equations. The time-marching methods considered are the 2^{nd}-order backwards differencing formula and the 4^{th}-order explicit first stage, single diagonal coefficient, diagonally implicit Runge-Kutta method. The efficiency of approximate factorization and Newton-Krylov algorithms are investigated for solving the nonlinear problem arising at each iteration. Laminar two-dimensional flows around a cylinder and an airfoil are studied. The relative efficiency of the methods varies with the grid resolution. The backwards differencing method with approximate factorization dual time stepping is very efficient on the coarse grid, whereas the implicit Runge-Kutta scheme combined with the Newton-Krylov algorithm is the most efficient on the finer grids and when lower errors are required. The combination of the implicit Runge-Kutta method with the Newton-Krylov algorithm is shown to be very efficient for high-fidelity time-accurate simulations.

Acknowledgements

First and foremost, I would like to thank my supervisor, Professor D. W. Zingg, for his guidance and supervision of my research.

I would also like to thank the members of the Compuational Fluid Dynamics Group at UTIAS. I would especially like to thank John Gatsis, Jay Liu, Marian Nemec, Scott Northrup, Mohammad Tabesh, and Peterson Wong.

To my friends and family, thanks for your patience and understanding. To Vivien, thanks for the support and encouragement.

Sammy Isono

University of Toronto Institute for Aerospace Studies
June 28, 2003

Contents

List of Figures

List of Tables

List of Symbols

$\hat{E}, \hat{F}$	convective flux
$\hat{Q}$	flow variables
$\hat{S}$	viscous flux
$\hat{D}$	dissipation
ρ	density
a	speed of sound
e	energy
p	pressure
u, v	Cartesian velocities
M	Mach number
$\mathcal{P}r$	Prandtl number
Re	Reynolds number
U, V	contravariant velocities
V_n	normal velocity component
V_t	tangential velocity component
μ	laminar viscosity
C_l	coefficient of lift
C_d	coefficient of drag
ξ, η	curvilinear coordinates
J^{-1}	metric Jacobian
ϵ	finite-difference stepsize
ϵ_m	machine zero
A	flow Jacobian
Δt	time step
$\Delta\tau$	"pseudo" time step
GMRES	generalized minimum residual method

Chapter 1

Introduction

Accurate numerical solution of the unsteady Navier-Stokes equations is needed in many areas including large eddy and direct simulations of turbulent flows, aeroacoustics and simulations of unsteady phenomena such as vortex shedding and buffet. Until fairly recently, computational fluid dynamics (CFD) dealt mainly with steady state problems. This was due to constraints in computing costs. As computers continue to become more powerful, we can begin to solve these unsteady problems.

Standard computational aerodynamics procedures are inefficient for unsteady computations requiring high accuracy. One of the main reasons for this is that standard CFD codes use low order time marching. Increasing the accuracy in time could lead to improved efficiency.

This thesis will examine and compare two implicit time-marching methods in order to determine which is more suitable for application in computational aeroacoustics and large eddy simulation (LES). The methods are compared in terms of efficiency in CPU time to acheive a given level of accuracy.

The nonlinear problem arising at each time step will be solved using dual time-stepping with approximate factorization and a Newton-Krylov algorithm. These iterative methods used at each time step will be referred in this thesis as subiterations. The subiterations will be compared for efficiency as well.

1.1 Applications

Computational aeroacoustics is a relatively new area of research. Khorrami et al.[9] use a globally 2^{nd}-order accurate flow solver for the three-dimensional, time-dependant,

thin-layer Navier-Stokes equations to study the source of tones generated by the slat of an airfoil in high lift configuration. Singer et al.[15] make use of computational aeroacoustics to model the flow around a multi-element airfoil in high lift configurations. They solve the Ffowcs Williams and Hawkings equation [7] using data from highly resolved, time-dependent, Reynolds-averaged Navier-Stokes (RANS) calculations. For both papers, the numerical analysis agrees qualitatively with experimental results. Tam and Pastouchenko [17] use a seven-point stencil dispersion- relation-preserving scheme [16] to simulate the flow of a wall jet through a gap. They derive an equation to predict the frequency of the generated tone which gives good agreement with experimental measurements. These papers show that current computers are capable of modeling acoustic behavior for practical cases.

RANS gives a time-averaged model of the turbulent flowfield. This limits the accuracy of the simulations. The solutions are also dependant on the effectiveness of the turbulence model. Recent studies in computational aeroacoustics have made use of LES which provide a more accurate prediction for aerodynamic flows. DeBonis and Scott [4] simulate a high-Reynolds number turbulent jet using LES. They use a globally 4^{th}-order accurate scheme with a five-stage, 4^{th}-order Runge-Kutta time-stepping algorithm. The results accurately simulate the physics of the turbulent jet. LES is still not practical for solving complex flows due to the computational costs. The bottleneck for LES is usually the time-marching [4]. An efficient time-marching method can make LES practical for application to more complex problems.

1.2 Computational Methods

The Navier-Stokes equations are a set of equations derived from the three universal laws of conservation for mass, momentum, and energy. Combined with an equation of state (to close the system of equations), the Navier-Stokes equations provide a mathematical model capable of representing a wide range of fluid flows. For this study, the unsteady thin-layer Navier-Stokes equations will be used to model the flow. The thin-layer Navier-Stokes equations assume that the viscous terms containing derivatives in the directions parallel to the body surface can be neglected.

For this study we examine implicit time-marching methods. In order to permit flexibility in grid generation, implicit time-marching methods are often preferred as a result of their unconditional stability. A-stable linear multistep methods are restricted to 2^{nd}-order

accuracy [10]. Recently, Bijl et al. [1, 2] and Carpenter et al. [3] presented A-stable implicit Runge-Kutta methods of up to 5^{th}-order which are promising for solving unsteady flow problems. Bijl et al. recommend their 4^{th}-order scheme as the most robust and demonstrate its superiority over the 2^{nd}-order backwards difference method for an unsteady laminar flow over a cylinder. The benefits of the higher-order scheme increase as the required error tolerance decreases. The papers study the 1^{st}, 2^{nd}, and 3^{rd}-order backwards differencing schemes, and 3^{rd}, 4^{th}, and 5^{th}-order accurate ESDIRK schemes. The 1^{st}-order backwards differencing (which is simply implicit Euler time marching) cannot compete with the higher-order schemes at all in terms of efficiency for solving unsteady problems. This agrees with the results from De Rango and Zingg [6].

An implicit time-marching method produces a nonlinear problem at each time step or at each stage in the case of a multi-stage scheme such as ESDIRK. The efficiency of an implicit time-marching method depends on the algorithm used to solve the nonlinear problem. Bijl et al. [1] use an implicit-explicit multigrid scheme. For this thesis, an approximate-factorization scheme as well as a Newton-Krylov algorithm are used. The Krylov solver we use is the Generalized Minimal Residual (GMRES) algorithm [14].

1.3 Objectives

The objective of this thesis is to compare the relative efficiency of the dual time-stepping approximate factorization and Newton-Krylov subiteration algorithms and also to compare the backwards differencing and implicit Runge-Kutta schemes when used in conjunction with these solution algorithms.

Chapter 2

Governing Equations

The two-dimensional thin-layer Navier-Stokes equations are solved. In generalized curvilinear co-ordinates, the equations take the following form:

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

where $\hat{Q}$ is a vector containing the unknowns for mass, momentum, and energy:

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

$\hat{E}$ and $\hat{F}$ contain the fluxes in the ξ and η directions respectively:

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

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

where U and V are the contravariant velocities given by:

$$\begin{aligned} U &= \xi_t + \xi_x u + \xi_y v \\ V &= \eta_t + \eta_x u + \eta_y v \end{aligned}$$

and J represents the metric Jacobian of the transformation:

$$J^{-1} = (x_\xi y_\eta - x_\eta y_\xi) \tag{2.5}$$

The viscous terms are

$$\hat{S} = J^{-1} \begin{bmatrix} 0 \\ \eta_x m_1 + \eta_y m_2 \\ \eta_x m_2 + \eta_y m_3 \\ \eta_x(um_1 + vm_3 + m_4) + \eta_y(um_2 + vm_3 + m_5) \end{bmatrix} \tag{2.6}$$

with

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

Pressure is related to the conservative variables by the equation of state for a perfect gas.

$$p = (\gamma - 1)\left(e - \frac{1}{2}\rho(u^2 + v^2)\right) \tag{2.7}$$

2.1 Spatial Discretization

2^{nd}-order centered-differencing is used to perform the spatial discretization. In order to maintain stability, 2^{nd}- and 4^{th}-order artificial dissipation (see [8]) are used. Written as difference operators, the 2^{nd}- and 4^{th}- difference dissipation terms are

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

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

The dissipation coefficients are given by

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

where

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

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

The time-marching methods were implemented in two different flow solvers: Cyclone and Probe. Cyclone uses approximate factorization with implicit Euler time-marching. Probe uses a Newton-Krylov algorithm.

Cyclone uses 1^{st}-order explicit boundary conditions and Probe uses implicit boundary conditions. For the cylindrical grid, periodic boundary conditions were applied to the wakecut. For all test cases, no circulation correction was used. All the simulations are for viscous laminar flows. The dissipation coefficients are $\kappa_2 = 0$ and $\kappa_4 = 0.01$ for all of the test cases. For a detailed description of the treatment of the boundaries for this thesis see the works of Pueyo [13](Probe) and Nemec [11](Cyclone).

Chapter 3

Temporal Discretization

The time-marching methods investigated are the 2^{nd}-order backward difference formula (BDF) and the Explicit first stage, Single diagonal coefficient, Diagonally Implicit Runge-Kutta (ESDIRK) scheme of 4^{th} order.

Applying the 2^{nd}-order backwards differencing time-marching method to Eq. (2.1) gives the following equation:

$$\frac{3\hat{Q}_{n+1} - 4\hat{Q}_n + \hat{Q}_{n-1}}{2\Delta t} + \hat{D}(\hat{Q}_{n+1}) = G(\hat{Q}_{n+1}) = 0 \tag{3.1}$$

where $\hat{D}(\hat{Q}) = \partial_\xi \hat{E}(\hat{Q}) + \partial_\eta \hat{F}(\hat{Q}) - \mathcal{R}e^{-1}\partial_\eta \hat{S}(\hat{Q})$. This is a nonlinear relation for the solution at the next time step $\hat{Q}_{n+1}$.

The ESDIRK scheme was developed by Bijl, Carpenter, and Vatsa [1]. A general ESDIRK scheme of s stages is given by the following equations:

$$\frac{\hat{Q}^k - \hat{Q}_n}{a_{kk}\Delta t} + \frac{1}{a_{kk}} \sum_{j=1}^{k} a_{kj}(\hat{D}(\hat{Q}^k)) = G(\hat{Q}^k) = 0, \qquad k = 1, .., s \tag{3.2}$$

The method can also be expressed in predictor-corrector notation giving

$$\hat{Q}^k = \hat{Q}_n + \Delta t \sum_{j=1}^{k} a_{kj} \hat{D}(\hat{Q}^j), \qquad k = 1, .., s \tag{3.3}$$

The solution at the next time step is found by the following equation:

$$\hat{Q}_{n+1} = \hat{Q}_n + \Delta t \sum_{j=1}^{s} b_j \hat{D}(\hat{Q}^j) \tag{3.4}$$

The terms a_{ij} and b_j in the above equations are Butcher coefficients of the scheme. The following is a Butcher table of the coefficients for this scheme with $s = 6$. c_i indicates the point in the time interval $t + \Delta t$ which the solutions at each stage represent (i.e. the

0	0	0	0	0	0	0
c_2	a_{21}	a_{22}	0	0	0	0
c_3	a_{31}	a_{32}	a_{33}	0	0	0
c_4	a_{41}	a_{42}	a_{43}	a_{44}	0	0
c_5	a_{51}	a_{52}	a_{53}	a_{54}	a_{55}	0
c_6	a_{61}	a_{62}	a_{63}	a_{64}	a_{65}	a_{66}
	b_1	b_2	b_3	b_4	b_5	b_6

Table 3.1: Butcher table for a 6-stage ESDIRK scheme

solution at stage k evaluates $Q(t + c_k\Delta t))$. For ESDIRK schemes, the "stiffly accurate" assumption ($a_{ij} = b_j$) is enforced which automatically extends A-stability into L-stability. A numerical method is A-stable if it is stable for all ODEs that are inherently stable [10]. L-stability guarantees that eigenvalues approaching $-\infty$ are damped in one time step. Also, the diagonal terms (a_{kk}) are all chosen to be equal so that all the stages have the same stability.

ESDIRK schemes can be optimized to decrease the number of subiterations required to obtain a solution at a given level of accuracy. The number of stages and the diagonal coefficients are two important parameters. In the paper by Bijl et al.[1], it is stated that several combinations were tested to determine an optimal choice for the number of stages and diagonal coefficients. For the 4^{th}-order scheme, the optimal values were found to be $a_{kk} = \frac{1}{4}$ and s=6 stages. In the article, 3^{rd}, 4^{th}, and 5^{th}-order ESDIRK schemes were compared and it was concluded that the 4^{th}-order scheme is more robust and efficient than the others. For this thesis only the 4^{th}-order ESDIRK scheme was considered. A Butcher table for the 4^{th}-order ESDIRK scheme is given in Appendix B.

Equations (3.1) and (3.2) represent nonlinear problems which must be solved at every time step (or stage). In order to achieve 4^{th}-order accuracy in time, the nonlinear equation at each stage of ESDIRK must be solved to an appropriately small error. The solution at each time step is found using dual time stepping with approximate factorization or Newton-Krylov as described in the following sections.

For each stage of ESDIRK, the initial guess used at the begining of the subiterations is the solution for the previous stage. Other initial guesses were used, however, this method produced the best results.

3.1 Dual Time Step with Approximate Factorization

The dual time step scheme which we use is the strategy employed by Venkateswaran and Merkle [18]. A "pseudo" time derivative is added to Eq. (2.1). The resulting equation is

$$\frac{\partial \hat{Q}}{\partial \tau} + \frac{\partial \hat{Q}}{\partial t} + \hat{D}(\hat{Q}) = 0 \tag{3.5}$$

Applying implicit Euler time-marching to Eq. (3.5) gives

$$\frac{\hat{Q}_{p+1} - \hat{Q}_p}{\Delta \tau} + G(\hat{Q}_{p+1}) = 0 \tag{3.6}$$

where the function $G(\hat{Q}_{p+1})$ is from Eq. (3.1) for BDF and from Eq. (3.2) for ESDIRK. For example, if dual time stepping is applied to 2^{nd}-order backwards differencing (Eq. (3.1)), the equation to be solved at each time step becomes:

$$\frac{\hat{Q}_{p+1} - \hat{Q}_p}{\Delta \tau} + \left[\frac{3\hat{Q}_{p+1} - 4\hat{Q}_n + \hat{Q}_{n-1}}{2\Delta t} + \hat{D}(\hat{Q}_{p+1}) \right] = 0 \tag{3.7}$$

With a local linearization in pseudo-time, one obtains

$$[S + \Delta\tau \partial_\xi \hat{A}_p + \Delta\tau \partial_\eta \hat{B}_p] \Delta \hat{Q}_p = -\frac{\Delta \tau}{\alpha \Delta t} (\hat{Q}_p - \mathcal{N} + \alpha \Delta t \hat{D}(\hat{Q}_p)) \tag{3.8}$$

where

$$S = \left\{ \frac{3}{2\Delta t} + \frac{3\Delta\tau}{2\Delta t} \right\} I$$

and

$$\mathcal{N} = \frac{4}{3}\hat{Q}_n - \frac{1}{3}\hat{Q}_{n-1}$$

Solving Eq. (3.7) to steady state in the "pseudo" time variable τ results in the solution of Eq. (3.1) The dual time step approach is not necessary for BDF time-marching. However, when compared with a time-linearized approximately factored scheme, the dual time step approach is much more efficient since it reduces both linearization and factorization error and provides a simple way of avoiding boundary conditions that are 1^{st}-order in time (see De Rango and Zingg[6]).

Similarly, applying dual time-stepping to ESDIRK (i.e. combining Eqs. (3.2) and (3.6)) gives at stage k,

$$\frac{\hat{Q}^k_{p+1} - \hat{Q}^k_p}{\Delta \tau} + \left[\frac{\hat{Q}^k_{p+1} - \hat{Q}_n}{a_{kk}\Delta t} + \frac{1}{a_{kk}} \sum_{j=1}^{k-1} a_{kj} \hat{D}(\hat{Q}^j) + \hat{D}\hat{Q}^k_{p+1} \right] = 0 \tag{3.9}$$

Again, applying a local linearization in pseudo-time, one obtains

$$[S + \Delta\tau\partial_\xi \hat{A}_p^k + \Delta\tau\partial_\eta \hat{B}_p^k]\Delta\hat{Q}_p^k = -\frac{\Delta\tau}{\alpha\Delta t}(\hat{Q}_p^k - \mathcal{N} + \alpha\Delta t\hat{D}(\hat{Q}_p)) \tag{3.10}$$

where

$$S = \left\{\frac{1}{a_{kk}\Delta t} + \frac{\Delta\tau}{a_{kk}\Delta t}\right\} I$$

and

$$\mathcal{N} = \hat{Q}_n - \Delta t \sum_{j=1}^{k-1} a_{kj}\hat{D}(\hat{Q}_n^j)$$

A "steady state" solution is found at each stage.

Approximate factorization is applied to Equations (3.8) and (3.10) in order to accelerate the convergence to steady state in "pseudo time". Applying approximate factorization and diagonalization to the equations gives

$$\begin{aligned} T_\xi[S + \Delta\tau\partial_\xi\Lambda_\xi]\hat{N}_d[S + \Delta\tau\partial_\eta\Lambda_\eta]T_\eta^{-1}\Delta\hat{Q}_p \\ = -\frac{\Delta\tau}{\alpha\Delta t}(\hat{Q}_p - \mathcal{N} + \alpha\Delta t\hat{D}(\hat{Q}_p)) \end{aligned} \tag{3.11}$$

for BDF time-marching and

$$\begin{aligned} T_\xi[S + \Delta\tau\partial_\xi\Lambda_\xi]\hat{N}_d[S + \Delta\tau\partial_\eta\Lambda_\eta]T_\eta^{-1}\Delta\hat{Q}_p^k \\ = -\frac{\Delta\tau}{\alpha\Delta t}(\hat{Q}_p^k - \mathcal{N} + \alpha\Delta t\hat{D}(\hat{Q}_p)) \end{aligned} \tag{3.12}$$

for ESDIRK time-marching. Where

$$\hat{N}_d = T_\xi^{-1}S^{-1}T_\eta$$

The matrices Λ_ξ and Λ_η are diagonal matrices containing the eigenvalues of the flux Jacobians $\hat{A}$ and $\hat{B}$ respectively. The matrices T_ξ and T_η contain the corresponding eigenvectors. A detailed derivation of Equations. (3.11) and (3.12) are given in the appendices.

By simplifying using approximate factorization, the problem at each "pseudo" time-step requires the solution of two block pentadiagonal systems of equations. Diagonalization simplifies the problem further and requires the solution of two scalar pentadiagonal systems instead. Note: $\Delta\tau = 10^4$ is used for all cases using approximate factorization

3.2 Newton-Krylov method

Another method to solve Eqs. (3.1) and (3.2) is to apply Newton's method. This section gives an overview of the Newton-Krylov algorithm used in this thesis. For a more thorough desciption of the algorithm see Pueyo [13]. Expanding Eqs. (3.1) and (3.2) in a Taylor series gives

$$G(\hat{Q}^{p+1}) = G(\hat{Q}^p) + \left(\frac{\partial G(\hat{Q}^p)}{\partial \hat{Q}^p}\right)\Delta\hat{Q}^p + ... = 0 \tag{3.13}$$

Ignoring higher order terms gives the following linear system, which is solved at each iteration:

$$A^p\Delta\hat{Q}^p = -G(\hat{Q}^p) \tag{3.14}$$

where A^p is the Jacobian of $G(\hat{Q}^p)$ given by

$$A^p = \frac{\partial G(\hat{Q}^p)}{\partial \hat{Q}^p} = \frac{I}{\alpha\Delta t} + \frac{\partial \hat{D}(\hat{Q}^p)}{\partial \hat{Q}^p} \tag{3.15}$$

where $\alpha = \frac{2}{3}$ for BDF and $\alpha = a_{kk}$ for ESDIRK. If this linear system is solved iteratively to some finite tolerance, the resulting method is called an inexact Newton method.

3.2.1 Linear Problem

For this research, we use the Generalized Minimal Residual method (GMRES) to solve the linear system of equations for each Newton iteration. GMRES, which was developed by Saad and Schultz [14], is a Krylov-based iterative method for solving nonsymmetric linear systems of the form

$$\mathcal{A}x = b$$

Krylov methods are based on Krylov subspaces. A Krylov subspace is defined as:

$$K_m = \text{span}\left\{v_1, Av_1, A^2v_1, ..., A^{m-1}v_1\right\}$$

where the vector v_1 is defined as

$$v_1 = \frac{r_0}{||r_0||_2} = \frac{b - \mathcal{A}x_o}{||b - \mathcal{A}x_o||_2}$$

GMRES finds the iterate $x_m \in x_0 + K_m$ that minimizes the L_2 norm of the residual $r_m = b - \mathcal{A}x_m$. Once an initial guess x_o is selected, an orthogonal basis of K_m is formed

using a modified Gram-Schmidt orthogonalization known as Arnoldi's method.

1. For $j = 1, 2, ..., m$
2. $h_{i,j} = (\mathcal{A}v_j, v_i), i = 1, 2, ..., j$
3. $\hat{v}_{j+1} = \mathcal{A}v_j - \sum_{i=1}^{j} h_{i,j} v_j$
4. $h_{j+1,j} = ||\hat{v}_{j+1}||$
5. $v_{j+1} = \hat{v}_{j+1}/h_{j+1,j}$
6. end

It can be shown that

$$\mathcal{A}V_m = V_{m+1}\bar{H}_m$$

where V_m is an $N \times m$ matrix with the column vectors $v_1, ..., v_m$ and $\bar{H}_m$ is a Hessenberg matrix containing the $h_{i,j}$ coefficients found using Arnoldi's method. Any vector x in the space $x_0 + K_m$ can be written as

$$x = x_0 + V_m y \tag{3.16}$$

where y is a vector of dimension m. The residual can be rewritten as a function of y as follows:

$$\begin{aligned} ||r(y)||_2 &= ||b - \mathcal{A}x||_2 \\ &= ||b - \mathcal{A}(x_0 + V_m y)||_2 \\ &= ||r_0 - \mathcal{A}V_m y||_2 \\ &= ||\beta v_1 - V_{m+1}\bar{H}_m y||_2 \\ &= ||V_{m+1}(\beta e_1 - \bar{H}_m y)||_2 \end{aligned}$$

where $\beta = ||r_0||_2$, and e_1 is the first column of the $m \times m$ identity matrix. Since the columns of the matrix V_{m+1} are orthonormal,

$$||r(y)||_2 = ||\beta e_1 - \bar{H}_m y||_2$$

The problem has now been reduced to finding the vector y_m which minimizes the residual. This is relatively inexpensive since it is a least-squares problem of size $(m+1) \times m$ where $m \ll N$. Once the vector y_m, which minimizes the residual, has been found, x_m can be determined by using Eq. (3.16).

Restarted GMRES

GMRES is guaranteed to converge in at most $k = N$ steps. This is not practical, however, if N is large. A major drawback to GMRES is that the storage required increases

linearly and CPU time required increases quadratically with each iteration. One method to avoid this problem is to restart the iteration. After a chosen number of iterations $m \ll N$, all data is cleared and the procedure is restarted using x_m as the initial guess. This is referred to as the restarted version of GMRES (GMRES(m)). The choice of m is very important. If the number of iterates m is not enough, the solution may not converge, and if it is too large, then there is no benefit since it will require excessive work and storage. Saad and Shultz [14] have shown that if the coefficient matrix $\mathcal{A}$ is nearly positive definite, then a reasonable value for **m** can be chosen. The algorithm for restarted GMRES is given as follows.

1. Given x_0, compute $r_0 = b - \mathcal{A}x_0$ and $v_1 = r_0/||r_0||_2$
2. For $j = 1, ..., m$
   $w_j \mathcal{A} v_j$
   $h_{i,j} = (w_j, v_i)\ \ i = 1, 2, ..., j$
   $\hat{v}_{j+1} w_j - \sum_{i=1}^{j} h_{i,j} v_j$
   $h_{j+1,j} = ||\hat{v}_{j+1}||_2$
   $v_j + 1 = \hat{v}_{j+1}/h_{j+1,j}$
   Perform rotation on $\bar{H}_j$ and rhs vector
   If $||r_j||_2$ is small enough, exit
3. Form approximate solution
   Solve for y_m
   Form $x_m = x_0 + V_m y_m$
4. Restart
   Compute $r_m = b - \mathcal{A}x_m$
   If $||r_j||_2$ is small enough, stop
   Set $x_0 \leftarrow x_m$ and go to 1

Preconditioning

In order to be effective, GMRES requires the use of preconditioning. Preconditioning can be used to transform the coefficient matrix of the system which arises at each time step so that it is nearly positive definite. For this thesis, right preconditioning is used and takes the following form:

$$\mathcal{A}\mathcal{M}^{-1}\mathcal{M}x = b$$

where $\mathcal{M} \approx \mathcal{A}$. To form the preconditioner, we use an incomplete lower-upper factorization (ILU). In this method, an LU decomposition is approximated

$$\mathcal{A}v = \mathcal{M} - \epsilon = \mathcal{L}\mathcal{U} - \epsilon$$

where $\mathcal{L}$ and $\mathcal{U}$ are lower and upper triangular matrices. $\mathcal{L}$ and $\mathcal{U}$ are found using Gaussian elimination to the matrix $\mathcal{A}$ (or some approximation of $\mathcal{A}$). The factors can be less or more accurate depending on the number of nonzero terms which are retained. When only elements which have corresponding entries in $\mathcal{A}$ are kept, this is referred to as ILU(0). For this thesis, ILU(1) is used.

Matrix-free GMRES

GMRES does not explicitly require the formation of the matrix $\mathcal{A}v$. Only matrix-vector products are required. We take advantage of this fact by approximating the matrix-vector product as follows:

$$\mathcal{A}v \approx \frac{G(\hat{Q} + \epsilon v) - G(\hat{Q})}{\epsilon}$$

where ϵ is a small scalar used to perturb the state quantities $\hat{Q}$ in the direction of v. The performance of this matrix free approach is very sensitive to the choice for the value of ϵ. A large value will result in an inaccurate approximation, while a small value can lead to round-off error. The method we use is the strategy proposed by Nielson et al.[12] which involves choosing ϵ as follows

$$\epsilon||v||_2 = \sqrt{\epsilon_m}$$

where ϵ_m is the value of "machine zero".

Chapter 4

Results and Discussion

The temporal efficiency of the time-marching schemes is compared for unsteady laminar flow around a cylinder and an airfoil. The results are then used to compare their accuracy. A comparison is made on the effectiveness of approximate factorization and Newton-Krylov for solving the subiterations. For both of the cases presented, the exact solution to the problem is not known, so a very small time-step was used to obtain a reference solution. The reference solution is not free of spatial error, but temporal error is negligible.

The initial conditions for the test cases were obtained through two stages. First, the flow was simulated from freestream conditions using a reasonably small time step for several shedding cycles until a periodic steady state was reached. The flow was then advanced using a much smaller time step, with the 4^{th}-order time-marching method, until a new steady state was reached with the smaller time step. This solution was stored in a restart file as initial conditions for simulations at larger time steps.

The reference solution was obtained using the smaller time step size used to obtain the initial conditions. For all cases, the ESDIRK time-marching method was used to obtain the reference solution with a tight tolerance for the subiterations. Approximate factorization was used for the cylinder case and the Newton-Krylov algorithm was used for all of the airfoil cases to solve the subiterations for the reference solutions. The flow was advanced for approximately five periods of the cycle for the airfoil case and three periods for the cylinder case.

All error values discussed in this section were found by integrating the difference from the reference solution for C_l and C_d. The time steps used were chosen to be multiples of the reference Δt.

4.1 Test Cases

4.1.1 Cylinder

The grid used for the cylinder case was a 97×65 O-mesh with an off-wall spacing of 0.001. The boundary is at a distance of 20 diameters from the cylinder. The grid is shown in Fig. 4.1. The free stream Mach number is 0.3 with a Reynolds number of 1200. This case results in the periodic shedding of vortices. The vortices are shed from alternating sides of the cylinder. Fig. 4.2 shows a snapshot of the pressure contours in the flowfield. Bijl et al. [1] used this test case, with a slightly different grid. The reference solution was found using a time step of 0.01. The non-linear residual from the subiterations was reduced sufficiently so that the only error present is due to the spatial discretization. The initial values were obtained by simulating the flow for several cycles with a time step of 0.1 then simulting for another 20 cycles at a time step of 0.01.

The Strouhal number is often used to characterize unsteady flow. The Strouhal number is given by the following equation,

$$S = \frac{fD^2}{\nu \mathcal{R}e} \tag{4.1}$$

where f represents the frequency of the shed vortices, D represents a characteristic length, ν represents the kinematic viscosity, and $\mathcal{R}e$ is the Reynolds number. In non-dimensional form, the equation becomes:

$$S = \frac{n}{M} \tag{4.2}$$

where n represents the frequency of the shed vortices calculated using nondimensional time, and M is the freestream Mach number. The Strouhal number for this case was found to be 0.2566. This value is larger than the value of 0.2489 reported in [1]. Using a finer grid would give more accurate results; however, since we are concerned with temporal accuracy, this grid is adequate for studying the time-marching schemes. Figure 4.3 shows the lift and drag history for one shedding cycle. The curves shown are the reference solution, and solutions obtained using the BDF time-marching method for various time step sizes. The BDF solutions are shown since the errors are larger and it is easier to see the effect of varying Δt.

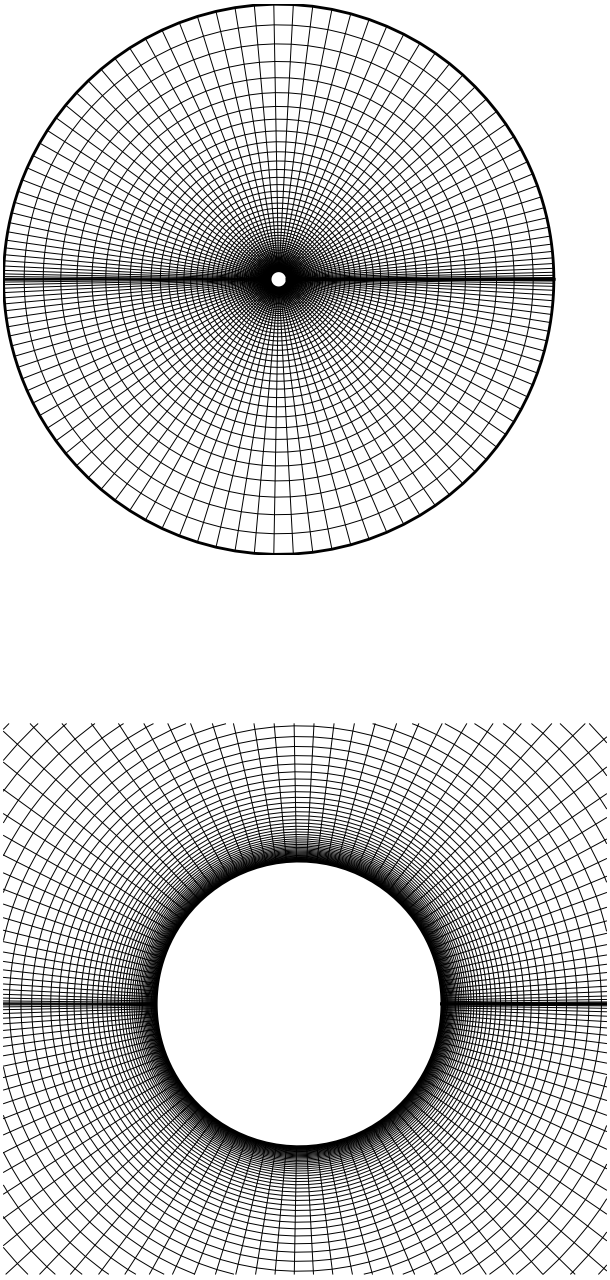

Figure 4.1: 97×65 O-mesh used for laminar flow over a cylinder

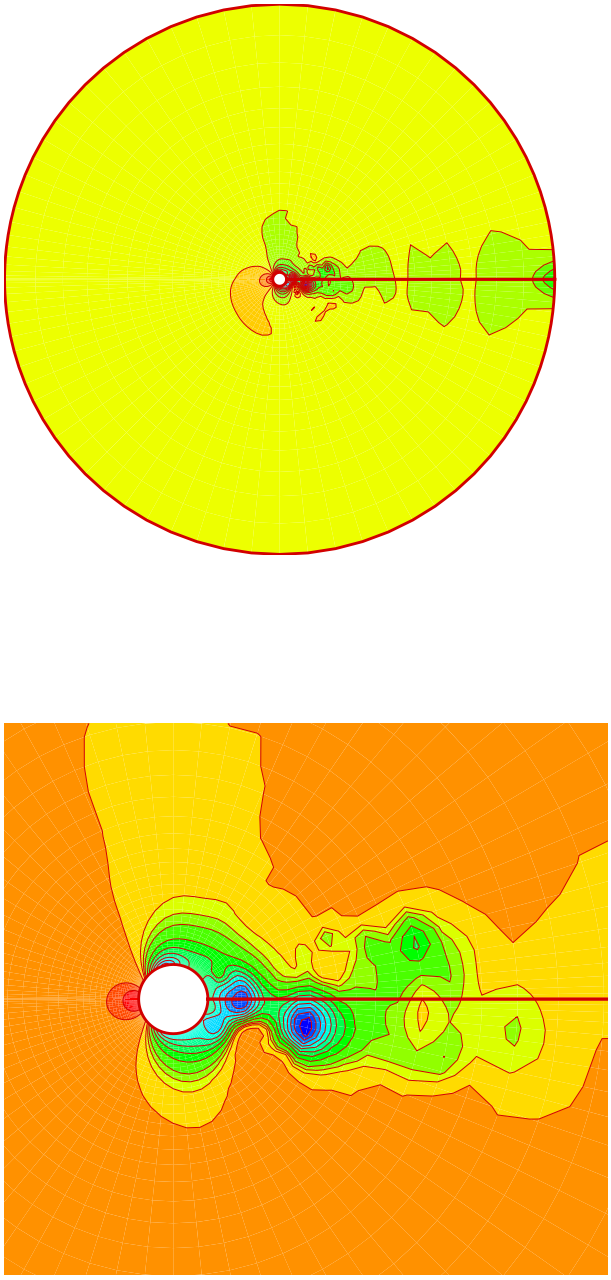

Figure 4.2: Pressure contours for flow over a cylinder ($M = 0.3$, $Re = 1200$)

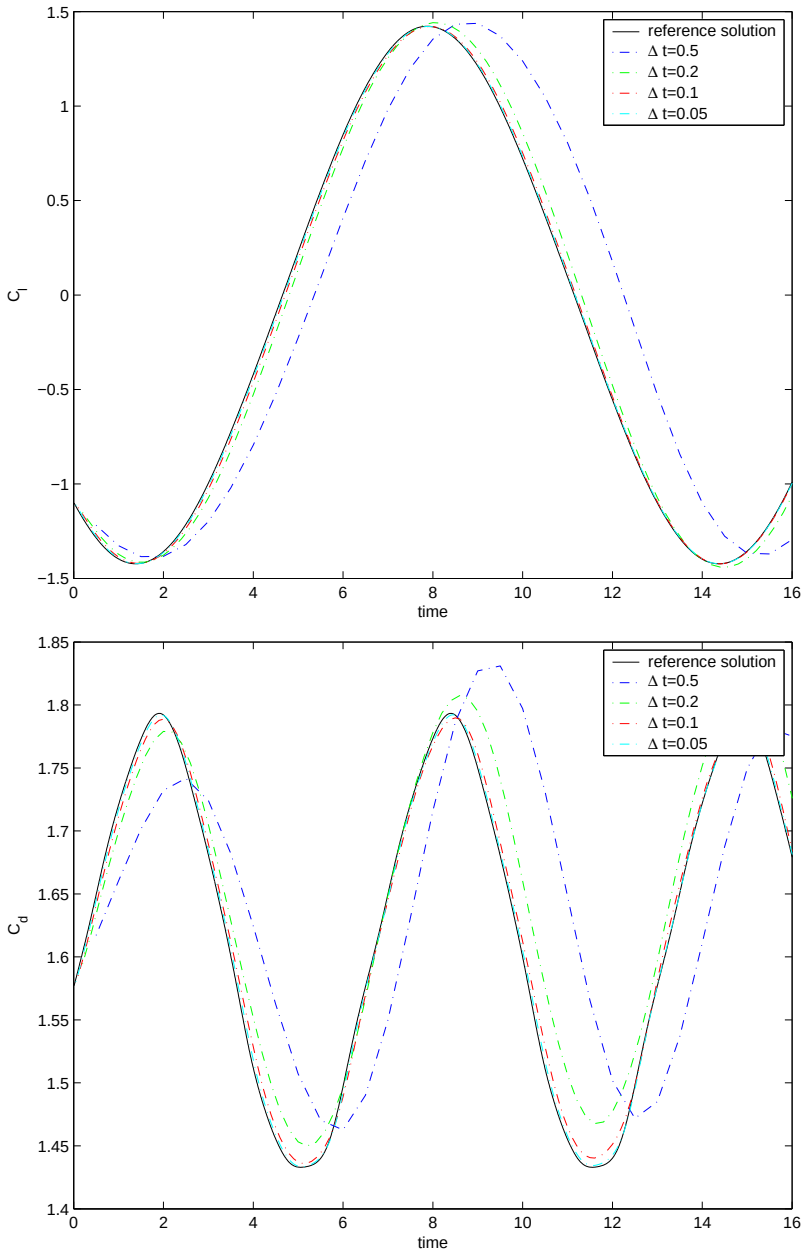

Figure 4.3: C_l and C_d vs time for flow over a cylinder

4.1.2 Airfoil

The second test case involves laminar flow over the NACA 0012 airfoil. The free stream Mach number is 0.2 with a Reynolds number of 800, and the angle of attack is 20°. Three grids were used for this case. This flow also results in a periodic shedding of vortices. A snapshot of the pressure contours in the flowfield can be seen in Figure 4.4. The solution shown in the figure was obtained using the finest of the three grids.

The three grids used for this case were generated using a hyperbolic grid generator (HYGRID). The first grid is a 169×49 C-mesh with 50 grid points on the upper and lower surfaces of the airfoil and 35 points in the wake. The off-wall spacing is 0.01 at the surface of the airfoil and the distance to the outer boundary is 12 chord lengths. The minimum spacing for the clustering at the leading edge is 0.005 and 0.001 for the trailing edge. This grid can be seen in Figure 4.5. The second grid is a 249×65 C-mesh with 75 points on the upper and lower surfaces and 50 points in the wake. The off-wall spacing is 10^{-4} at the surface of the airfoil and the distance to the outer boundary is 12 chord lengths. The minimum spacings for the clustering at the leading and trailing edges are the same as for the coarser grid. The finest grid used is a 499×130 C-mesh with 150 points on the upper and lower surfaces and 100 points in the wake. The minimum spacing for the clustering on the leading edge is 0.0025 and 0.0005 for the trailing edge. The distance to the outer boundary is 12 chord lengths with an off-wall spacing of 0.00005 at the surface of the airfoil.

Figures 4.6, 4.7, and 4.8 show plots of the lift and drag history for one period of the vortex shedding cycle for the coarse, medium and fine grids respectively. The time steps used to obtain the reference solutions were 0.01 for all of the airfoil grids. Solutions for several Δt obtained using the BDF time-marching method are also plotted. From the graphs it can be seen that a time step of about $\Delta t = 0.1$ gives reasonable results for the coarse and medium grids, and $\Delta t = 0.05$ for the fine grid.

Figure 4.9 shows a plot of the lift history over one cycle. The plots given are the reference solutions obtained on the three grids. It can be seen that the coarse grid does not provide the spatial resolution necessary to obtain accurate results. The medium and fine grids produce reasonably close results. The Strouhal numbers are 0.5105, 0.5214, and 0.5417 for the coarse, medium and fine grid respectively.

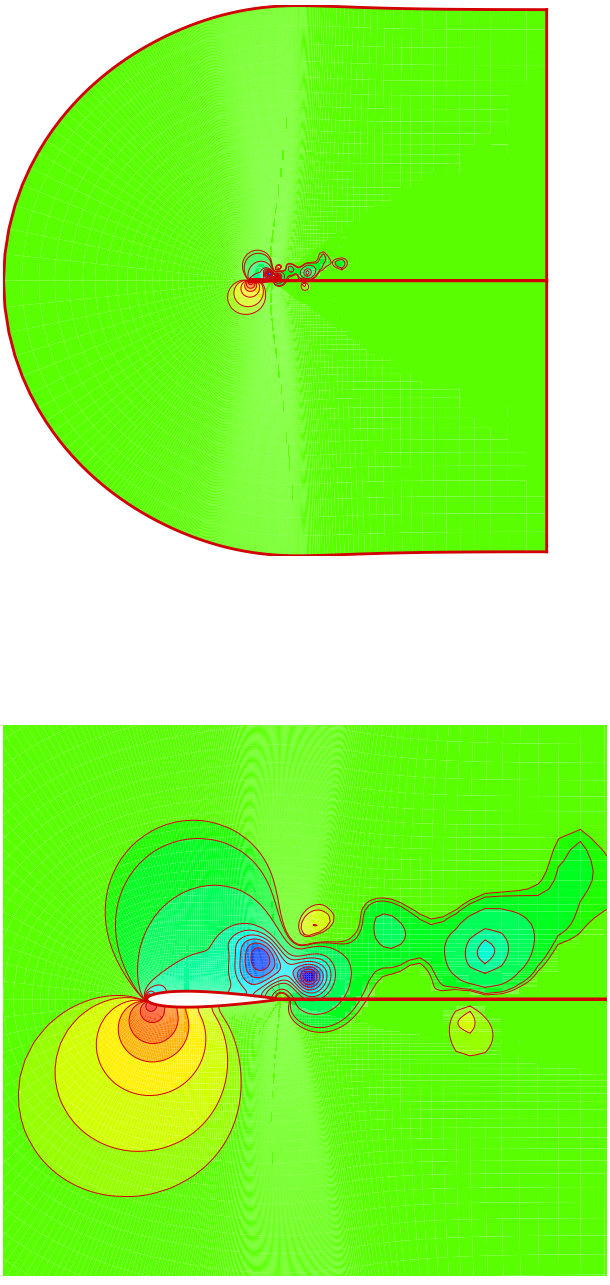

Figure 4.4: Pressure contours for flow over the NACA 0012 airfoil ($M = 0.2$, $Re = 800$, $\alpha = 20°$)

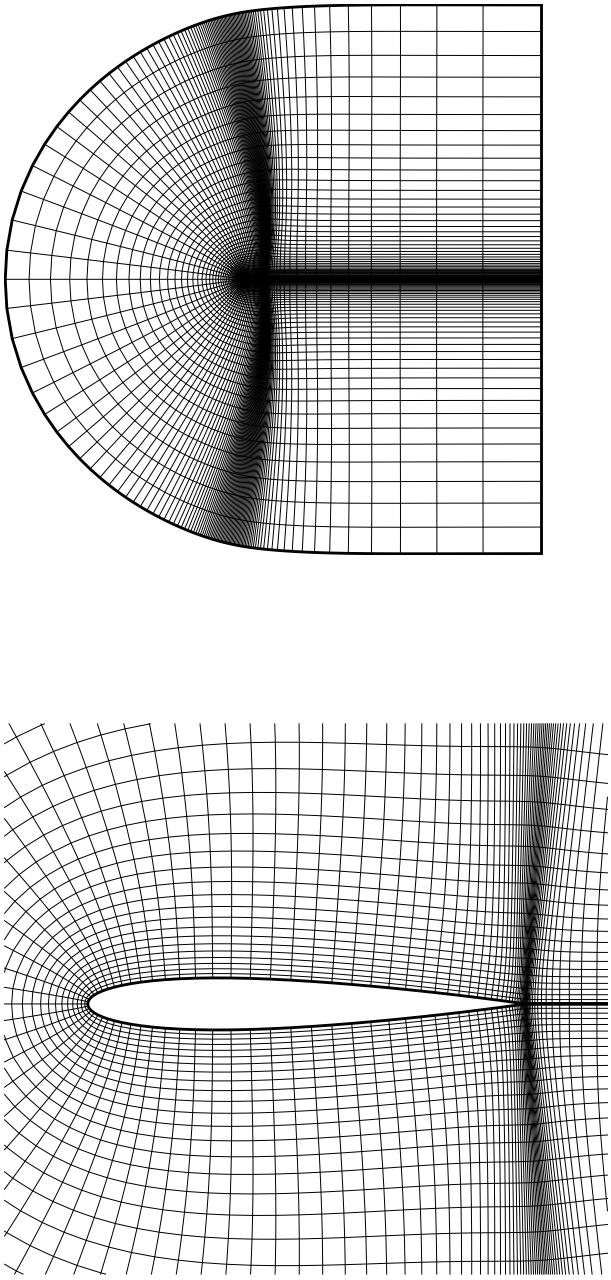

Figure 4.5: 169×49 C-mesh used for laminar flow over NACA 0012 airfoil

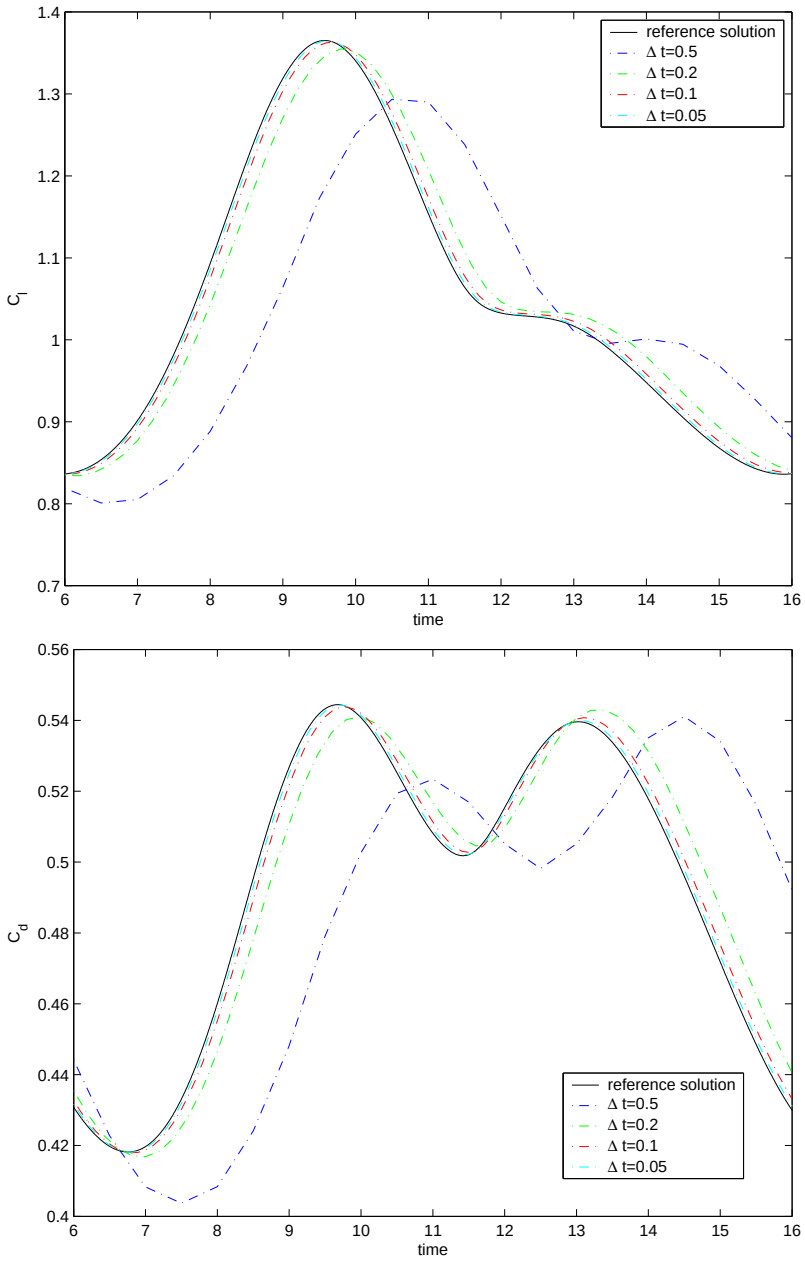

Figure 4.6: C_l, and C_d vs time found using coarse grid

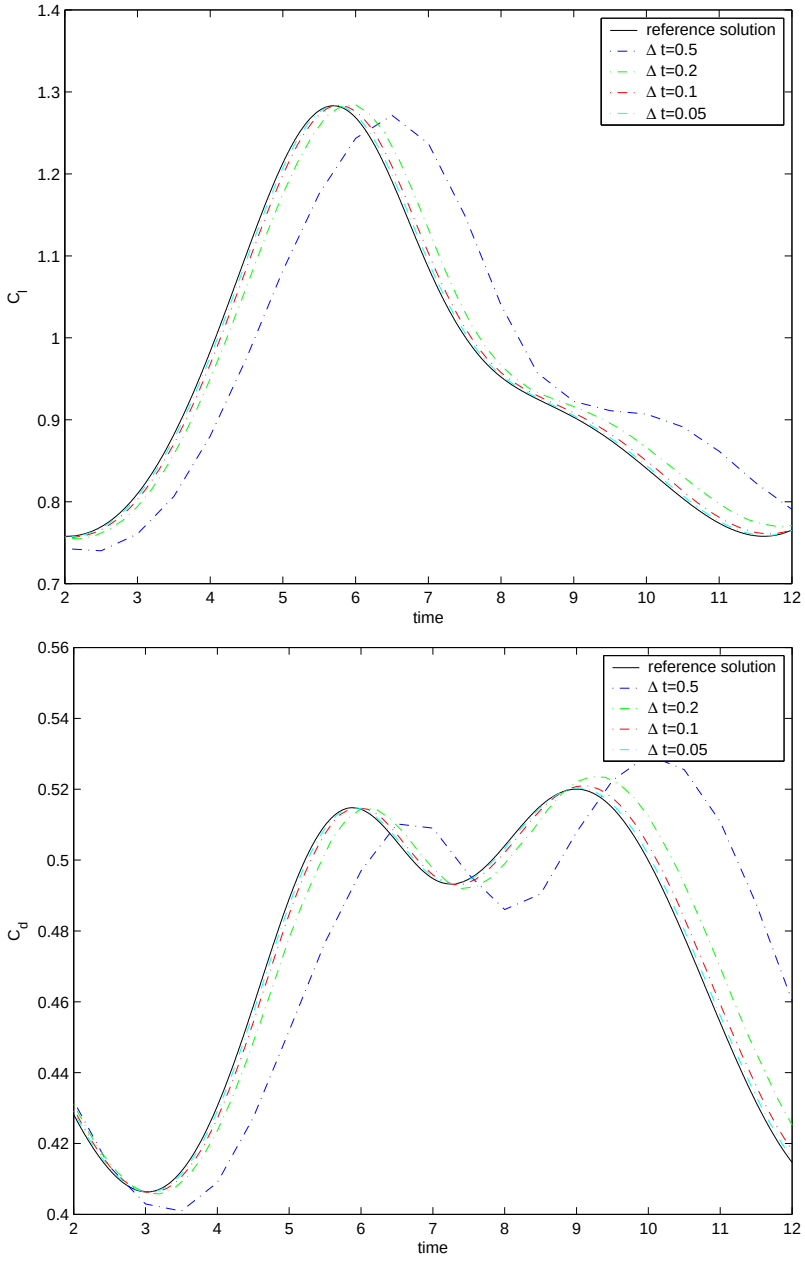

Figure 4.7: C_l, and C_d vs time found using medium grid

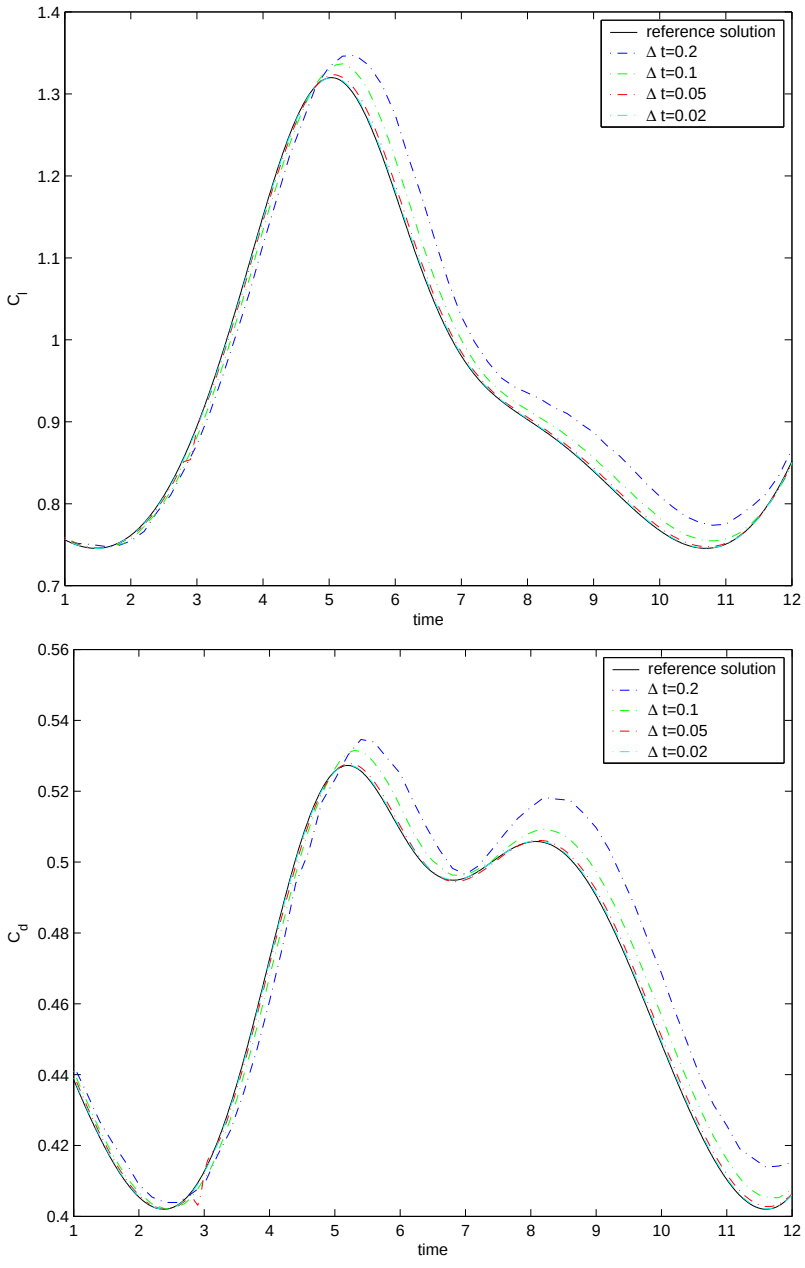

Figure 4.8: C_l, and C_d vs time found using fine grid

Figure 4.9: Comparison of reference solutions from three grids

4.2 Efficiency

All efficiency results are for the total cpu time required for the simulations. All of the results were found using an Alpha EV68 1000 MHz processor. The efficiency of the methods is examined in three steps. First, the most efficient parameters for the subiterations are found. The subiteration methods, approximate factorization and Newton-Krylov, are then compared to determine which method is more efficient at solving the subiterations. Finally, the two time-marching methods are compared.

4.2.1 Subiteration Efficiency

In order to obtain accurate results in time, the residual from the subiterations must be reduced such that the error due to linearization and boundary conditions is far lower than the error due to temporal discretization. However, reducing the error lower than necessary does not improve the accuracy of the results and simply slows down the calculation of the solution. Therefore an important aspect of subiterations is to determine when they should be terminated. The subiterations can be terminated once a predetermined tolerance is reached, or a fixed number of subiterations can be chosen. The Newton-Krylov

method also requires the selection of the number (or tolerance) of the GMRES iterations.

BDF-AF For the BDF time-marching scheme, De Rango and Zingg [5] have shown that using a fixed number of subiterations is very effective. Since each iteration converges to roughly the same residual, using a fixed tolerance yields similar results in terms of efficiency. This was found to be true for all cases. Figures 4.10 and 4.11 show the error vs the cpu time required for the cylinder and coarse-grid airfoil cases respectively.

Analysis of the runs on the finer grids gives the same results. Two subiterations generally gives the most efficient results. Solving to a fixed tolerance did not improve the efficiency. From this point, whenever the BDF time-marching method is used with approximate factorization subiterations, the results will refer to the solutions obtained using 2 subiterations.

ESDIRK-AF With the ESDIRK time-marching scheme, the residual from the subiterations varied by a larger amount and a fixed tolerance has to be used instead. For example, with $\Delta t = 0.2$ and a tolerance of 10^{-7}, the number of subiterations ranges from 5 to 9. This variation is more noticeable with the finer grids than with the coarse grid. It was found that a subiteration tolerance of 10^{-8} gives the most efficient results. Figures 4.12 and 4.13 show the error versus the cpu time required for the cylinder and coarse grids respectively. For this point on, a subiteration tolerance of 10^{-8} is used for the ESDIRK-AF method.

Newton-Krylov Algorithm

The efficiency of the Newton iterations depends on the convergence tolerance of the linear residual. Solving the residual from the linear problem beyond the level required for the nonlinear iteration increases the computational effort required, while providing little or no benefit to the convergence of the nonlinear problem. On the other hand, the convergence rate of the Newton iterations is limited by the solution to the linear problem.

For this study, the Newton-Krylov problems encountered at each time step have initial values which closely approximate the final solution. Hence the inexact-Newton approach can be used without a start-up phase.

It was found that, for the coarse grid, inner tolerances of 0.5 and 0.01 give the most efficient results for BDF and ESDIRK respectively. However, when applied to the medium grid, both methods seemed to work better with a tolerance of 0.1. Since the medium grid

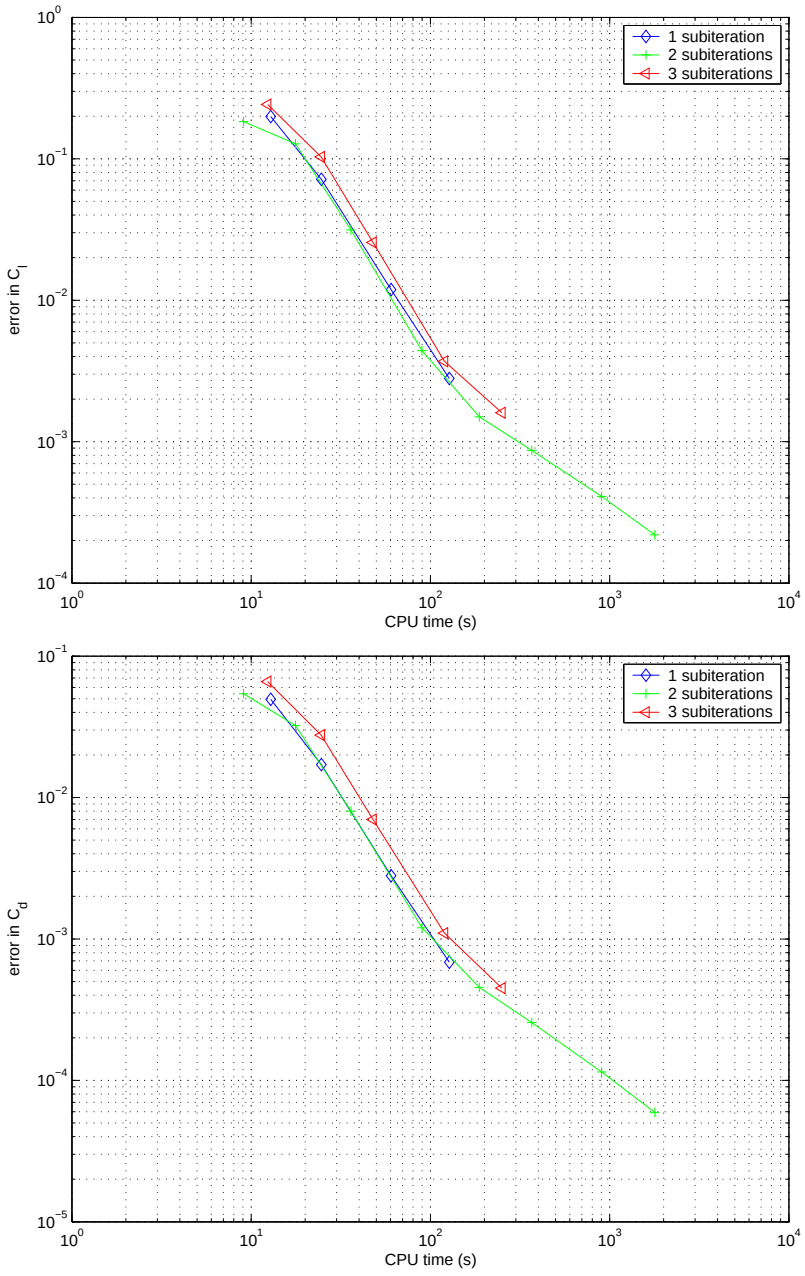

Figure 4.10: Comparison of approximate factorization subiterations for the BDF time-marching method (cylinder)

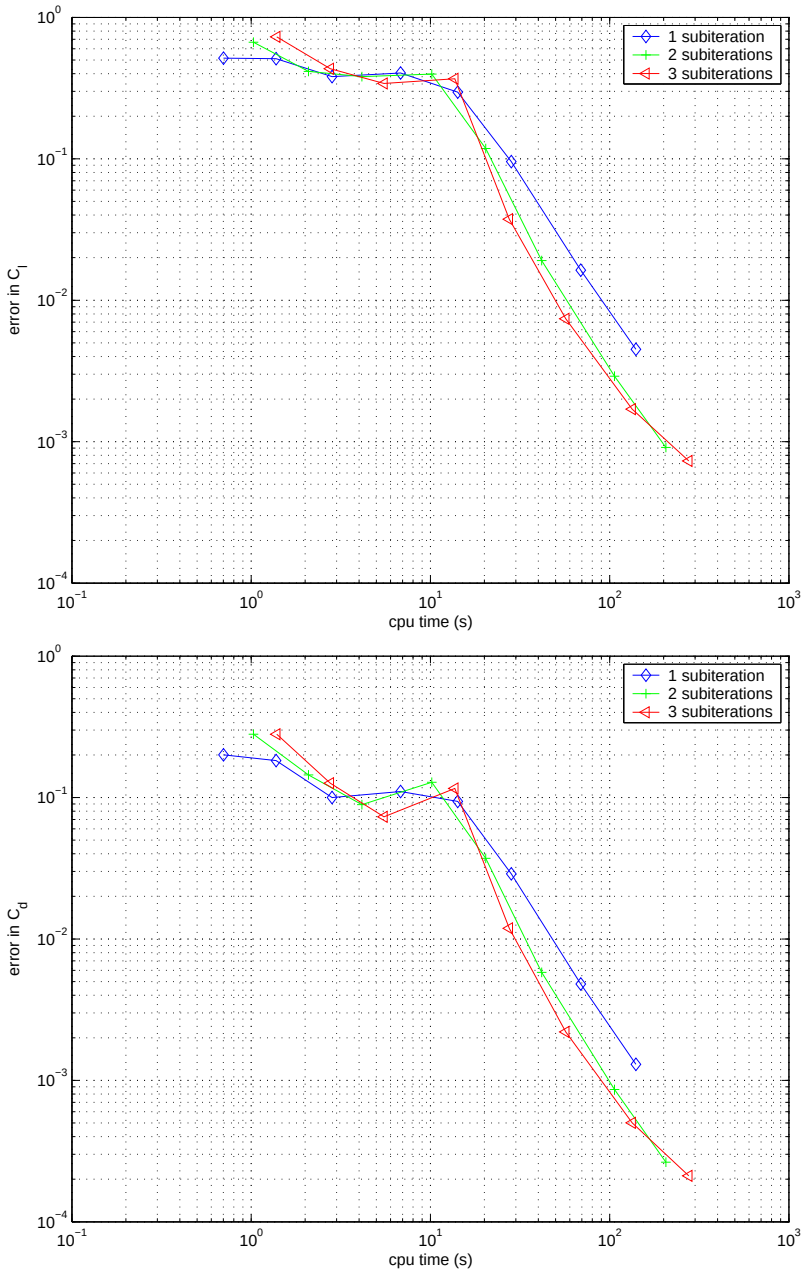

Figure 4.11: Comparison of approximate factorization subiterations for the BDF time-marching (coarse grid)

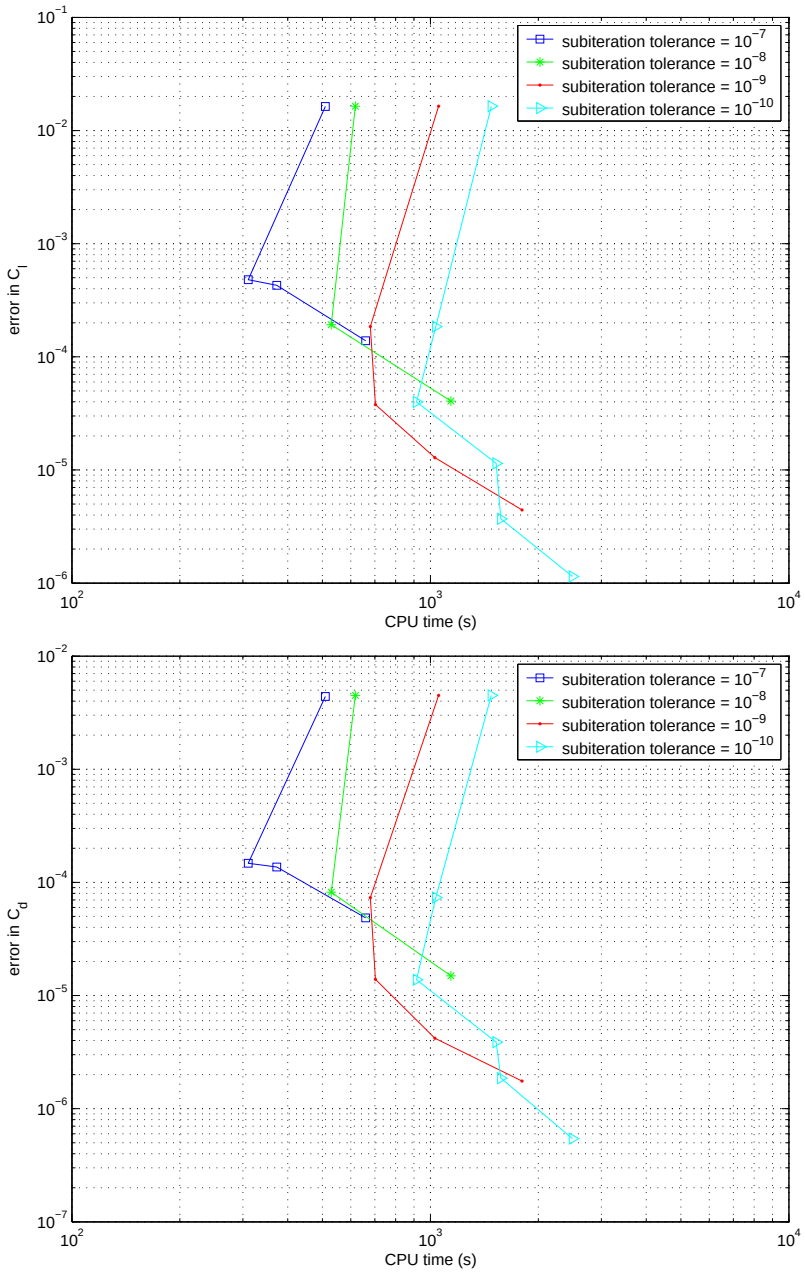

Figure 4.12: Comparison of approximate factorization subiterations for the ESDIRK time-marching method (cylinder grid)

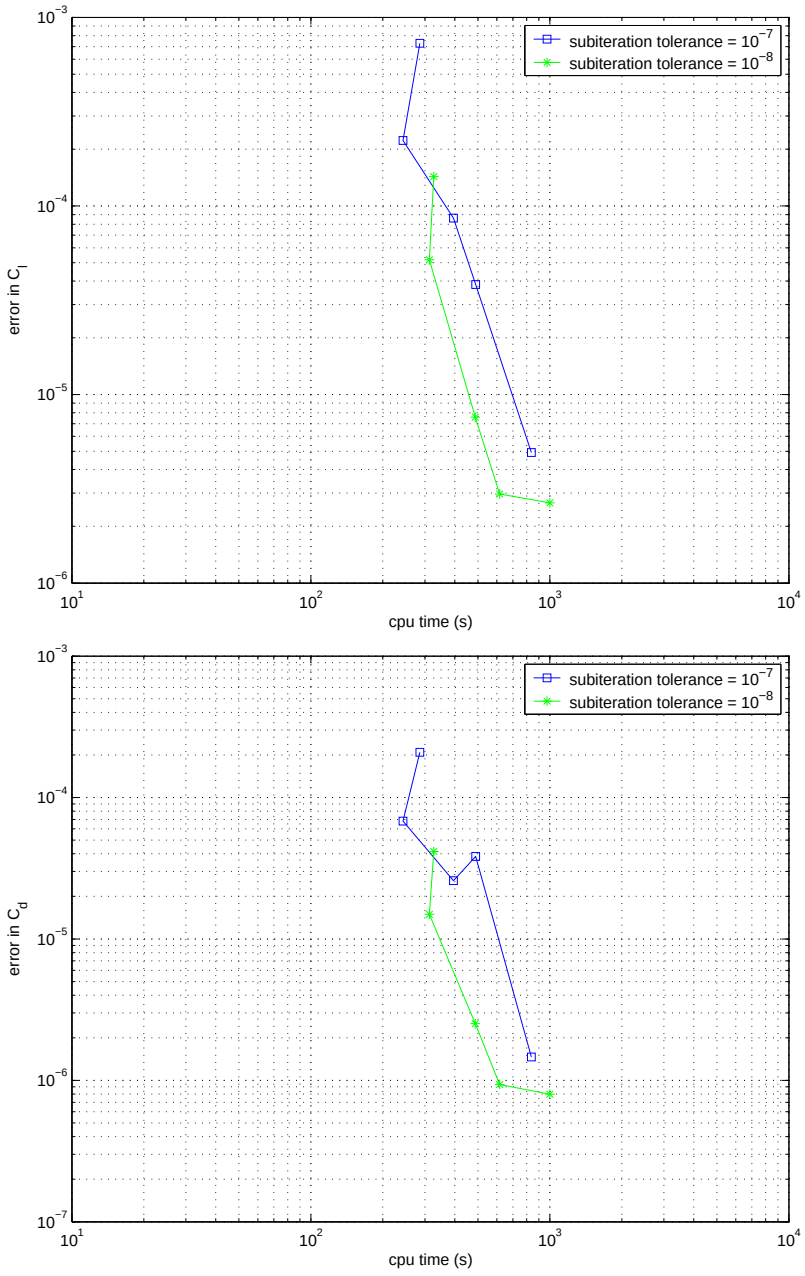

Figure 4.13: Comparison of approximate factorization subiterations for the ESDIRK time-marching method (coarse grid)

is more representative of practical test cases, all the results presented in this thesis are for a GMRES tolerance of 0.1.

Unlike the approximate factorization subiterations, the Newton subiterations give consistent errors when a fixed number of subiterations is used. Therefore, the results which are shown are for a fixed number of subiterations only. Results from the coarse grid for BDF and ESDIRK can be see in Figures 4.14 and 4.16 respectively. Results from the medium grid using BDF and ESDIRK are shown in Figures 4.15 and 4.17.

The amount which the Newton iterations reduce the nonlinear residual varies depending on the time step size. When a large time step is used the Newton iterations do not reduce the nonlinear residual as much, however, since at larger time steps, the error in the solution due to the accuracy of the scheme is also larger, there is no need to increase the number of subiterations. From Figures 4.14 and 4.15, it is clear that 2 Newton subiterations is the most efficient for the BDF time-marching scheme. One Newton subiteration is not used because this leads to instability. For the remainder of this thesis, whenever the BDF-NK time-marching scheme is used, the number of subiterations is 2. For the ESDIRK-NK time-marching method, four subiterations is generally the most efficient, so the four subiteration method is used for the remainder of this thesis.

4.2.2 Overall Efficiency

Approximate Factorization vs Newton-Krylov

BDF Figures 4.18, 4.19, and 4.20 show the efficiency of the BDF scheme in terms of cpu time for the coarse, medium, and fine grids respectively. For all grids, the lift and drag results give similar trends. On the coarse grid, at an error tolerance of about 10^{-3}, approximate factorization is about 3 times faster than Newton-Krylov. However, on the medium and fine grids, approximate factorization cannot be used to reach this same tolerance in a reasonable amount of time. Since the medium and fine grids are more representative of realistic cases, the Newton-Krylov subiterations are more appropriate to be used with BDF.

ESDIRK Figures 4.21 and 4.22 show the efficiency of the ESDIRK scheme in terms of cpu time for the coarse and medium grids respectively. For all grids, the lift and drag results give similar trends. On the coarse grid, the subiterations are equally efficient. However,

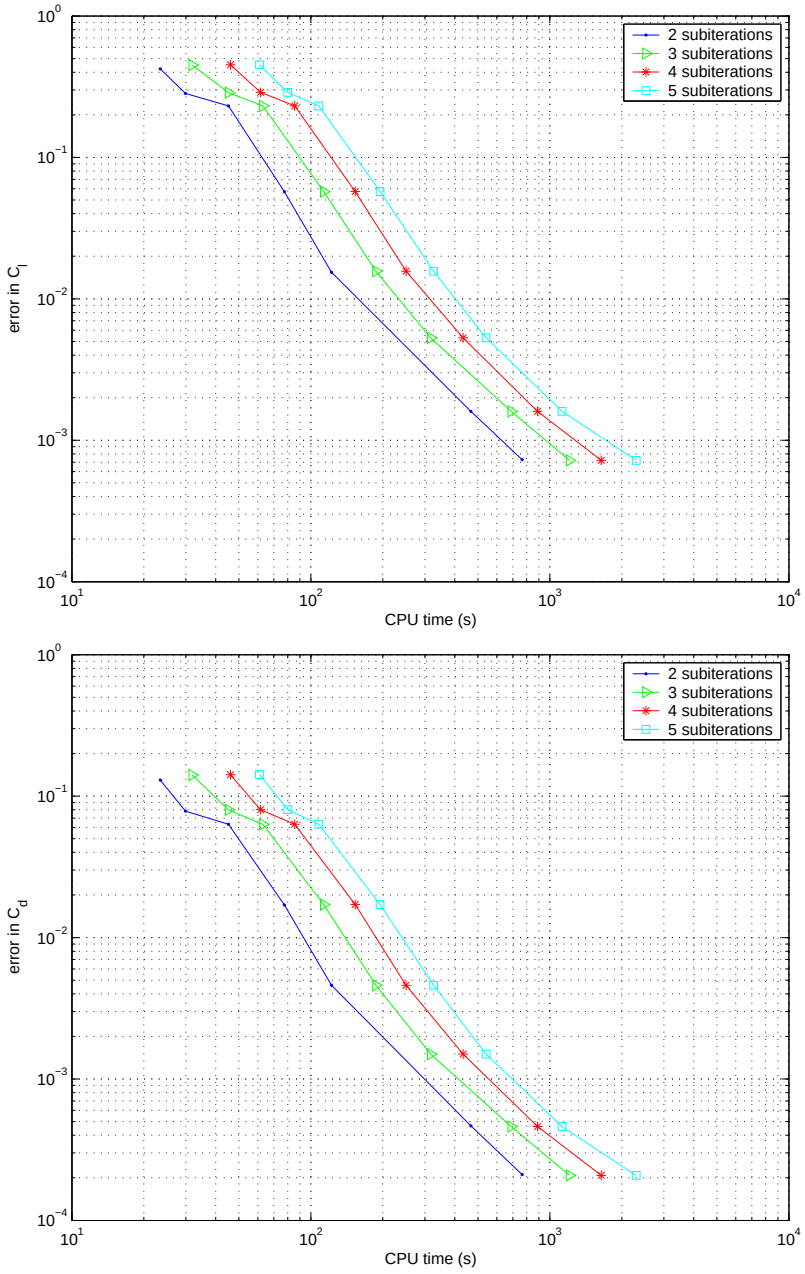

Figure 4.14: Comparison of subiteration parameters for BDF time-marching (coarse grid)

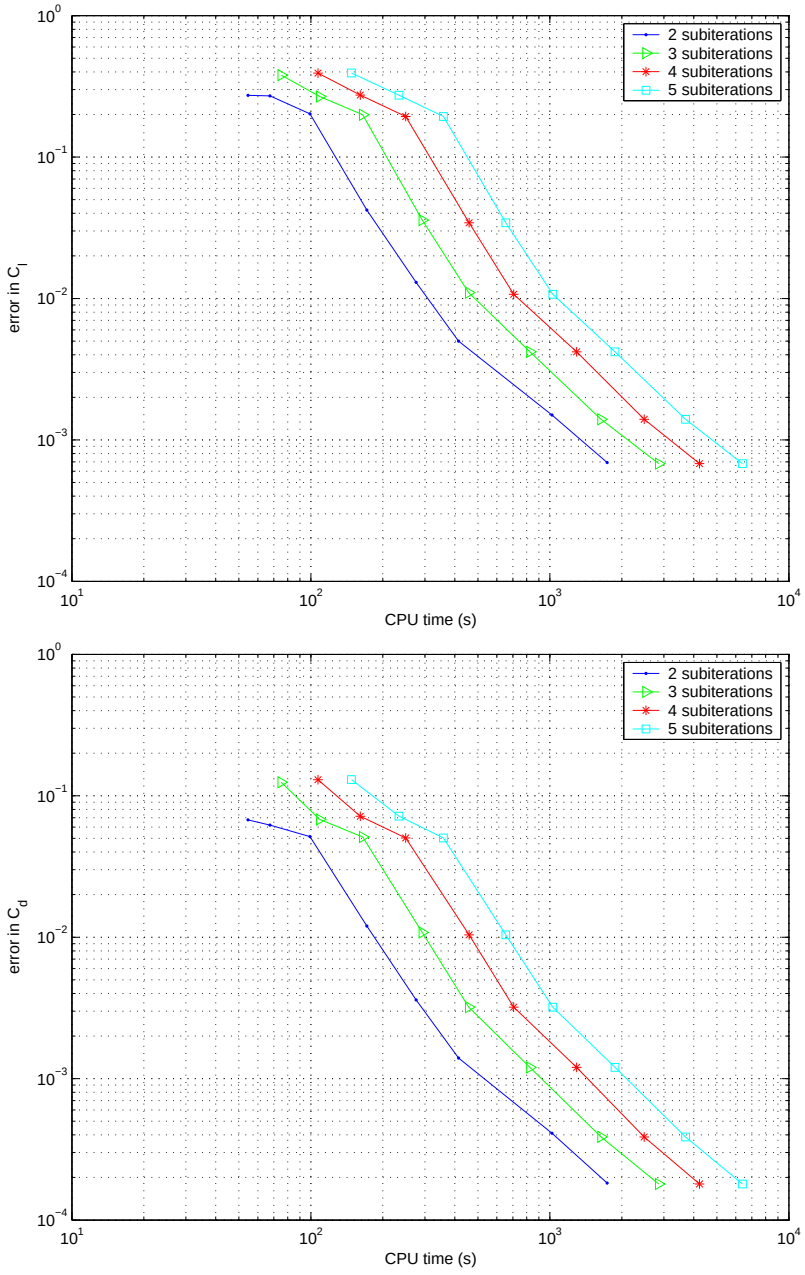

Figure 4.15: Comparison of subiteration parameters for BDF time-marching (medium grid)

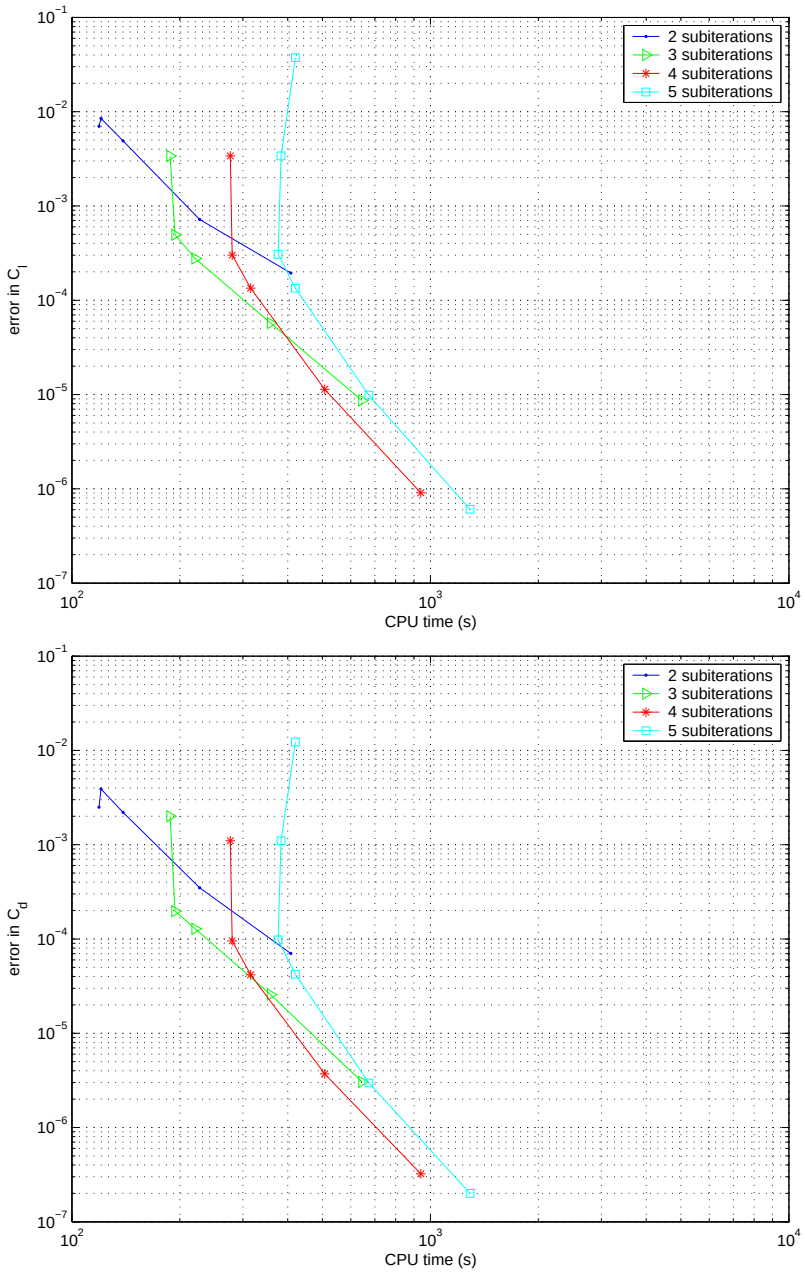

Figure 4.16: Comparison of subiteration parameters for ESDIRK time-marching method (coarse grid)

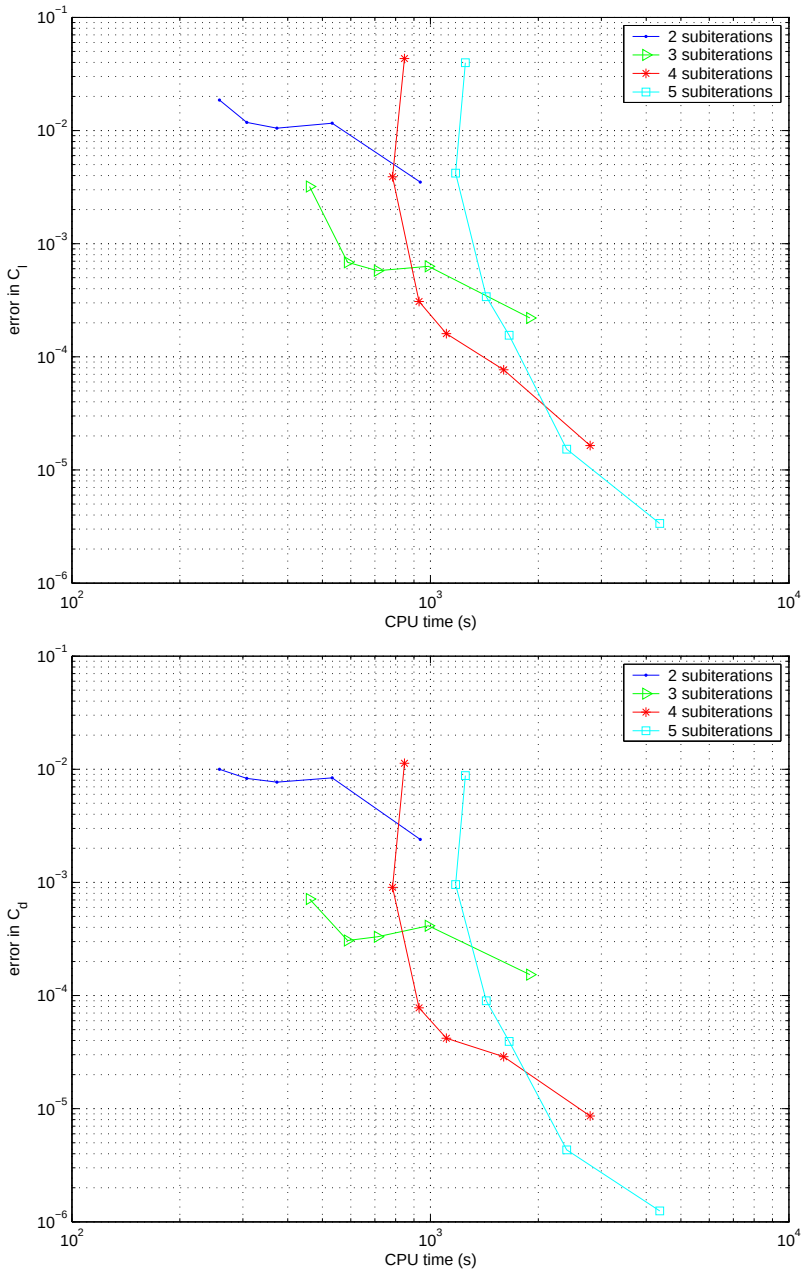

Figure 4.17: Comparison of subiteration parameters for ESDIRK time-marching method (medium grid)

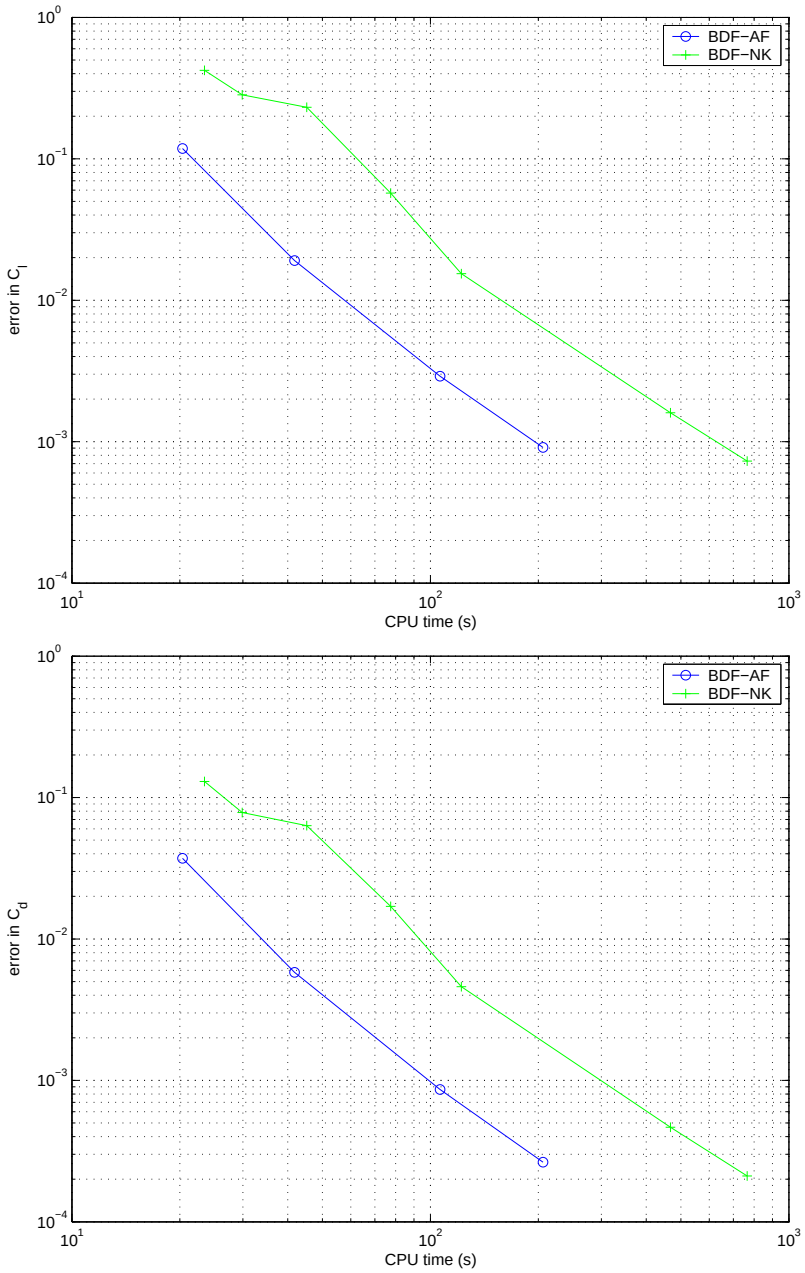

Figure 4.18: Comparison of subiteration algorithms for the BDF time-marching method (coarse grid)

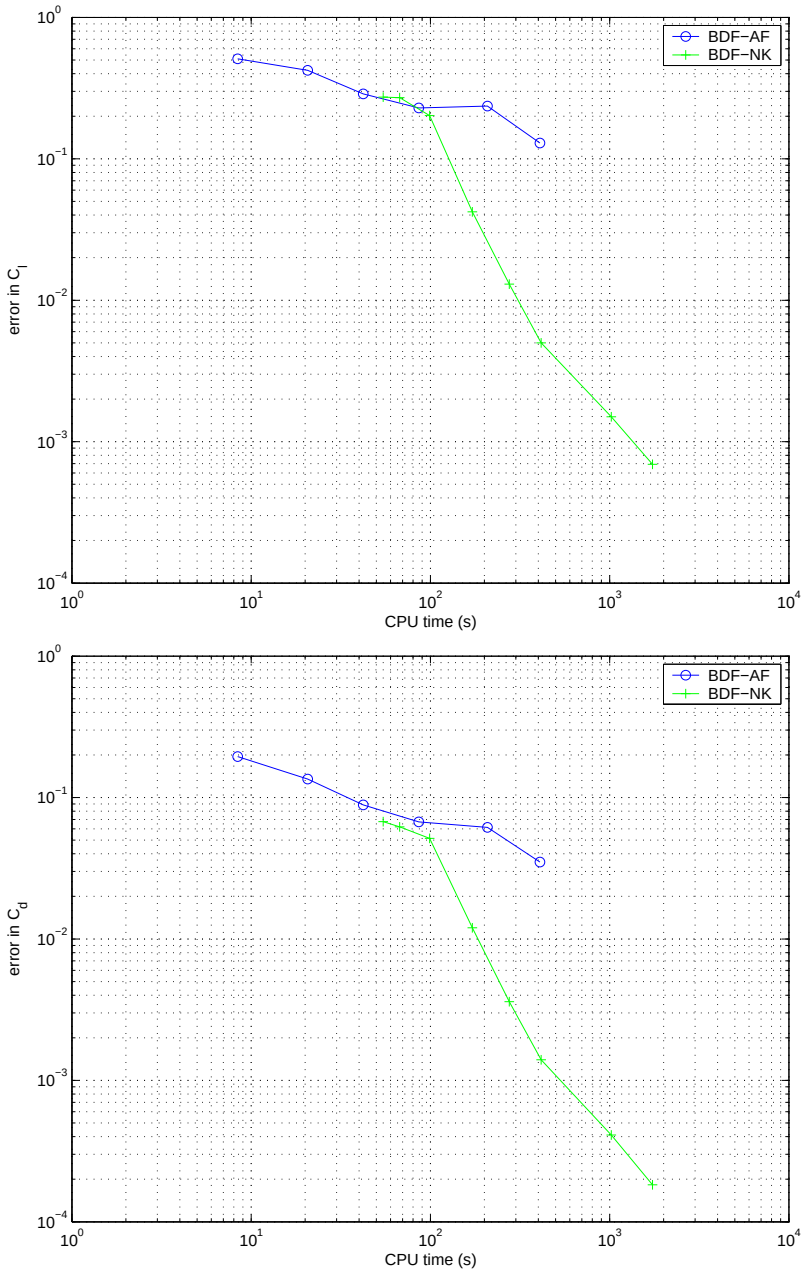

Figure 4.19: Comparison of subiteration algorithms for the BDF time-marching method (medium grid)

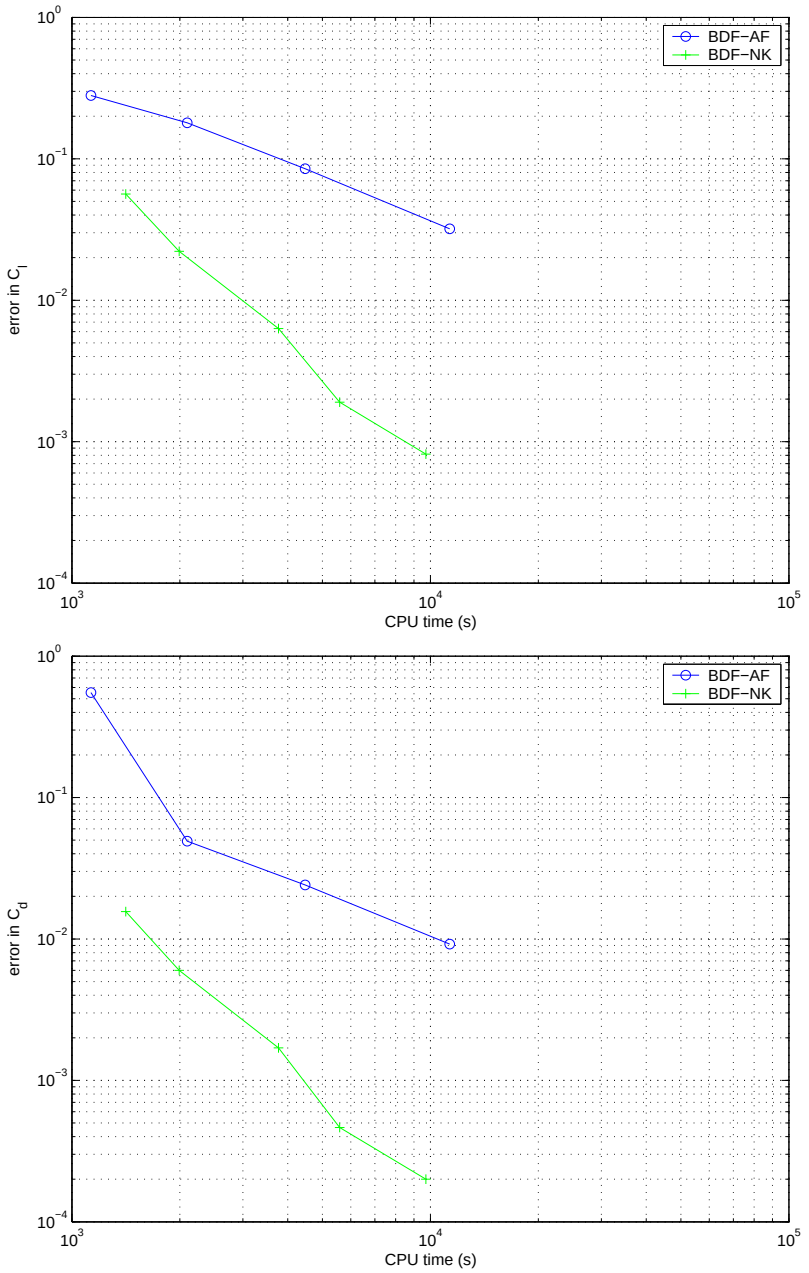

Figure 4.20: Comparison of subiteration algorithms for the BDF time-marching method (fine grid)

on the medium the approximate factorization subiterations take far too long to converge. On the fine grid, the problem with the approximate factorization subiterations was even greater and no reliable solutions could be obtained. Therefore, for ESDIRK the Newton-Krylov method should be used.

ESDIRK vs BDF

Cylinder Figure 4.23 shows the error versus the cpu time required to solve the cylinder test case. The graph shows the two time-marching methods using approximate factorization subiterations. At higher error levels, the BDF scheme performs better than the ESDIRK scheme; however, if greater accuracy is required, the ESDIRK scheme performs better.

Airfoil Figures 4.24, 4.25, and 4.26 show the efficiency of the BDF scheme in terms of cpu time for the coarse, medium, and fine grids respectively. For all grids, the lift and drag results give similar trends. At higher error levels, the BDF time-marching method is more efficient. If errors less than 10^{-3} are required, however, the ESDIRK time-marching method performs better. Therefore, for studies requiring high accuracy in time, the ESDIRK time-marching method with Newton-Krylov subiterations is the most efficient of the methods investigated.

4.3 Accuracy

This section examines the error vs Δt for the BDF and ESDIRK time-marching methods. Only the results obtained using the Newton-Krylov subiterations are presented for the airfoil test cases. It is important to note that the graphs in this section do not represent the efficiency of the methods.

4.3.1 Cylinder

Figure 4.27 presents the error in lift and drag versus the time step size for the cylinder test case. As in the previous section, the errors are found by integrating the difference from the reference solution over several periods. The solutions obtained using the ESDIRK time-marching scheme are far more accurate than the solutions found using BDF at any given time step size. For example, using $\Delta t = 0.5$ with the ESDIRK scheme gives

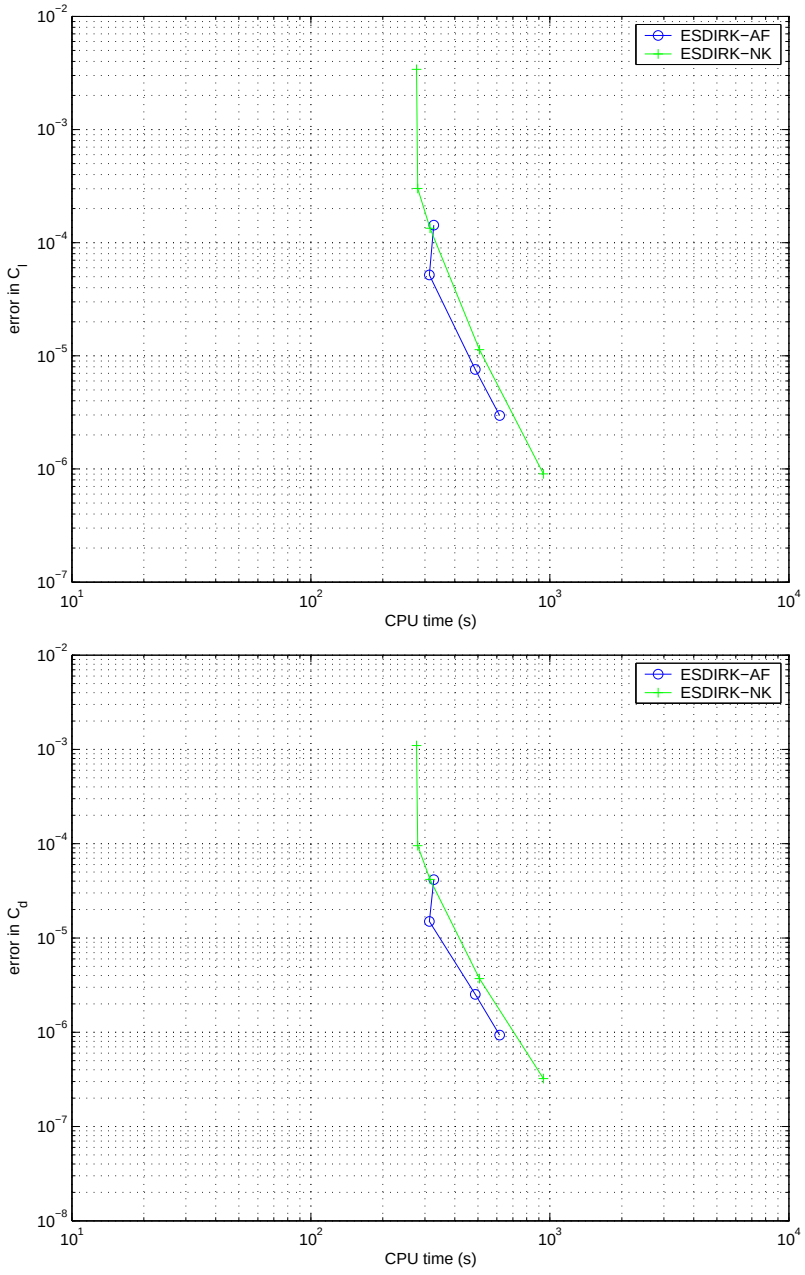

Figure 4.21: Comparison of subiteration algorithms for the ESDIRK time-marching method (coarse grid)

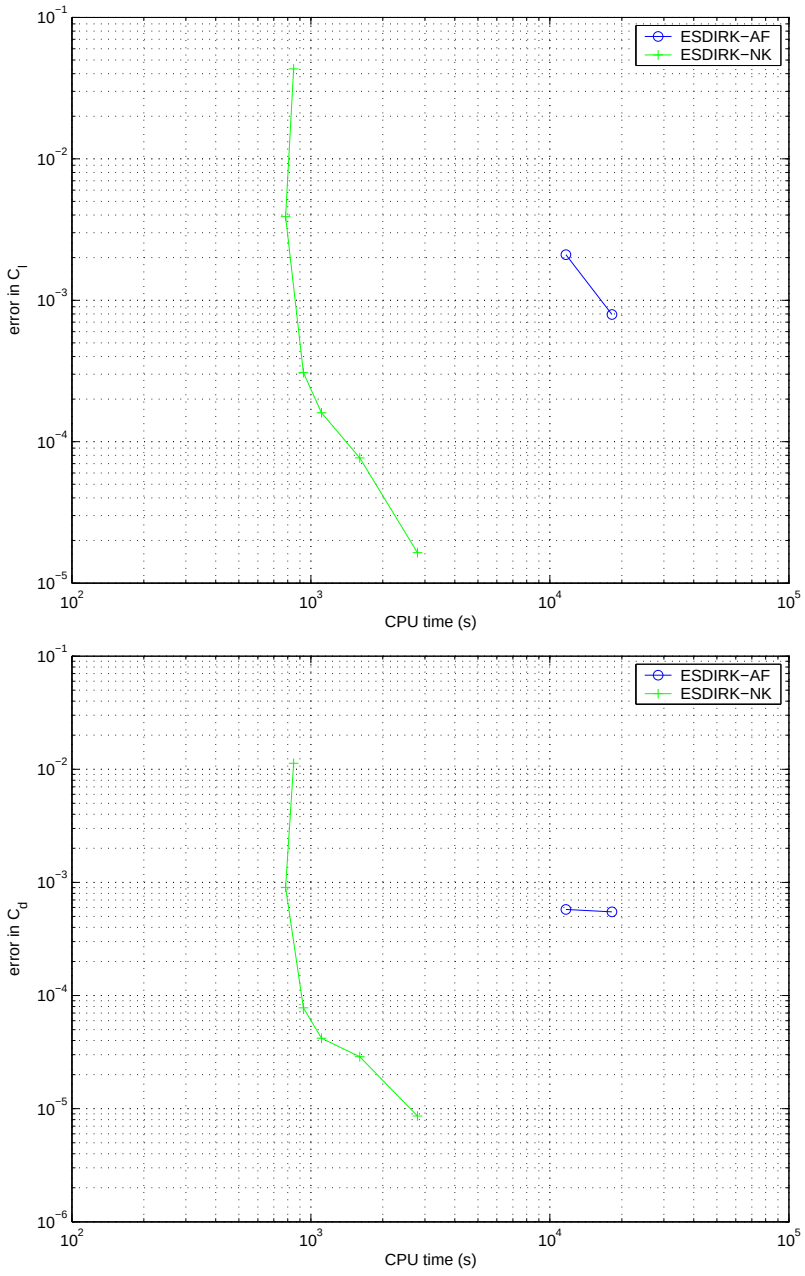

Figure 4.22: Comparison of subiteration algorithms for the ESDIRK time-marching method (medium grid)

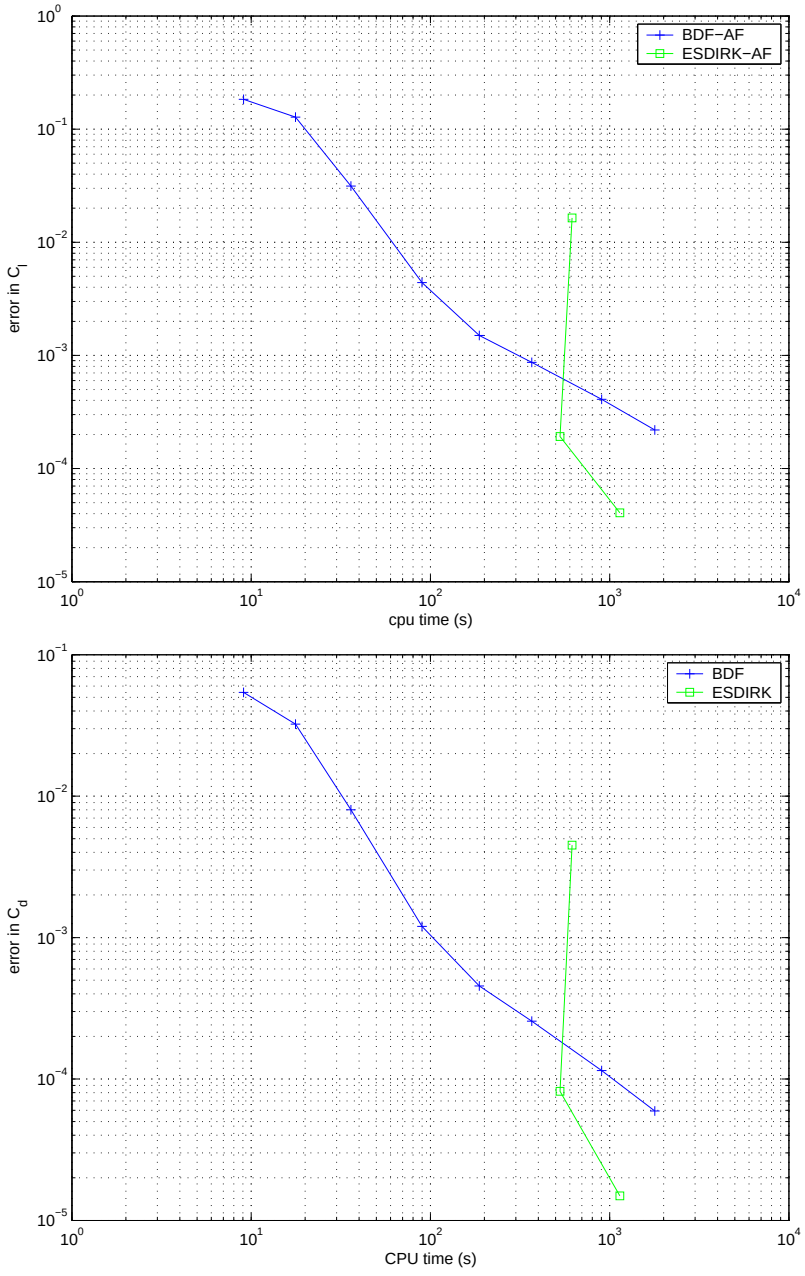

Figure 4.23: Error vs CPU time for laminar flow over cylinder

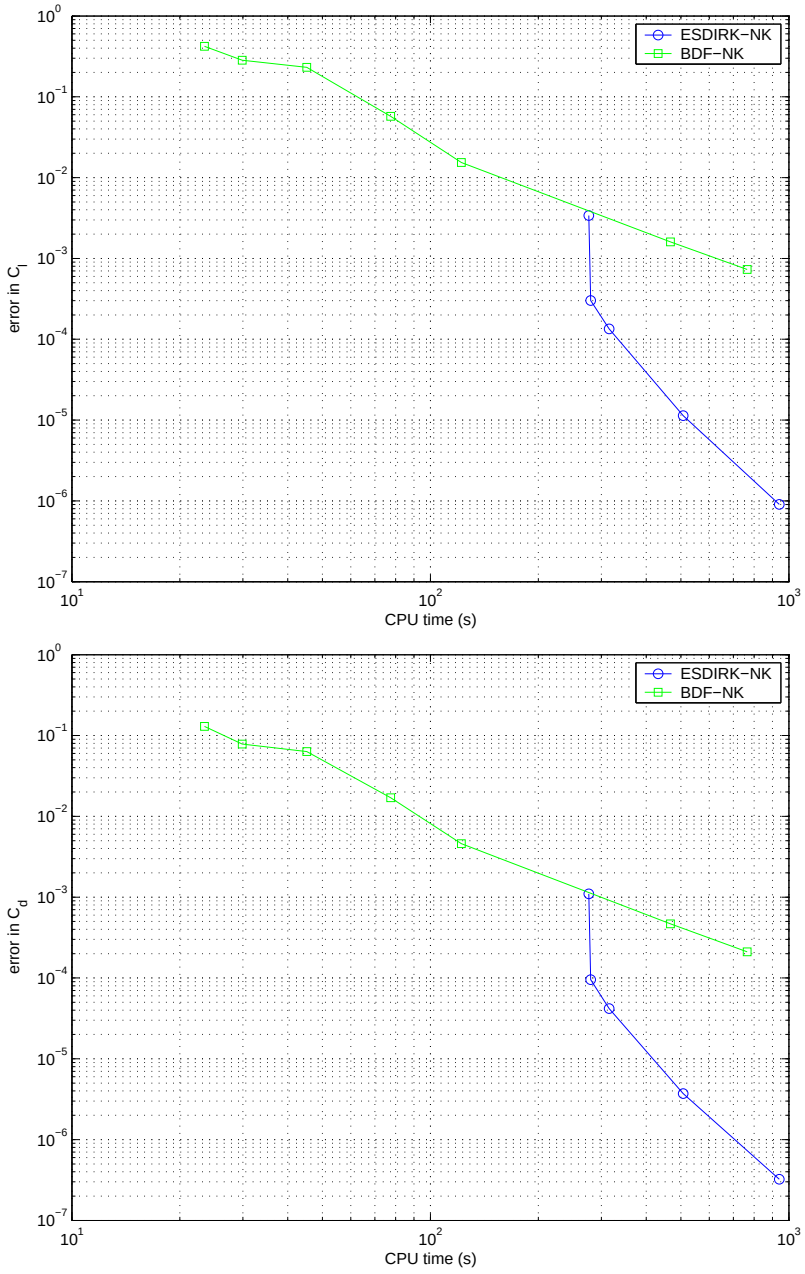

Figure 4.24: Efficiency comparison of ESDIRK and BDF (coarse grid)

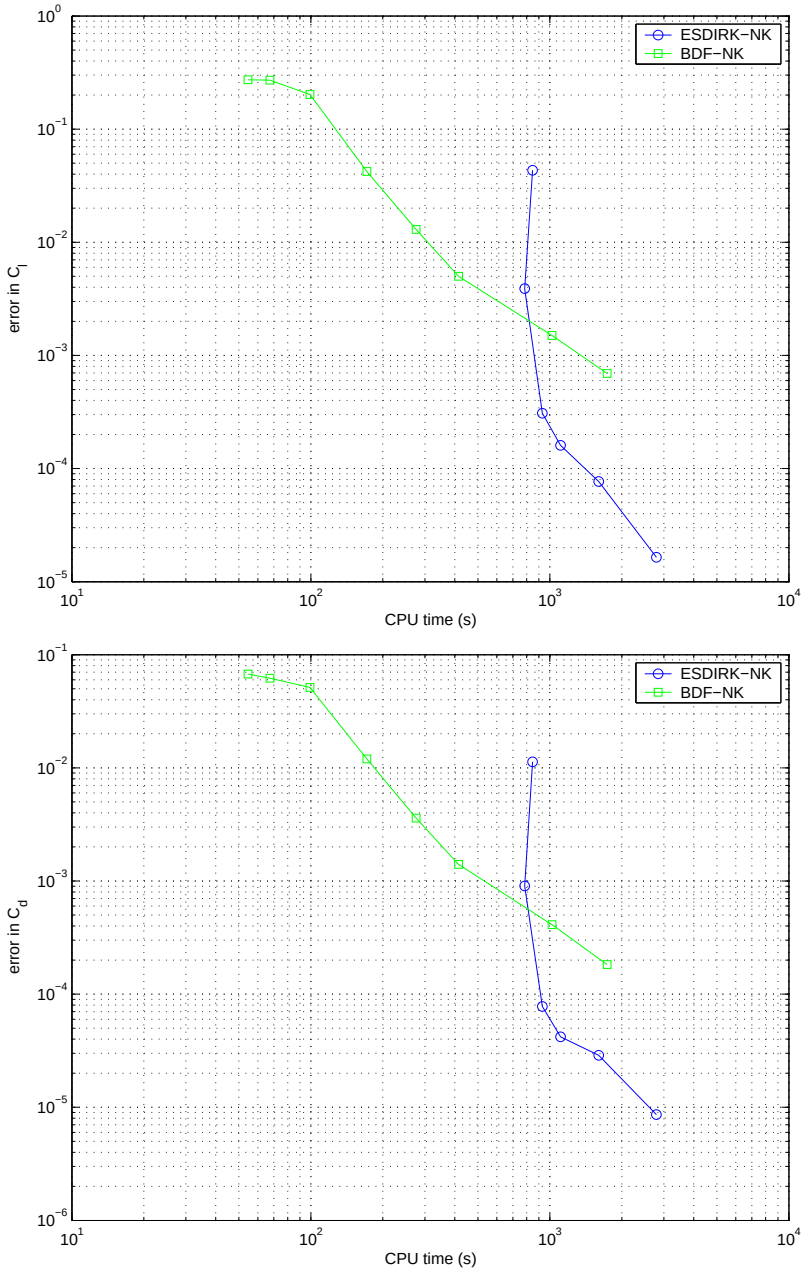

Figure 4.25: Efficiency comparison of ESDIRK and BDF (medium grid)

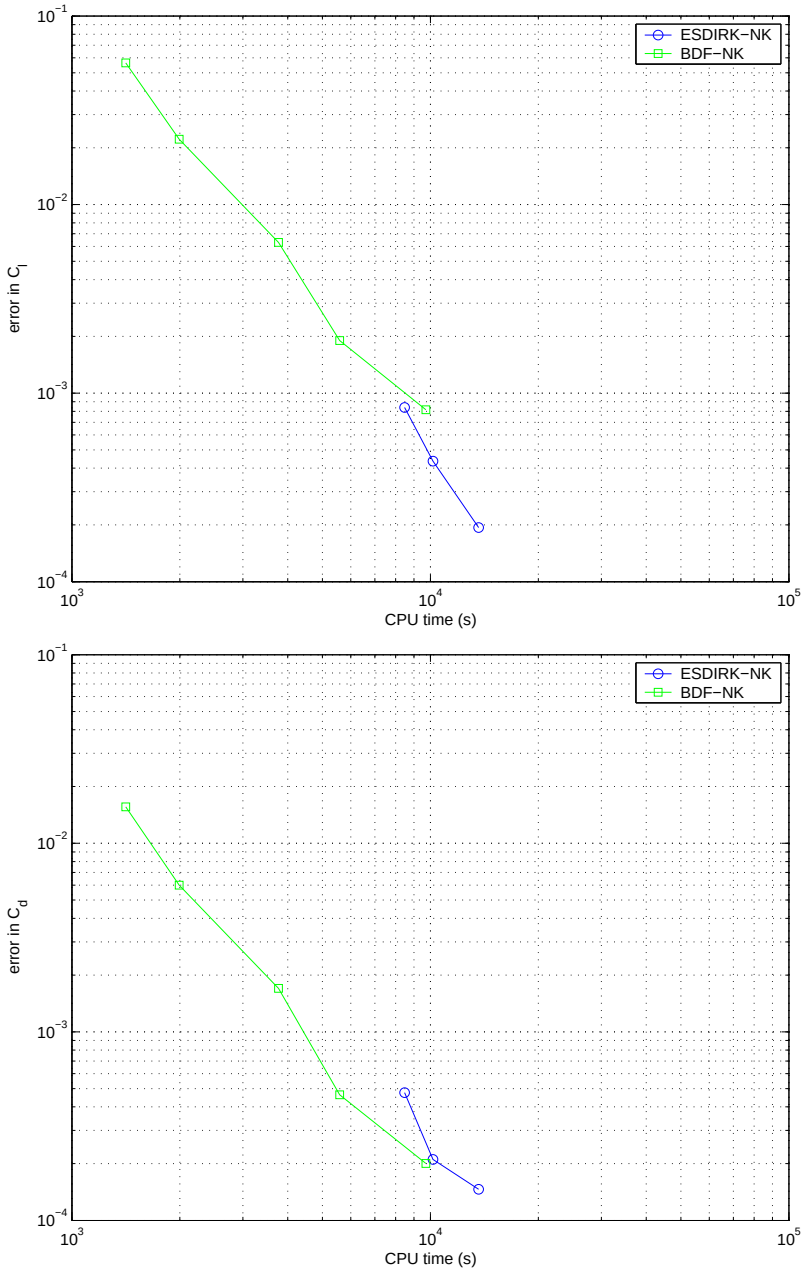

Figure 4.26: Efficiency comparison of ESDIRK and BDF (fine grid)

an error in lift of about $3.5 * 10^{-3}$. In order to obtain a solution with the same error using BDF2 a time step that is ten times smaller is required.

4.3.2 Airfoil

The error in lift and drag versus the time step size is shown in Figures 4.28, 4.29, and 4.30 for the coarse, medium and fine grids respectively.

The graphs for the airfoil test case are very similar to the results obtained for the cylinder case. Again, solutions obtained using the ESDIRK time-marching scheme are far more accurate than the solutions found using BDF at any given time step size.

The slopes shown in this section do not agree with the order of the methods. The reason is that the results shown in this section were for the optimal solutions from the previous section. The slopes of the lines would increase if the residuals from the nonlinear problem were reduced further. Reducing the residual for the nonlinear residual using the ESDIRK time-marching scheme resulted in a slope of 3.69. It is not clear why the slope of the graph was not 4.

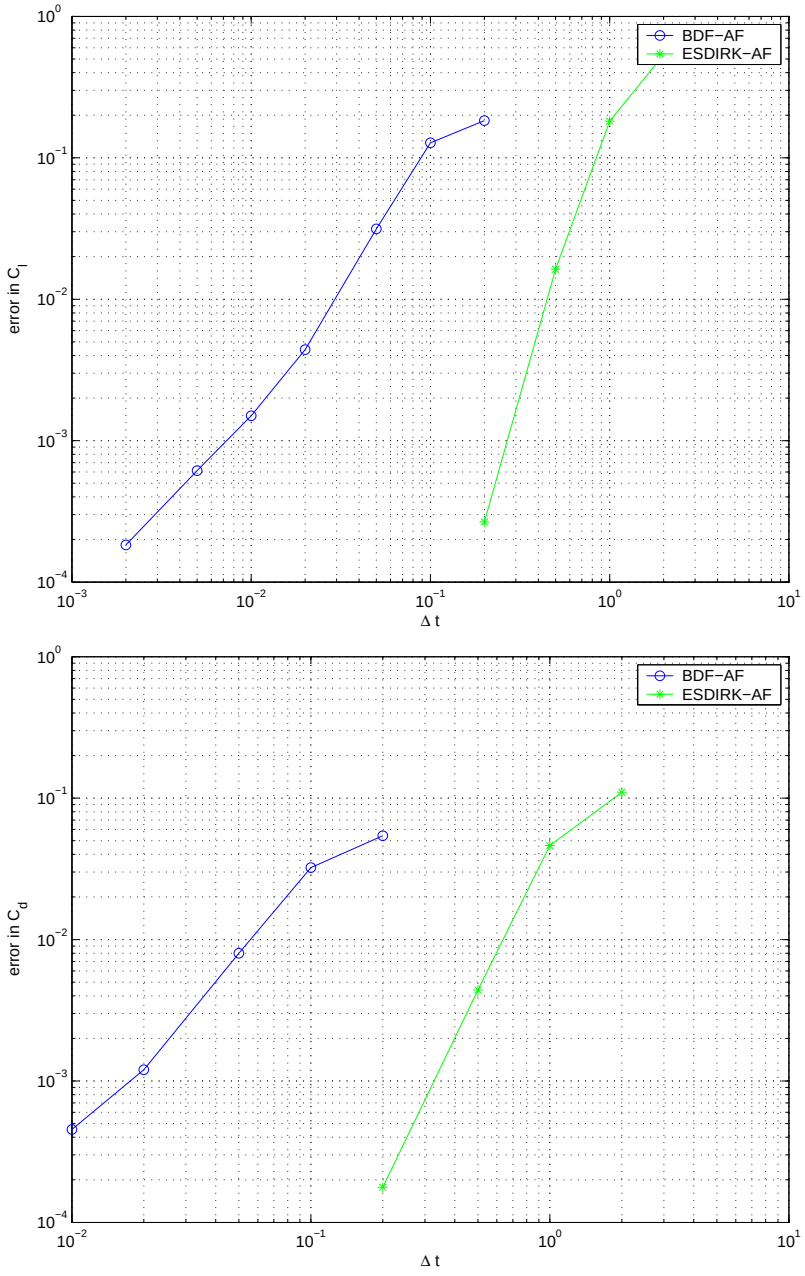

Figure 4.27: Error vs Δt for laminar flow over cylinder

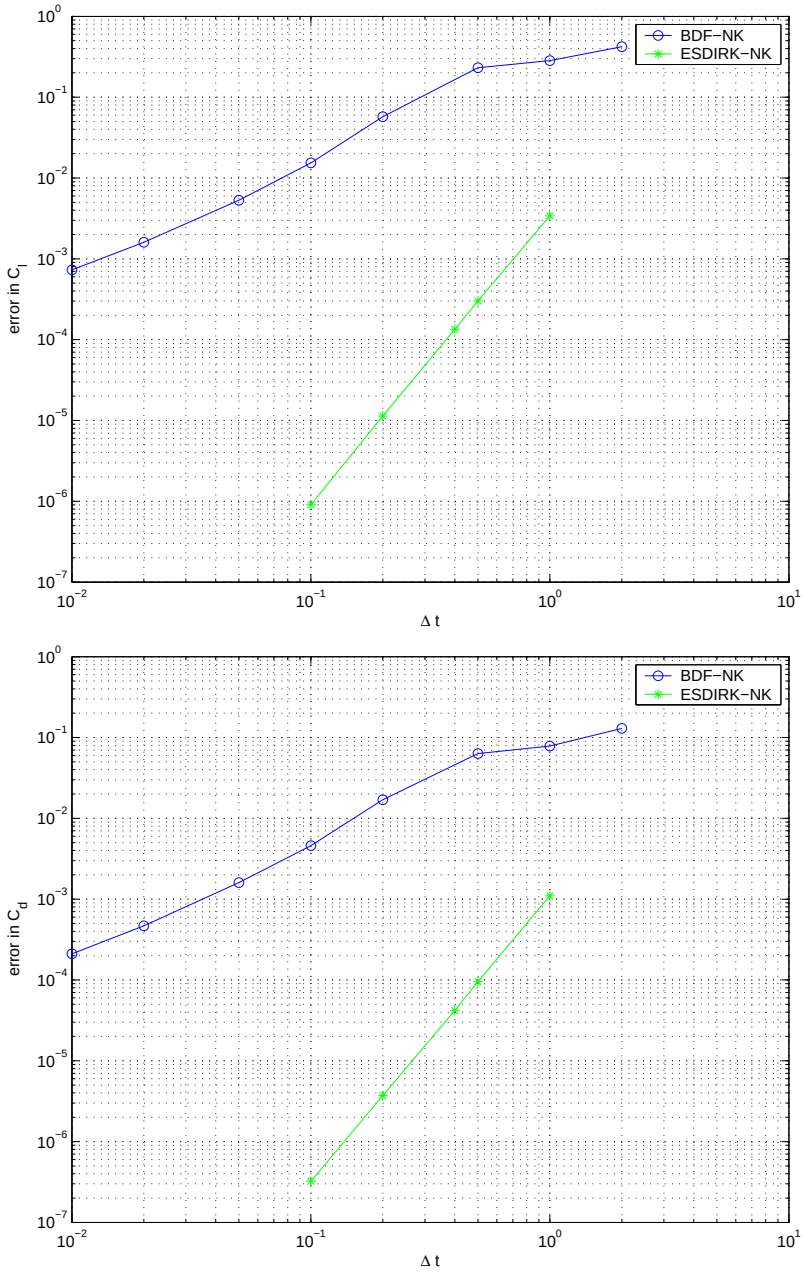

Figure 4.28: Error vs time step size for laminar flow over NACA 0012 airfoil (coarse grid)

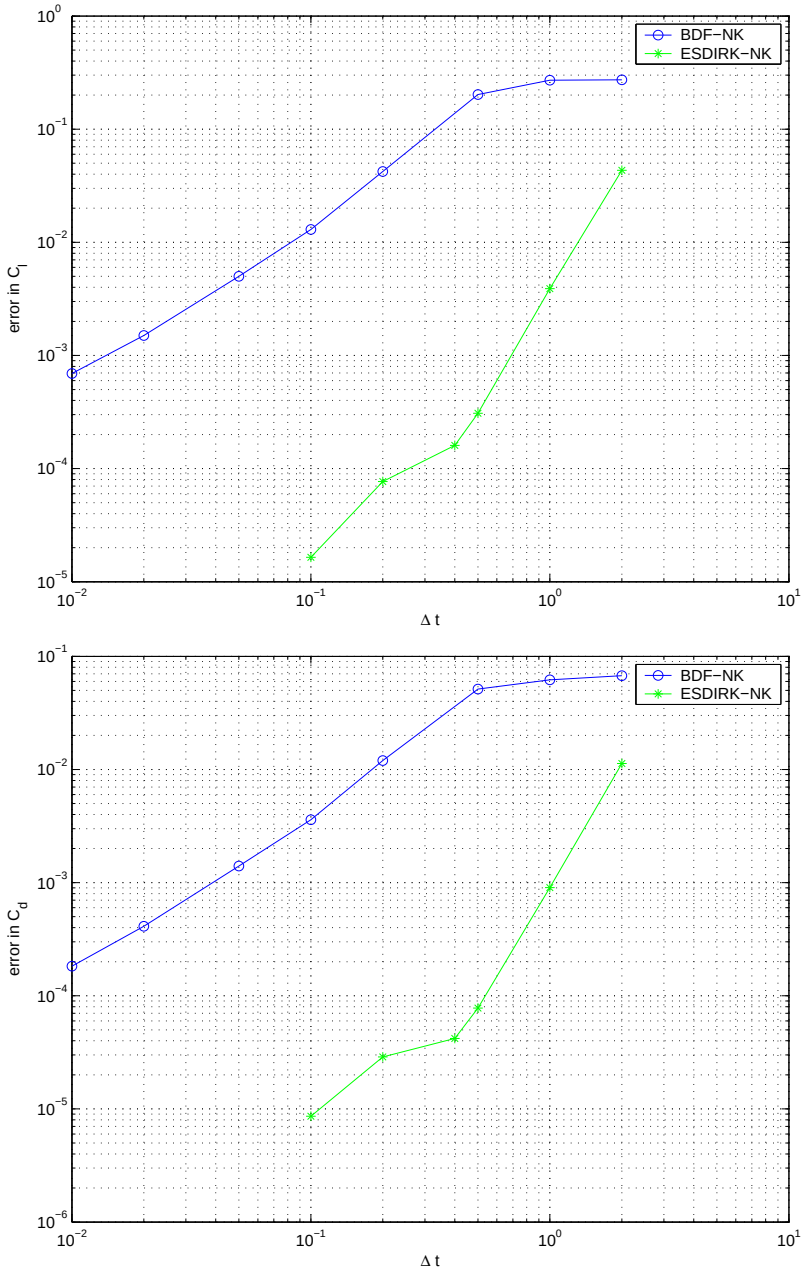

Figure 4.29: Error vs time step size for laminar flow over NACA 0012 airfoil (medium grid)

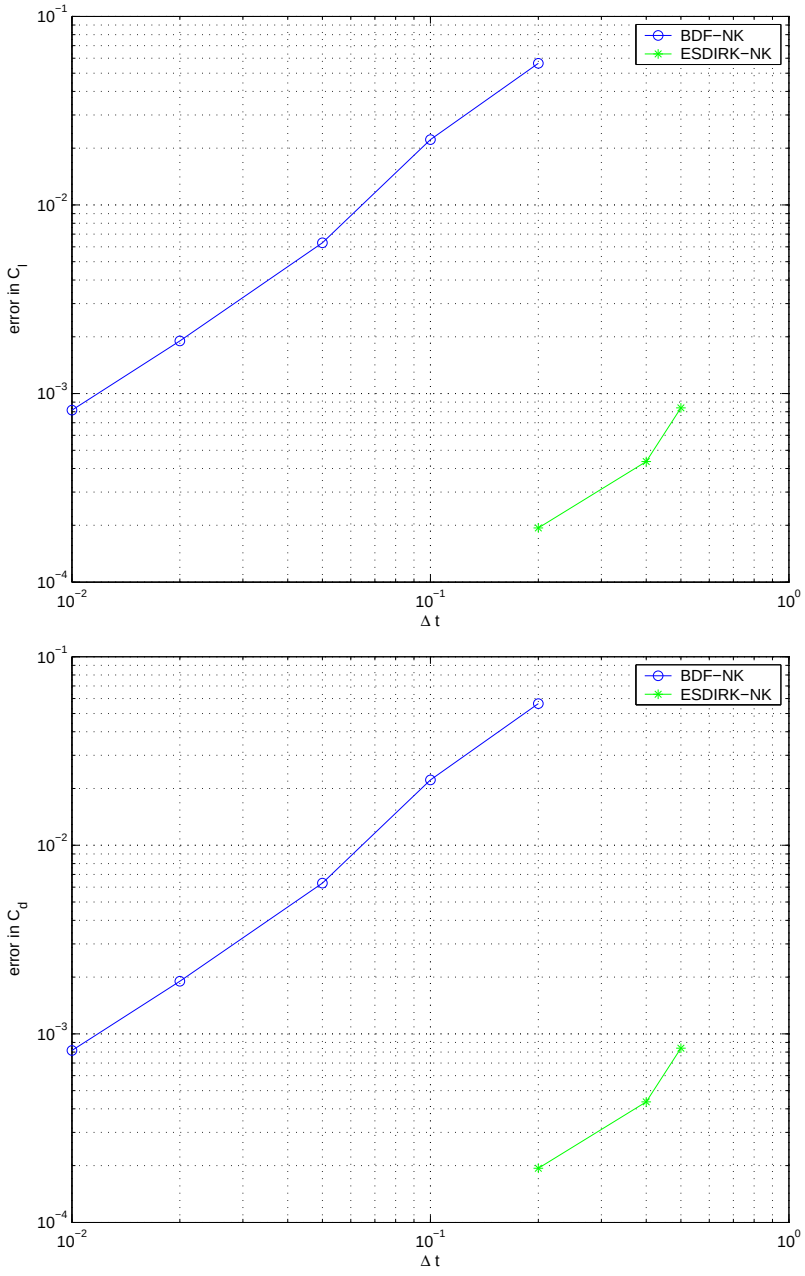

Figure 4.30: Error vs time step size for laminar flow over NACA 0012 airfoil (fine grid)

References

[1] Bijl, H., Carpenter, M. H., and Vatsa, V. N. Time integration schemes for the unsteady Navier-Stokes equations. Tech. Rep. No. 2001-2612, AIAA, June 2001.

[2] Bijl, H., Carpenter, M. H., Vatsa, V. N., and Kennedy, C. A. Implicit time integration schemes for the unsteady compressible Navier-Stokes equations: Laminar flow. *Journal of Computational Physics Vol. 179* (March 2002), pp. 313–329.

[3] Carpenter, M. H., Viken, S. A., and Nielsen, E. J. The efficiency of high order temporal schemes. Tech. Rep. No. 2003-0086, AIAA, January 2003.

[4] De Bonis, R. J., and Scott, J. Large-eddy simulation of a turbulent compressible round jet. *AIAA Journal Vol. 40*, No. 7 (July 2002), pp. 1346–1354.

[5] De Rango, S. Implicit Navier-Stokes computations of unsteady flows using subiteration methods. Master's thesis, University of Toronto Institute of Aerospace Studies, January 1996.

[6] De Rango, S., and Zingg, D. W. Improvements to a dual-time-stepping method for computing unsteady flows. *AIAA Journal Vol. 35*, No. 9 (1999), pp. 1548–1550.

[7] Ffowcs Willimams, J., and Hawkings, D. Sound generated by turbulence and surfaces in arbitrary motion. *Philosophical Transactions of the Royal Society of London Vol. A264*, No. 1151 (1969), pp. 321–342.

[8] Jameson, A., Schmidt, W., and Turkel, E. Numerical solutions of the Euler equations by finite volume methods using Runge-Kutta time-stepping. Tech. Rep. No. 81-1259, AIAA, June 1981.

[9] Khorrami, M. R., Berkman, M. E., and Choudhari, M. Unsteady flow computations of a slat with a blunt trailing edge. *AIAA Journal Vol. 38*, No. 11 (November 2000), pp. 2050–2058.

[10] Lomax, H., Pulliam, T. H., and Zingg, D. W. *Fundamentals of Computational Fluid Dynamics*. Springer-Verlag, 2001.

[11] Nemec, M. *Optimal Shape Design of Aerodynamic Configurations: A Newton-Krylov Approach*. PhD thesis, University of Toronto, November 2002.

[12] Nielsen, E. J., Anderson, W. K., Walters, R., and Keyes, D. Application of Newton-Krylov methodology to a three-dimensional unstructured Euler code. Tech. Rep. No. 95-1740, AIAA, June 1995.

[13] Pueyo, A. *An Efficient Newton-Krylov Method for the Euler and Navier-Stokes Equations*. PhD thesis, University of Toronto, December 1997.

[14] Saad, Y., and Schultz, M. H. GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems. *SIAM Journal on Scientific and Statistical Computing* (1986), pp. 856–869.

[15] Singer, B. A., Lockard, D. P., and Brentner, K. S. Computational aeroacoustic analysis of slat trailing-edge flow. *AIAA Journal Vol. 38*, No. 9 (September 2000), pp. 1558–1564.

[16] Tam, C., and Webb, J. Dispersion-relation-preserving finite difference schemes for computational acoustics. *Journal of Compuational Physics Vol. 107* (1993), pp. 262–281.

[17] Tam, C. K. W., and Pastouchenko, N. Gap tones. *AIAA Journal Vol. 39*, No. 8 (August 2001), pp. 1442–1447.

[18] Venkateswaran, S., and Merkle, C. L. Dual time stepping and preconditioning for unsteady computations. Tech. Rep. No. 95-0078, AIAA, January 1995.

Appendix A

Dual Time Stepping Scheme

The following is a detailed derivation of the dual time-step subiteration scheme with approximate factorization and diagonalization. We begin with the equation for dual time stepping:

$$\begin{aligned} \frac{\partial \hat{Q}}{\partial \tau} &= -G(\hat{Q}) \\ &= -\hat{Q} + \mathcal{N} - \alpha \Delta t \hat{D}(\hat{Q}) \end{aligned} \tag{A.1}$$

where G is given by Eq. (3.1) for the 2^{nd}-order backwards differencing and by Eq. (3.2) for the ESDIRK method, $\alpha = \frac{2}{3}$ for 2^{nd}-order BDF and $\alpha = a_{kk}$ for ESDIRK, $\mathcal{N}$ contains values which are not re-evaluated during a time step. (i.e. $\mathcal{N}$ is not a function of the pseudo-time step(τ). For 2^{nd}-order BDF:

$$\mathcal{N} = \frac{4}{3}\hat{Q}_n - \frac{1}{3}\hat{Q}_{n-1} \tag{A.2}$$

and for 4^{th}-order ESDIRK:

$$\mathcal{N} = Q_n - \Delta t \sum_{j=1}^{k-1} a_{kj} \hat{D}(\hat{Q}_j) \tag{A.3}$$

Applying implicit Euler time-marching to Eq. (A.1) gives

$$\hat{Q}_{p+1} = \hat{Q}_p - \Delta \tau G(\hat{Q}_{p+1}) \tag{A.4}$$

where $\Delta\tau$ is the pseudo-time-step size. $G(\hat{Q}_{p+1})$ can be linearized about time level p giving

$$\begin{aligned} G_{p+1} &= G_p + \frac{\partial G_p}{\partial \hat{Q}_p} \Delta \hat{Q}_p \\ &= G_p + \left(I + \alpha \Delta t \frac{\partial \hat{D}_p}{\partial \hat{Q}_p} \right) \Delta \hat{Q}_p \end{aligned} \tag{A.5}$$

Substituting Eq. (A.5) into Eq. (A.4) gives

$$\hat{Q}_{p+1} = \hat{Q}_p - \tau \left(I + \alpha \Delta t \frac{\partial \hat{D}_p}{\partial \hat{Q}_p} \right) - \Delta \tau G(\hat{Q}_p) \tag{A.6}$$

Rearanging the above equations and simplifying gives

$$\begin{aligned}
\left[I + \Delta \tau I + \alpha \Delta t \Delta \tau \hat{D}'(\hat{Q}_p) \right] \Delta \hat{Q}_p &= -\Delta \tau G(\hat{Q}_p) \\
\left[\frac{1}{\alpha \Delta t} I + \frac{\Delta \tau}{\alpha \Delta t} I + \Delta \tau \hat{D}'(\hat{Q}_p) \right] \Delta \hat{Q}_p &= -\frac{\Delta \tau}{\alpha \Delta t} G(\hat{Q}_p) \\
\left[S + \Delta \tau \hat{D}'(\hat{Q}_p) \right] \Delta \hat{Q}_p &= -\frac{\Delta \tau}{\alpha \Delta t} \left(\hat{Q}_p - \mathcal{N} + \alpha \Delta t \hat{D}(\hat{Q}_p) \right)
\end{aligned}$$

where

$$S = \left\{ \frac{1}{\alpha \Delta t} + \frac{\Delta \tau}{\alpha \Delta t} \right\} I$$

Applying approximate factorization and diagonalizing gives:

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

where

$$\hat{N}_d = T_\xi^{-1} S^{-1} T_\eta$$

Iterating to steady state, the solution for the stage (ESDIRK) or time step (BDF) will be taken as the most recent solution. Convergence of the inner iterations results in the exact solution to Eqs. (3.1)and (3.2)

Appendix B

Coefficients for ESDIRK4

The following is the Butcher table for the 4^{th}-order ESDIRK scheme used in this thesis.

0	0	0	0	0	0	0
$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$	0	0	0	0
$\frac{83}{250}$	$\frac{8611}{62500}$	$\frac{-1743}{31250}$	$\frac{1}{4}$	0	0	0
$\frac{31}{50}$	$\frac{5012029}{34652500}$	$\frac{-654441}{2922500}$	$\frac{174375}{388108}$	$\frac{1}{4}$	0	0
$\frac{17}{20}$	$\frac{15267082809}{155376265600}$	$\frac{-71443401}{120774400}$	$\frac{730878875}{902184768}$	$\frac{2285395}{8070912}$	$\frac{1}{4}$	0
1	$\frac{82889}{524892}$	0	$\frac{15625}{83664}$	$\frac{69875}{102672}$	$\frac{-2260}{8211}$	$\frac{1}{4}$
b_i	$\frac{82889}{524892}$	0	$\frac{15625}{83664}$	$\frac{69875}{102672}$	$\frac{-2260}{8211}$	$\frac{1}{4}$

Table B.1: Butcher table for 4^{th}-order ESDIRK