A Three-Dimensional Newton-Krylov Navier-Stokes Flow Solver Using a One-Equation Turbulence Model

by

David C. W. Kam

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

Copyright © 2007 by David C. W. Kam

Abstract

A Three-Dimensional Newton-Krylov Navier-Stokes Flow Solver Using a One-Equation Turbulence Model

David C. W. Kam
Masters of Applied Science
Graduate Department of Aerospace Engineering
University of Toronto
2007

A three-dimensional Navier-Stokes flow solver for structured grids is developed using a Newton-Krylov algorithm. Turbulence is modelled using the Spalart-Allmaras one-equation model. Spatial derivatives are approximated using centered-differences with second and fourth-difference dissipation. The equations are transformed into curvilinear coordinates and linearized using Newton's method. The reverse Cuthill-McKee method is applied to reorder the equations, and a preconditioner is formed based on an incomplete lower-upper factorization of a first-order approximate Jacobian. The linear system at each Newton iteration is then solved using a generalized minimal residual method. An approximate-Newton startup phase is followed by a Jacobian-free inexact-Newton phase.

Various aspects of the code are studied in this work. A parametric study attempting to optimize the performance of the code was performed on the following variables: ILU fill-levels, inexact-Newton convergence parameter, residual reduction tolerance parameter for switching to the inexact-Newton phase, and a preconditioning parameter used in the formation of the approximate Jacobian. The solver was validated by comparing with a well-validated flow solver, OPTIMA2D, and a grid convergence study is presented for laminar and turbulent flows about the ONERA M6 wing.

Acknowledgements

I would like to begin by thanking Professor David W. Zingg for his patience, encouragement and expertise throughout these last two years. It has been a great privilege and honour to have studied and worked under his tutelage.

Some of my most memorable moments at UTIAS have been in the company of the varied and eclectic group of characters with whom I shared the small confines of the lab. In particular, I have benefited greatly from the friendship of John Gatsis and Mo Tabesh, both of whom made the lab such a fun and colourful place. The computer expertise of Scott Northrup, and the help of Jason Hicken in learning to use TYPHOON, ICEMCFD, and Tecplot were also much appreciated. For this, and to everyone else in the lab, I would like to say thank you.

A special thanks goes out to my parents for their encouragement and support in this endeavour and all others. Their sacrifices made possible everything I have today.

Finally, I would like to thank Tessa Yeung for her friendship throughout these two years. She has always been there, with an encouraging word when I was disheartened, a sympathetic ear when I needed to talk, and a hot meal when I had an empty stomach. This thesis would not have been possible without her.

David C. W. Kam

University of Toronto Institute for Aerospace Studies
October 5, 2007

Contents

Abstract **iii**

Acknowledgment's **v**

1 Introduction **1**

1.1 Motivation 1

1.2 Background 1

1.3 Objectives 3

2 Governing Equations **5**

2.1 Navier-Stokes Equations 5

2.2 Turbulence Model 7

2.3 Curvilinear Coordinate Transformation 9

2.4 Boundary Conditions 10

2.4.1 Far-field Boundary 10

2.4.2 Solid Wall 11

3 Numerical Algorithm **13**

3.1 Spatial Discretization 13

3.1.1 Inviscid Fluxes 13

3.1.2 Viscous Fluxes 15

3.1.3 Turbulence Model 16

3.1.4 Boundary Conditions 17

3.2 Linearization and Newton's Method 19

3.2.1 Linearization of the Interior Scheme 21

3.2.2 Linearization of the Boundary Conditions 23

3.3 Solution to the Linear Problem 26

    3.3.1 Jacobian-free GMRES 26
    3.3.2 Inexact-Newton Method 26
    3.3.3 Preconditioner 27
    3.3.4 Incomplete LU Factorization 27
    3.3.5 Reverse Cuthill-McKee Nodal Reordering 28
  3.4 Algorithm Startup 28
    3.4.1 Local Time-Stepping 28

4 Grid Generation **31**
  4.1 Blocking Strategy 31
  4.2 Problem Definition 31
  4.3 Smoothing 32

5 Algorithm Optimization **37**
  5.1 Test Cases 37
  5.2 Parametric Study 38
    5.2.1 ILU Fill Level (k) 38
    5.2.2 Preconditioning Parameter (σ) 39
    5.2.3 Inexact-Newton Parameter (η) 46
    5.2.4 Approximate-Newton Convergence Parameter ($R_{d_{tol}}$) 46
  5.3 Summary of Optimal Parameters 47

6 Results and Validation **51**
  6.1 2D Validation 51
    6.1.1 Blasius Solution 51
    6.1.2 NACA0012 Airfoil 52
  6.2 3D Validation 53
    6.2.1 Grid Convergence Studies 58

7 Conclusions **63**
  7.1 Summary 63
  7.2 Recommendations 64

Appendices **69**

A 3D Curvilinear Coordinate Transformation 71

A.1 3D Viscous Flux Terms in Curvilinear Coordinates 73

A.2 Turbulence Model in Curvilinear Coordinates 75

B Vorticity Differentiation 79

B.1 Vorticity Differention Terms for S_1 82

B.2 Vorticity Differention Terms for S_2 84

B.3 Vorticity Differention Terms for S_3 86

C TYPHOON Input Files 89

C.1 Subsonic Turbulent Flow 89

C.2 Transonic Turbulent Flow 90

List of Tables

2.1 Logic for far-field boundary conditions using Riemann invariants 11

4.1 Orthogonal smoothing parameters in ICEMCFD 34

5.1 ONERA M6 wing test cases 37
5.2 Grids used in the ONERA M6 wing parametric study 38
5.3 List of parameters in the numerical study 38
5.4 Summary of results from parametric study 50

6.1 Flow conditions for the 2D validation test cases 52
6.2 Summary of grid characteristics used in the 3D laminar convergence study 58
6.3 TYPHOON laminar lift and drag grid convergence results for the ONERA M6 at M=0.5, Re=600, α=3.0° 58
6.4 Summary of grid characteristics used in 3D turbulent convergence study . 60

List of Figures

3.1 Halo nodes on a two-dimensional slice of two three-dimensional blocks for a 13-point stencil 20

4.1 TYPHOON's 12-block configuration 32
4.2 Surface mesh on an ONERA M6 wing 33
4.3 Mesh inversion at the wingtip and leading edge intersection due to changes in wingtip spacing 34
4.4 Mesh smoothed using multiblock method (a) and resulting flow solution (b); Mesh smoothed using orthogonality method (c) and resulting flow solution (d). 35

5.1 Laminar subsonic ILU fill level parameter (k) optimization; approximate-Newton phase (M=0.5, $\alpha = 3.0°$, Re=600, grid A) 39
5.2 Laminar subsonic ILU fill level parameter (k) optimization; Jacobian-free phase (M=0.5, $\alpha = 3.0°$, Re=600, grid A) 40
5.3 Laminar transonic ILU fill level parameter (k) optimization; approximate-Newton phase (M=0.84, $\alpha = 3.0°$, Re=600, grid B) 40
5.4 Laminar transonic ILU fill level parameter (k) optimization; Jacobian-free phase (M=0.84, $\alpha = 3.0°$, Re=600, grid B) 41
5.5 Turbulent subsonic ILU fill level parameter (k) optimization; approximate-Newton phase (M=0.5, $\alpha = 3.0°$, Re=2.88×10^6, grid C) 41
5.6 Turbulent subsonic ILU fill level parameter (k) optimization; Jacobian-free phase (M=0.5, $\alpha = 3.0°$, Re=2.88×10^6, grid D) 42
5.7 Turbulent transonic ILU fill level parameter (k) optimization; approximate-Newton phase (M=0.84, $\alpha = 3.0°$, Re=2.88×10^6, grid C) 42
5.8 Turbulent transonic ILU fill level parameter (k) optimization; Jacobian-free phase (M=0.84, $\alpha = 3.0°$, Re=2.88×10^6, grid D) 43

5.9 Laminar subsonic preconditioning parameter (σ) optimization (M=0.5, α = 3.0°, Re=600, grid A) 43
5.10 Laminar transonic preconditioning parameter (σ) optimization (M=0.84, α = 3.0°, Re=600, grid B) 44
5.11 Turbulent subsonic preconditioning parameter (σ) optimization (M=0.5, α = 3.0°, Re=2.88$\times10^6$, grid C) 44
5.12 Turbulent transonic preconditioning parameter (σ) optimization (M=0.84, α = 3.0°, Re=2.88$\times10^6$, grid D) 45
5.13 Laminar inexact-Newton parameter (η_{AN}) optimization; approximate-Newton phase (grid A) 46
5.14 Laminar inexact-Newton parameter (η_{JF}) optimization; Jacobian-free phase (grid A) 47
5.15 Turbulent inexact-Newton parameter (η_{AN}) optimization; approximate-Newton phase (grid C) 48
5.16 Turbulent inexact-Newton parameter (η_{JF}) optimization; Jacobian-free phase (grid D) 48
5.17 Laminar approximate-Newton residual reduction tolerance parameter ($R_{d_{tol}}$) - grids A and B 49
5.18 Turbulent approximate-Newton residual reduction tolerance parameter ($R_{d_{tol}}$) - grids C and D 49

6.1 Comparison of TYPHOON surface velocity profile against Blasius solution 52
6.2 The computational grid used in the Blasius solution comparison. Note that every second node has been removed for the purposes of this graphic. Mesh lines are in red, while block boundaries are in black 53
6.3 TYPHOON laminar subsonic flow over a NACA0012 airfoil (case 1) . . . 54
6.4 TYPHOON laminar transonic flow over a NACA0012 airfoil (case 2) . . 55
6.5 TYPHOON turbulent subsonic flow over a NACA0012 airfoil (case 3) . . 56
6.6 TYPHOON turbulent transonic flow over a NACA0012 airfoil (case 4) . . 57
6.7 TYPHOON convergence histories for laminar grid convergence study on the ONERA M6 wing 59
6.8 Pressure contours of an ONERA M6 wing at M=0.5, Re=600, α=3.0° (800k-node grid) 60
6.9 ONERA M6 wing C_p distribution, $M = 0.6998, \alpha = 0.04°$, $\mathcal{R}e$= 11.74$\times10^6$ 61

6.10 TYPHOON convergence histories for the 3D turbulent grid convergence study 62

6.11 Pressure contours of an ONERA M6 wing at M=0.6998, Re=11.74×10^6, α=0.04° (500k-node grid) 62

Chapter 1

Introduction

1.1 Motivation

The rising cost of fossil fuels has been of increasing concern in the aviation industry over the last decade. The desire to reduce costs, as well as to minimize environmental impact has lead to the development of aerodynamically efficient aircraft. This is of particular urgency in the commercial aircraft industry, where consumer demands are forecast to increase in every sector into the foreseeable future [34] .

Since the early 1970's, computational fluid dynamics has emerged as the third branch in aerodynamics, complementing the experimental and theoretical fields [1]. Computational methods offer a cost-effective alternative to the experimental testing that has traditionally been done in windtunnels. The rapid advances in the CFD field has been made largely possible by increasing processor speeds throughout the last three decades. However, the ability today to accurately and efficiently solve the Navier-Stokes equations on full wing-body geometries would also not have been possible without significant improvements in algorithms.

1.2 Background

The backbone of all CFD algorithms is the set of Navier-Stokes equations which completely describes a Newtonian fluid flow. A set of nonlinear partial differential equations, it can be linearized and discretized at a finite number of points in the computational domain. The manner in which this discretization occurs leads to the two main groups of CFD codes: unstructured and structured. The former uses cells that may vary in geometry, while the latter can handle only quadrilateral (2D) or tetrahedrals (3D) cells. Unstructured grids typically are used in conjunction with finite-volume schemes, while structured grids can be solved with both finite-volume and finite-difference schemes. Unstructured grids can be generated on much more complex geometries and refinement at a given area in the flow field can be accomplished by splitting cells as required. Thus unstructured grids are well-suited to mesh adaptation methods. Structured grids have less freedom in refinement, although the use of node clustering, flaring, and multiblock methods serve to mitigate some of these concerns. One drawback with unstructured grids is the need to track the relative and absolute locations of each cell in a large indexing system. In contrast, structured grids by nature have a linear ordering of nodes.

Time-stepping methods are typically used to march the nonlinear equations to a converged solution. Two basic classes of such methods exist: implicit and explicit. Explicit methods are easier to implement and cheaper to compute, but are prone to instability problems which may either require impractically small time steps and many iterations, or cause the solution to diverge altogether. Implicit methods on the other hand pay a storage and computation penalty for solving a more difficult formulation, while allowing for larger time steps and fewer iterations. This is due to their inherent stability. One famous example of an implicit solver is the ARC2D algorithm developed by Pulliam and Steger [29, 33] at the NASA Ames Research Center.

The ability to create fast and efficient Navier-Stokes flow solvers is generally limited by the time it takes to solve the large linear system of equations that arises from Newton's method. Direct inversion of such matrices is prohibitively expensive. A very effective method for two-dimensional flows is to approximately factor the matrix into two matrices, each containing entries from one of the two coordinate directions [29]. This method was superseded by current methods such as approximate-Newton and Jacobian-free inexact-Newton schemes which use an iterative method to inexactly solve the linear system and have been found to be much faster [27, 28]. The inexact method can drastically reduce the amount of time the algorithm spends at each Newton iteration. The success of iterative solvers depends on the properties of the matrix it is applied to. For non-symmetric indefinite systems such as those arising from the application of an inexact Newton method to the Navier-Stokes equations, a Krylov subspace method using a generalized minimal residual (GMRES) method [30] is frequently used [36, 4, 23, 8, 26].

When compared to other iterative approaches, GMRES was shown to be faster than the stabilized biconjugate gradient, and conjugate gradient squared methods [27, 28].

Codes such as HURRICANE [37] and OPTIMA2D [22] both use the Newton-Krylov algorithm. One reason for the popularity of the GMRES method is that it does not require the storage of the Jacobian, but can instead function using only matrix-vector products. Pueyo and Zingg showed that this Newton-Krylov method outperformed approximate-factorization, approximate-factorization with multigrid, and also the Jacobian-present formulation [27, 28].

A survey of other three-dimensional Navier-Stokes solvers reveals a wide variety of solution methods and grid techniques.

NSU3D by Mavriplis [18, 13] uses a line/point Jacobi method, while Frink uses a Gauss-Seidel-like method in USM3D [7]. Both NSU3D and USM3D are 3D Navier-Stokes flow solvers that use a node-centered finite-volume formulation.

Another unstructured finite-volume flow solver is FUN3D. The solution to the linear system of equations is obtained by using implicit point relaxation, implicit line relaxation, or the Newton-Krylov algorithm [26].

OVERFLOW [12] is a Navier-Stokes flow solver that uses finite-differencing with matrix dissipation on overset meshes. An approximate factorization or a lower-upper symmetric Gauss-Seidel method can be used on the linear system [11]. In addition, grid sequencing and multigrid are used to accelerate code convergence and preconditioning is performed to improve performance at low Mach numbers.

Flo3xx, by May and Jameson, is a finite-volume code that uses a Gauss-Seidel method to solve the linear system and multigrid to accelerate convergence. It can solve flows on both structured and unstructured meshes and is designed to handle both cell-centered and node-centered discretizations [19].

It is useful to note that all the Navier-Stokes algorithms described above are coupled with the Spalart-Allmaras one-equation turbulence model. This model has become quite popular in the computational aerodynamics community and has been shown to be a robust and accurate alternative to two-equation models such as the $k - \omega$ and $k - \epsilon$ techniques.

1.3 Objectives

Nichols and Zingg [25] successfully demonstrated the speed and accuracy with which the Newton-Krylov algorithm can be applied to the Euler equations in three dimensions. This multiblock solver is known as TYPHOON. An efficient and robust flow solver is now desired for real-world aerodynamic optimizations at high Reynolds numbers. Thus, the need to expand TYPHOON's ability to handle viscous flows is established.

The focus of this thesis is to apply the full Navier-Stokes equations to the existing algorithm, and to integrate the Spalart-Allmaras turbulence model into the flow equations.

Chapter 2

Governing Equations

This chapter presents a summary of the governing equations used in the numerical algorithm. The first section details the mean flow equations; the second deal, with the Spalart-Allmaras one-equation turbulence model.

2.1 Navier-Stokes Equations

The Navier-Stokes equations are a collection of conservation equations that together fully describe fluid flow. This set of mathematical statements consists of the conservation of mass, momentum, and energy. The conservative form of the compressible Navier-Stokes equations is as follows:

$$\frac{\partial Q}{\partial t} + \frac{\partial E}{\partial x} + \frac{\partial F}{\partial y} + \frac{\partial G}{\partial z} = \frac{1}{Re}\Big(\frac{\partial E_v}{\partial x} + \frac{\partial F_v}{\partial y} + \frac{\partial G_v}{\partial z}\Big) \tag{2.1}$$

where Re, the Reynolds number, is given by the following equation:

$$Re = \frac{\rho_\infty c a_\infty}{\mu_\infty} \tag{2.2}$$

The flow variables are contained in Q, while the inviscid fluxes are given in E,F and G:

$$Q = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e \end{bmatrix}, \quad E = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ u(e+p) \end{bmatrix}, \quad F = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vw \\ v(e+p) \end{bmatrix}, \quad G = \begin{bmatrix} \rho w \\ \rho uw \\ \rho vw \\ \rho w^2 + p \\ w(e+p) \end{bmatrix}$$

where ρ is the density, ρu, ρv and ρw are the components of momentum, and e is the total energy. Pressure, p, is related to the energy e via the equation of state for a perfect gas:

$$p = (\gamma - 1)\left[e - \frac{1}{2}\rho(u^2 + v^2 + w^2)\right] \tag{2.3}$$

The value of γ is 1.4. The viscous flux tensors E_v, F_v and G_v in Equation 2.1 are given by:

$$E_v = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ E_{v,5} \end{bmatrix}, \quad F_v = \begin{bmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ \tau_{yz} \\ F_{v,5} \end{bmatrix}, \quad G_v = \begin{bmatrix} 0 \\ \tau_{zx} \\ \tau_{zy} \\ \tau_{zz} \\ G_{v,5} \end{bmatrix} \tag{2.4}$$

where the shear and normal stress terms are given by:

$$\begin{aligned}
\tau_{xx} &= 2(\mu + \mu_t)u_x - \tfrac{2}{3}\mu\left(u_x + v_y + w_z\right) \\
\tau_{xy} &= (\mu + \mu_t)\left(u_y + v_x\right) \\
\tau_{xz} &= (\mu + \mu_t)\left(u_z + w_x\right) \\
\tau_{yy} &= 2(\mu + \mu_t)v_y - \tfrac{2}{3}\mu\left(u_x + v_y + w_z\right) \\
\tau_{yz} &= (\mu + \mu_t)\left(v_z + w_y\right) \\
\tau_{zz} &= 2(\mu + \mu_t)w_z - \tfrac{2}{3}\mu\left(u_x + v_y + w_z\right) \\
E_{v,5} &= u\tau_{xx} + v\tau_{xy} + w\tau_{xz} + (\mu\mathcal{P}r^{-1} + \mu_t\mathcal{P}r_t^{-1})(\gamma - 1)^{-1}\partial_x(a^2) \\
F_{v,5} &= u\tau_{yx} + v\tau_{yy} + w\tau_{yz} + (\mu\mathcal{P}r^{-1} + \mu_t\mathcal{P}r_t^{-1})(\gamma - 1)^{-1}\partial_y(a^2) \\
G_{v,5} &= u\tau_{zx} + v\tau_{zy} + w\tau_{zz} + (\mu\mathcal{P}r^{-1} + \mu_t\mathcal{P}r_t^{-1})(\gamma - 1)^{-1}\partial_z(a^2)
\end{aligned}$$

where μ and μ_t are the dynamic laminar and dynamic turbulent eddy viscosities respectively. $\mathcal{P}r$ and $\mathcal{P}r_t$ are the laminar and turbulent Prandtl numbers, assumed as constants 0.72 and 0.90. The sound of speed, a, as calculated using the ideal gas assumption:

$$a = \sqrt{\gamma R T} \tag{2.5}$$

where R is the specific gas constant, and T is the temperature.

The above equations have been nondimensionalized using the following relations:

$$x = \frac{\bar{x}}{c}, \quad y = \frac{\bar{y}}{c}, \quad z = \frac{\bar{z}}{c}, \quad u = \frac{\bar{u}}{a_\infty}, \quad v = \frac{\bar{v}}{a_\infty}, \quad w = \frac{\bar{w}}{a_\infty}, \quad \rho = \frac{\bar{\rho}}{\rho_\infty}, \quad e = \frac{\bar{e}}{\rho_\infty a_\infty^2}, \quad t = \frac{\bar{t}a_\infty}{c}$$

where c is the chord length. The subscript ∞ identifies freestream values, and the overbar, dimensional values.

The dynamic laminar viscosity μ is found using Sutherland's viscosity law [35] given by:

$$\mu = \frac{\bar{\mu}}{\mu_\infty} = \left(\frac{\bar{T}}{T_\infty}\right)^{3/2} \frac{T_\infty + S^*}{\bar{T} + S^*} \tag{2.6}$$

where S^* is the Sutherland constant of 198.6° R for air, T_∞ is the freestream temperature of 460.0° R for air.

2.2 Turbulence Model

The effects of turbulence in the Navier-Stokes equations are captured using Spalart and Allmaras' one-equation model. An eddy viscosity-like term $\tilde{\nu}$ is solved as the sixth flow variable in the equations. The model has been shown to work well not only for two dimensional flows [9], but also three dimensional ones [16]. The equation is as follows:

$$\begin{aligned}\frac{D\tilde{\nu}}{Dt} = &\underbrace{\frac{c_{b1}}{Re}[1 - f_{t2}]\tilde{S}\tilde{\nu}}_{\text{Production}} + \underbrace{\frac{1 + c_{b2}}{\tilde{\sigma} Re}\nabla \cdot [(\nu + \tilde{\nu})\nabla\tilde{\nu}] - \frac{c_{b2}}{\tilde{\sigma} Re}(\nu + \tilde{\nu})\nabla^2\tilde{\nu}}_{\text{Advection and Diffusion}} \\ &- \underbrace{\frac{1}{Re}\left[c_{w1} f_w - \frac{c_{b1}}{\kappa^2} f_{t2}\right]\left(\frac{\tilde{\nu}}{d}\right)^2}_{\text{Destruction}} + \underbrace{Re f_{t1} \Delta U^2}_{\text{Trip}}\end{aligned} \tag{2.7}$$

where the first term is the production term, the second and third the combined advection/diffusion terms, the fourth the destruction term, and the last, a laminar/turbulent transition trip term. For the purposes of this thesis, the trip functions f_{t1} and f_{t2} are set to zero.

The kinematic eddy viscosity, $\nu_t = \mu_t/\rho$, is then calculated from the solution to this equation through:

$$\nu_t = \tilde{\nu} f_{v1} \tag{2.8}$$

where:

$$f_{v1} = \frac{\chi^3}{\chi^3 + c_{v1}^3} \tag{2.9}$$

and χ is the ratio of the kinematic eddy turbulent viscosity to the kinematic laminar viscosity $\nu = \mu/\rho$:

$$\chi = \frac{\tilde{\nu}}{\nu} \tag{2.10}$$

A modified vorticity term, $\tilde{S}$ is used in the production and destruction terms, where:

$$\tilde{S} = S + \frac{\tilde{\nu}}{\kappa^2 d^2} f_{v2} \tag{2.11}$$

and where d is the distance to the closest solid wall surface node. κ is the von Karman constant equal to 0.41 and S is the magnitude of the vorticity, written as:

$$S = \left[\left(\frac{\partial w}{\partial y} - \frac{\partial v}{\partial z} \right)^2 + \left(\frac{\partial u}{\partial z} - \frac{\partial w}{\partial x} \right)^2 + \left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \right)^2 \right]^{\frac{1}{2}} \tag{2.12}$$

f_{v2} is a viscous function described by:

$$f_{v2} = 1 - \frac{\chi}{1 + \chi f_{v1}} \tag{2.13}$$

The wall destruction term includes a function f_w given as:

$$f_w = g \left(\frac{1 + c_{w3}^3}{g^6 + c_{w3}^6} \right)^{\frac{1}{6}} \tag{2.14}$$

where

$$g = r + c_{w2}(r^6 - r) \tag{2.15}$$

with r, a nondimensional variable defined as:

$$r \equiv \frac{\tilde{\nu}}{\tilde{S} \kappa^2 d^2} \tag{2.16}$$

The remaining parameters in this turbulence model are as follows:

$$\begin{array}{ll} c_{b1} = 0.1355 & c_{b2} = 0.622 \\ c_{w1} = c_{b1}/\kappa^2 + (1 + c_{b2})/\tilde{\sigma} & c_{w2} = 0.3 \\ c_{w3} = 0.2 & c_{v1} = 7.1 \\ \tilde{\sigma} = 2/3 & \end{array}$$

Modifications to the vorticity-like term $\tilde{S}$ are made to ensure that it is non-negative [2]:

$$\tilde{S} = S f_{v3} + \frac{\tilde{\nu}}{\kappa^2 d^2} f_{v2} \tag{2.17}$$

where

$$f_{v2} = \left(1 + \frac{\chi}{c_{v2}} \right)^{-3} \tag{2.18}$$

$$f_{v3} = \frac{(1 + \chi f_{v1})(1 - f_{v2})}{\chi} \tag{2.19}$$

where c_{v2} is set at 5.0. This correction appears to improve the stability of the model.

2.3 Curvilinear Coordinate Transformation

Discretization of the physical domain about an airfoil is done using structured grids. The use of such grids allows for the direct mapping of physical boundaries to those in computational space. In addition, uniform computational grid spacing equal to one is obtained through the curvilinear transformation. The Navier-Stokes equations and turbulence model can then be solved in this mathematical environment. The benefit of using this type of grid includes the implicit indexing of the node locations since the grid spacing is equal to one. In the computational domain, the four independent variables x,y,z and t are mapped from the physical domain as follows:

$$\begin{aligned} \tau &= t \\ \xi &= \xi(x, y, z, t) \\ \eta &= \eta(x, y, z, t) \\ \zeta &= \zeta(x, y, z, t) \end{aligned} \tag{2.20}$$

The exact details of the transformation can be found in Appendix A, and were derived following the procedure described in [17].

The Navier-Stokes equations can then be described in the transformed coordinate system as:

$$\frac{\partial \hat{Q}}{\partial \tau} + \frac{\partial \hat{E}}{\partial \xi} + \frac{\partial \hat{F}}{\partial \eta} + \frac{\partial \hat{G}}{\partial \zeta} = \frac{1}{Re}\left(\frac{\partial \hat{E}_v}{\partial \xi} + \frac{\partial \hat{F}_v}{\partial \eta} + \frac{\partial \hat{G}_v}{\partial \zeta}\right) \tag{2.21}$$

where $\hat{(\cdot)} = J^{-1}(\cdot)$, and J is the metric Jacobian resulting from the transformation found to be:

$$J^{-1} = x_\xi y_\eta z_\zeta + y_\xi z_\eta x_\zeta + z_\xi x_\eta y_\zeta - x_\xi z_\eta y_\zeta - y_\xi x_\eta z_\zeta - z_\xi y_\eta x_\zeta \tag{2.22}$$

The inviscid fluxes are:

$$E = \begin{bmatrix} \rho U \\ \rho u U + \xi_x p \\ \rho v U + \xi_y p \\ \rho w U + \xi_z p \\ (e+p)U + \xi_t p \end{bmatrix}, \quad F = \begin{bmatrix} \rho V \\ \rho u V + \eta_x p \\ \rho v V + \eta_y p \\ \rho w V + \eta_z p \\ (e+p)V + \eta_t p \end{bmatrix}, \quad G = \begin{bmatrix} \rho W \\ \rho u W + \zeta_x p \\ \rho v W + \zeta_y p \\ \rho w W + \zeta_z p \\ (e+p)W + \zeta_t p \end{bmatrix} \tag{2.23}$$

where the contravariant velocities are given as:

$$\begin{aligned} U &= \xi_t + \xi_x u + \xi_y v + \xi_z w \\ V &= \eta_t + \eta_x u + \eta_y v + \eta_z w \\ W &= \zeta_t + \zeta_x u + \zeta_y v + \zeta_z w \end{aligned} \tag{2.24}$$

The viscous fluxes are:

$$\begin{aligned}\hat{E}_v &= J^{-1}\left(\xi_x E_v + \xi_y F_v + \xi_z G_v\right)\\ \hat{F}_v &= J^{-1}\left(\eta_x E_v + \eta_y F_v + \eta_z G_v\right)\\ \hat{G}_v &= J^{-1}\left(\zeta_x E_v + \zeta_y F_v + \zeta_z G_v\right)\end{aligned} \tag{2.25}$$

The shear stresses in the general curvilinear coordinate system are given by:

$$\begin{aligned}\tau_{xx} &= (\mu+\mu_t)\left[4\left(\xi_x u_\xi + \eta_x u_\eta + \zeta_x u_\zeta\right) - 2\left(\xi_y v_\xi + \eta_y v_\eta + \zeta_y v_\zeta + \xi_z w_\xi + \eta_z w_\eta + \zeta_z w_\zeta\right)\right]/3\\ \tau_{xy} &= (\mu+\mu_t)\left(\xi_y u_\xi + \eta_y u_\eta + \zeta_y u_\zeta + \xi_x v_\xi + \eta_x v_\eta + \zeta_x v_\zeta\right)\\ \tau_{xz} &= (\mu+\mu_t)\left(\xi_z u_\xi + \eta_z u_\eta + \zeta_z u_\zeta + \xi_x w_\xi + \eta_x w_\eta + \zeta_x w_\zeta\right)\\ \tau_{yy} &= (\mu+\mu_t)\left[4\left(\xi_y v_\xi + \eta_y v_\eta + \zeta_y v_\zeta\right) - 2\left(\xi_x u_\xi + \eta_x u_\eta + \zeta_x u_\zeta + \xi_z w_\xi + \eta_z w_\eta + \zeta_z w_\zeta\right)\right]/3\\ \tau_{yz} &= (\mu+\mu_t)\left(\xi_z v_\xi + \eta_z v_\eta + \zeta_z v_\zeta + \xi_y w_\xi + \eta_y w_\eta + \zeta_y w_\zeta\right)\\ \tau_{zz} &= (\mu+\mu_t)\left[4\left(\xi_z w_\xi + \eta_z w_\eta + \zeta_z w_\zeta\right) - 2\mu\left(\xi_x u_\xi + \eta_x u_\eta + \zeta_x u_\zeta + \xi_y v_\xi + \eta_y v_\eta + \zeta_y v_\zeta\right)\right]/3\end{aligned}$$

2.4 Boundary Conditions

At each of the six sides of a grid block, a single boundary condition must be specified. Four boundary types are possible: far-field, solid wall, symmetry and interior. The selection of these boundary conditions must be done carefully not only to preserve the physics but also to maintain the accuracy and stability of the problem.

2.4.1 Far-field Boundary

Inviscid Boundaries

Six equations need to be specified at the inviscid far-field boundaries. These boundaries are far enough away from a solid surface as to be relatively unaffected by viscous considerations. Depending on the nature of the flow, these values are either extrapolated from an interior node, or are set to freestream values.

Riemann invariants are used to determine the appropriate conditions for the far-field. The characteristic approach has the benefit of nonreflectivity, allowing the solution to move freely through the boundary. The first three Riemann invariants are:

$$\begin{aligned}R_1 &= V_n - \tfrac{2a}{\gamma-1}\\ R_2 &= V_n + \tfrac{2a}{\gamma-1}\\ R_3 &= \tfrac{p}{\rho^\gamma}\end{aligned} \tag{2.26}$$

	subsonic	supersonic
Inflow	Freestream: $R_{1,3,4,5,6}$ Extrapolation: R_2	Freestream: all Extrapolation: none
Outflow	Freestream: R_1 Extrapolation: $R_{2,3,4,5,6}$	Freestream: none Extrapolation: all

Table 2.1: Logic for far-field boundary conditions using Riemann invariants

where R_1, R_2 and R_3 are obtained from locally one-dimensional characteristics corresponding to $\lambda_1 = V_n - a$, $\lambda_2 = V_n + a$, and $\lambda_3 = V_n$ respectively and V_n is the outward normal velocity component at the boundary. Note that R_3 is an isentropic relation.

Three more equations are necessary:

$$\begin{aligned} R_4 &= V_{t1} \\ R_5 &= V_{t2} \\ R_6 &= \tilde{\nu} \end{aligned} \tag{2.27}$$

where V_{t1} and V_{t2} are tangential velocities, and $\tilde{\nu}$ is the turbulent eddy viscosity variable. Determination of the appropriate selection of extrapolation versus freestream values are summarized in Table 2.1

Viscous Outflow

At a viscous outflow boundary downsteam of the wing, a simple zeroth-order extrapolation of the flow variables provides a sufficient closure of the equations, as suggested by Pueyo and Zingg [27].

2.4.2 Solid Wall

In contrast to the inviscid wall conditions which need to satisfy flow tangency, a viscous flow about a solid surface needs to satisfy the no-slip condition:

$$u = 0, \quad v = 0, \quad w = 0 \tag{2.28}$$

In addition to the no-slip boundary conditions, the pressure is determined from:

$$\left(\frac{\partial p}{\partial n}\right)_{wall} = 0 \tag{2.29}$$

If an adiabatic wall condition is also assumed, then:

$$\left(\frac{\partial T}{\partial n}\right)_{wall} = 0 \tag{2.30}$$

Assuming a perfect gas, $p = \rho RT$, and using Equation 2.29, one can then arrive at the fifth wall boundary condition:

$$\left(\frac{\partial \rho}{\partial n}\right)_{wall} = 0 \tag{2.31}$$

For the turbulence model, the turbulent eddy viscosity is zero at the surface:

$$\tilde{\nu} = 0 \tag{2.32}$$

Chapter 3

Numerical Algorithm

This chapter details the spatial discretization, time-stepping method, and the efficient method by which the system of linearized equations is solved.

3.1 Spatial Discretization

Finite differencing of the mean flow equations follows the procedure developed by Steger [33] and Pulliam [29], and implemented in ARC2D. A scalar artificial dissipation scheme by Jameson et al. [10] is also used.

3.1.1 Inviscid Fluxes

Inviscid fluxes in the Navier-Stokes equations involve first-order derivatives. Here, the algorithm uses a three-point center-difference spatial discretization on the interior nodes. In the computational domain, the spacing is uniform and equal to one. Therefore, the finite difference approximation can be described as:

$$\left(\frac{\partial \hat{E}}{\partial \xi}\right)_{j,k,m} \approx \frac{\hat{E}_{j+1,k,m} - \hat{E}_{j-1,k,m}}{2} - A_{D_{j,k,m}} \tag{3.1}$$

$$\left(\frac{\partial \hat{F}}{\partial \eta}\right)_{j,k,m} \approx \frac{\hat{F}_{j,k+1,m} - \hat{E}_{j,k-1,m}}{2} - A_{D_{j,k,m}} \tag{3.2}$$

$$\left(\frac{\partial \hat{G}}{\partial \zeta}\right)_{j,k,m} \approx \frac{\hat{G}_{j,k,m+1} - \hat{G}_{j,k,m-1}}{2} - A_{D_{j,k,m}} \tag{3.3}$$

where A_D is an artificial dissipation term.

Artificial Dissipation

Numerical dissipation is implemented to damp out high frequency oscillations that arise due to the use of the centered difference method. These oscillations are particularly prevalent in regions of shocks where sharp pressure gradients are experienced, and need to be eliminated in order for the code to converge.

The method through which this is accomplished is a second and fourth-difference scalar dissipation scheme developed by Jameson et al. [10]:

$$A_{D_{j,k,m}} = \nabla_\xi \left(D^{(2\xi)}_{j+\frac{1}{2},k,m} + D^{(4\xi)}_{j+\frac{1}{2},k,m} \right) \tag{3.4}$$

where ∇ is the backward-difference operator. The second and fourth-difference dissipative terms, $D^{(2\xi)}_{j+\frac{1}{2},k,m}$ and $D^{(4\xi)}_{j+\frac{1}{2},k,m}$ are given by:

$$D^{(2\xi)}_{j+\frac{1}{2},k,m} = d^{(2)}_{j+\frac{1}{2},k,m} \nabla_\xi \left(J_{j,k,m} \hat{Q}_{j,k,m} \right) \tag{3.5}$$

$$D^{(4\xi)}_{j+\frac{1}{2},k,m} = d^{(4)}_{j+\frac{1}{2},k,m} \nabla_\xi \Delta_\xi \nabla_\xi \left(J_{j,k,m} \hat{Q}_{j,k,m} \right) \tag{3.6}$$

The second-difference coefficient, $d^{(2)}_{j+\frac{1}{2},k,m}$ is defined as:

$$d^{(2)}_{j+\frac{1}{2},k,m} = 2 \left(\epsilon \sigma^{(\xi)} J^{-1} \right)_{j+\frac{1}{2},k,m} \tag{3.7}$$

where $\sigma^{(\xi)}$, the spectral radius of the flux Jacobian $\frac{\partial \hat{E}}{\partial \hat{Q}}$ is:

$$\sigma^{(\xi)} = |U| + a\sqrt{\xi_x^2 + \xi_y^2 + \xi_z^2} \tag{3.8}$$

and ϵ is:

$$\epsilon_{j,k,m} = \kappa_2 \left[0.5 \Upsilon^*_{j,k,m} + 0.25 \left(\Upsilon^*_{j+1,k,m} + \Upsilon^*_{j-1,k,m} \right) \right] \tag{3.9}$$

where

$$\Upsilon^*_{j,k,m} = \max \left(\Upsilon_{j+1,k,m}, \Upsilon_{j,k,m}, \Upsilon_{j-1,k,m} \right) \tag{3.10}$$

$$\Upsilon_{j,k,m} = \frac{|p_{j+1,k,m} - 2p_{j,k,m} + p_{j-1,k,m}|}{|p_{j+1,k,m} + 2p_{j,k,m} + p_{j-1,k,m}|} \tag{3.11}$$

Equation 3.11 acts as a switch that detects the presence of shocks while Equation 3.10 is a smoothing function for Υ.

A fourth-difference dissipative term is used in smooth regions and is switched off in regions near shocks where the term is found to cause oscillations. The fourth-difference coefficient is given by:

$$d^{(4)}_{j+\frac{1}{2},k,m} = \max\left[0, 2\kappa_4\left(\sigma^{(\xi)}J^{-1}\right)_{j+\frac{1}{2},k,m} - d^{(2)}_{j+\frac{1}{2},k,m}\right] \tag{3.12}$$

Values for κ_2 and κ_4 are supplied by the user and are typically 1.0 and 0.02 respectively. The construction of the dissipative terms in the η and ζ directions are done in a similar way, except that Equation 3.11 is omitted in the ζ-direction following Nemec [22].

3.1.2 Viscous Fluxes

The discretization of the viscous fluxes requires a slightly different approach since they involve second order derivatives of the form:

$$\partial_\xi\left(\alpha_{j,k,m}\partial_\xi\beta_{j,k,m}\right) \tag{3.13}$$

It is desirable to maintain the three-point stencil for all interior nodes while preserving the accuracy of the scheme. Thus, the following compact three-point stencil (developed by Pulliam in [29]) is used:

$$\Delta_\xi\left(\alpha_{j+\frac{1}{2},k,m}\left(\nabla_\xi\beta_{j,k,m}\right)\right) = \alpha_{j+\frac{1}{2},k,m}\left(\beta_{j+1,k,m} - \beta_{j,k,m}\right) - \alpha_{j-\frac{1}{2},k,m}\left(\beta_{j-1,k,m} - \beta_{j,k,m}\right) \tag{3.14}$$

where Δ is the forward-difference operator and

$$\alpha_{j-\frac{1}{2},k,m} = \frac{\alpha_{j-1,k,m} + \alpha_{j,k,m}}{2} \qquad \text{and} \qquad \alpha_{j+\frac{1}{2},k,m} = \frac{\alpha_{j,k,m} + \alpha_{j+1,k,m}}{2} \tag{3.15}$$

are the values of α at the half points of the nodes in the ξ direction. The differencing in the η and ζ directions are done in a similar manner.

Where the viscous terms take the form:

$$\partial_\eta\left(\alpha_{j,k,m}\partial_\xi\beta_{j,k,m}\right) \tag{3.16}$$

the spatial discretization is performed as follows:

$$\begin{aligned}\partial_\eta\left(\alpha_{j,k,m}\partial_\xi\beta_{j,k,m}\right) \;\approx\;& \frac{1}{2}\alpha_{j,k+1,m}\left(\frac{\beta_{j+1,k+1,m} - \beta_{j-1,k+1,m}}{2}\right) \\ -\;& \frac{1}{2}\alpha_{j,k-1,m}\left(\frac{\beta_{j+1,k-1,m} - \beta_{j-1,k-1,m}}{2}\right)\end{aligned} \tag{3.17}$$

Similarly, the cross-derivatives in the other combinations of directions follow this procedure. Note that these terms are dropped during the building of the approximate flow Jacobian.

3.1.3 Turbulence Model

The discretization of the Spalart-Allmaras turbulence model is done in much the same way as the viscous and inviscid terms as described in the original paper [32] and by Godin [9].

Convection

Each of the three convection terms can be discretized using a first-order upwinding scheme:

$$\begin{aligned}
U\frac{\partial\tilde{\nu}}{\partial\xi} &\approx U^{+}_{i,j,k}\left(\tilde{\nu}_{i,j,k}-\tilde{\nu}_{i-1,j,k}\right)+U^{-}_{i,j,k}\left(\tilde{\nu}_{i+1,j,k}-\tilde{\nu}_{i,j,k}\right) && (3.18)\\
V\frac{\partial\tilde{\nu}}{\partial\eta} &\approx V^{+}_{i,j,k}\left(\tilde{\nu}_{i,j,k}-\tilde{\nu}_{i,j-1,k}\right)+V^{-}_{i,j,k}\left(\tilde{\nu}_{i,j+1,k}-\tilde{\nu}_{i,j,k}\right) && (3.19)\\
W\frac{\partial\tilde{\nu}}{\partial\zeta} &\approx W^{+}_{i,j,k}\left(\tilde{\nu}_{i,j,k}-\tilde{\nu}_{i,j,k-1}\right)+W^{-}_{i,j,k}\left(\tilde{\nu}_{i,j,k+1}-\tilde{\nu}_{i,j,k}\right) && (3.20)
\end{aligned}$$

where U, V, and W are the contravariant velocities, and the $+$ and $-$ superscripts indicate forward and backwards differencing respectively, as given by:

$$\begin{aligned}
U^{+}_{i,j,k} &= \frac{1}{2}\left(U_{i,j,k}+|U_{i,j,k}|\right), \quad U^{-}_{i,j,k}=\frac{1}{2}\left(U_{i,j,k}-|U_{i,j,k}|\right)\\
V^{+}_{i,j,k} &= \frac{1}{2}\left(V_{i,j,k}+|V_{i,j,k}|\right), \quad V^{-}_{i,j,k}=\frac{1}{2}\left(V_{i,j,k}-|V_{i,j,k}|\right)\\
W^{+}_{i,j,k} &= \frac{1}{2}\left(W_{i,j,k}+|W_{i,j,k}|\right), \quad W^{-}_{i,j,k}=\frac{1}{2}\left(W_{i,j,k}-|W_{i,j,k}|\right) && (3.21)
\end{aligned}$$

Diffusion

All cross derivative terms in the diffusion terms are dropped once they are transformed into curvilinear coordinates. The diffusive terms are then approximated by the following:

$$\begin{aligned}
\nabla\cdot\left[(\nu+\tilde{\nu})\nabla\tilde{\nu}\right] &\approx (\xi_x+\xi_y+\xi_z)\partial_\xi\left[(\nu+\tilde{\nu})(\xi_x+\xi_y+\xi_z)\partial_\xi(\tilde{\nu})\right]\\
&+ (\eta_x+\eta_y+\eta_z)\partial_\eta\left[(\nu+\tilde{\nu})(\eta_x+\eta_y+\eta_z)\partial_\eta(\tilde{\nu})\right]\\
&+ (\zeta_x+\zeta_y+\zeta_z)\partial_\zeta\left[(\nu+\tilde{\nu})(\zeta_x+\zeta_y+\zeta_z)\partial_\zeta(\tilde{\nu})\right] && (3.22)
\end{aligned}$$

$$\begin{aligned}
\nabla^2\tilde{\nu} &\approx (\xi_x+\xi_y+\xi_z)\partial_\xi\left[(\xi_x+\xi_y+\xi_z)\partial_\xi(\tilde{\nu})\right]\\
&+ (\eta_x+\eta_y+\eta_z)\partial_\eta\left[(\eta_x+\eta_y+\eta_z)\partial_\eta(\tilde{\nu})\right]\\
&+ (\zeta_x+\zeta_y+\zeta_z)\partial_\zeta\left[(\zeta_x+\zeta_y+\zeta_z)\partial_\zeta(\tilde{\nu})\right] && (3.23)
\end{aligned}$$

As one might expect, the diffusion terms above exhibit the same form as the viscous terms in the Navier-Stokes equations. Equations 3.13 through 3.15 can then be used.

Production and Destruction

The vorticity components in the transformed coordinates of the production term can be discretized using simple centered differencing:

$$\begin{aligned}\tilde{S}_1 = \frac{1}{2}[\xi_y(w_{j+1,k,m} - w_{j-1,k,m}) &+ \eta_y(w_{j,k+1,m} - w_{j,k-1,m}) \\ +\zeta_y(w_{j,k,m+1} - w_{j,k,m-1}) &- \xi_z(v_{j+1,k,m} - v_{j-1,k,m}) \\ -\eta_z(v_{j,k+1,m} - v_{j,k-1,m}) &- \zeta_z(v_{j,k,m+1} - v_{j,k,m-1})]\end{aligned} \tag{3.24}$$

$$\begin{aligned}\tilde{S}_2 = \frac{1}{2}[\xi_z(u_{j+1,k,m} - u_{j-1,k,m}) &+ \eta_z(u_{j,k+1,m} - u_{j,k-1,m}) \\ +\zeta_z(u_{j,k,m+1} - u_{j,k,m-1}) &- \xi_x(w_{j+1,k,m} - w_{j-1,k,m}) \\ -\eta_x(w_{j,k+1,m} - w_{j,k-1,m}) &- \zeta_x(w_{j,k,m+1} - w_{j,k,m-1})]\end{aligned} \tag{3.25}$$

$$\begin{aligned}\tilde{S}_3 = \frac{1}{2}[\xi_x(v_{j+1,k,m} - v_{j-1,k,m}) &+ \eta_x(v_{j,k+1,m} - v_{j,k-1,m}) \\ +\zeta_x(v_{j,k,m+1} - v_{j,k,m-1}) &- \xi_y(u_{j+1,k,m} - u_{j-1,k,m}) \\ -\eta_y(u_{j,k+1,m} - u_{j,k-1,m}) &- \zeta_y(u_{j,k,m+1} - u_{j,k,m-1})]\end{aligned} \tag{3.26}$$

3.1.4 Boundary Conditions

Solid Wall

The solid wall boundary conditions for an inviscid flow can be found in Nichols' thesis [24].

The boundary conditions for a viscous flow are explained in Section 2.4.2. Unscaled values are used here and are discretized as:

$$\rho_{j,k,m_{wall}} - \rho_{j,k,m_{wall}\pm1} = 0 \tag{3.27}$$

$$(\rho u)_{j,k,m_{wall}} = 0 \tag{3.28}$$

$$(\rho v)_{j,k,m_{wall}} = 0 \tag{3.29}$$

$$(\rho w)_{j,k,m_{wall}} = 0 \tag{3.30}$$

$$p_{j,k,m_{wall}} - p_{j,k,m_{wall}\pm1} = 0 \tag{3.31}$$

$$\tilde{\nu}_{j,k,m} = 0 \tag{3.32}$$

where the ±1 index modifier indicates that the solid surface occurs either at the top $(m-1)$ or the bottom $(m+1)$ of a block.

Inviscid Far-field Boundary Condition

As discussed in Section 2.4.1, the selection of the inviscid far-field boundary conditions is dependent on the inflow or outflow condition at that point. The first two Riemann invariants can be discretized as follows:

$$\left(V_n - \frac{2}{(\gamma-1)}\sqrt{\frac{\gamma p}{\rho}}\right)_{j,k,m_{max}} - \left(V_n - \frac{2}{(\gamma-1)}\sqrt{\frac{\gamma p}{\rho}}\right)_{\infty} = 0 \tag{3.33}$$

$$\left(V_n + \frac{2}{(\gamma-1)}\sqrt{\frac{\gamma p}{\rho}}\right)_{j,k,m_{max}} - \left(V_n + \frac{2}{(\gamma-1)}\sqrt{\frac{\gamma p}{\rho}}\right)_{j,k,m_{max}-1} = 0 \tag{3.34}$$

where V_n is the normal velocity at the block face. Riemann invariants R_3, R_4, R_5 and R_6 depend on whether the flow is entering or exiting the boundary. For an inflow boundary:

$$\left(\frac{\rho^\gamma}{p}\right)_{j,k,m_{max}} - \left(\frac{\rho^\gamma}{p}\right)_{\infty} = 0 \tag{3.35}$$

$$(V_{t1})_{j,k,m_{max}} - (V_{t1})_{\infty} = 0 \tag{3.36}$$

$$(V_{t2})_{j,k,m_{max}} - (V_{t2})_{\infty} = 0 \tag{3.37}$$

$$\tilde{\nu}_{j,k,m_{max}} - \tilde{\nu}_{\infty} = 0 \tag{3.38}$$

where V_{t1} and V_{t2} are the tangential velocities on the boundary. For an outflow boundary:

$$\left(\frac{\rho^\gamma}{p}\right)_{j,k,m_{max}} - \left(\frac{\rho^\gamma}{p}\right)_{j,k,m_{max}-1} = 0 \tag{3.39}$$

$$(V_{t1})_{j,k,m_{max}} - (V_{t1})_{j,k,m_{max}-1} = 0 \tag{3.40}$$

$$(V_{t2})_{j,k,m_{max}} - (V_{t2})_{j,k,m_{max}-1} = 0 \tag{3.41}$$

$$\tilde{\nu}_{j,k,m_{max}} - \tilde{\nu}_{j,k,m_{max}-1} = 0 \tag{3.42}$$

Viscous Far-field Boundary Condition

As discussed in Section 2.4.1, this boundary condition is called where the flow is known to be outflowing, and subject to viscous effects (such as downstream of a wing). They are as follows:

$$\left(\rho_{j,k,m}\right)_{out} - \left(\rho_{j,k,m}\right)_{out-1} = 0 \tag{3.43}$$

$$\left((\rho u)_{j,k,m}\right)_{out} - \left((\rho u)_{j,k,m}\right)_{out-1} = 0 \tag{3.44}$$

$$\left((\rho v)_{j,k,m}\right)_{out} - \left((\rho v)_{j,k,m}\right)_{out-1} = 0 \tag{3.45}$$

$$\left((\rho w)_{j,k,m}\right)_{out} - \left((\rho w)_{j,k,m}\right)_{out-1} = 0 \tag{3.46}$$

$$\left(p_{j,k,m}\right)_{out} - \left(p_{j,k,m}\right)_{out-1} = 0 \tag{3.47}$$

$$\left(\tilde{\nu}_{j,k,m}\right)_{out} = 0 \tag{3.48}$$

Symmetry

The amount of computational effort can be drastically reduced by imposing symmetry boundaries where appropriate. This type of boundary is used for cases where the normal velocity and normal derivatives in any one of the three directions on a given flat face boundary are zero. Such cases include flows around infinite airfoils, and wing and wing-body geometries where no yaw and roll angles are present. The symmetry condition consists of zeroing all flow variable gradients across the plane to ensure the solution is unchanged, and ensuring flow tangency by specifying V_n=0.

Block Interfaces

Block interfaces occur where two blocks are connected. Continuity of the solution at this boundary is ensured through a series of “ghost” or “halo” nodes. These nodes share the same physical space as interior nodes in a neighbouring block (see Figure 3.1). No equations are solved at these nodes. Instead, the equations at the interior nodes in Block 2 are solved and then copied to the halo nodes in Block 1. The number of halo nodes used is dependent on the size of the stencil required for the spatial discretization. The nodes at the interface are solved independently on each block.

3.2 Linearization and Newton's Method

After the transformation and discretization of the Navier-Stokes equations, the system of equations takes the form:

$$\frac{d\hat{Q}}{d\tau} + R\left(\hat{Q}\right) = 0 \tag{3.49}$$

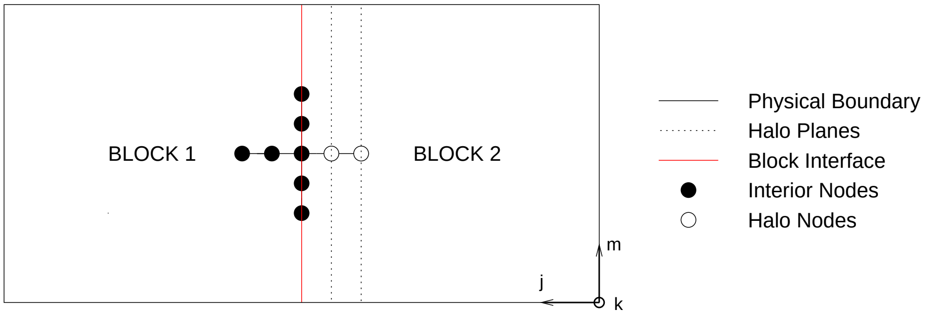


Figure 3.1: Halo nodes on a two-dimensional slice of two three-dimensional blocks for a 13-point stencil

where R is the set of discretized equations, and $\hat{Q}$ is now the set of discretized flow variables. Note that discretization of the spatial terms has turned the original PDEs into a set of coupled ODEs.

If one is interested only in steady-state solutions, then the equations simplify to:

$$R\left(\hat{Q}\right) = 0 \tag{3.50}$$

In addition, if we are interested in turbulent flows, then $\hat{Q}$ includes the turbulent eddy term $\tilde{\nu}$:

$$\hat{Q}_{j,k,m} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e \\ \tilde{\nu} \end{bmatrix}_{j,k,m} \tag{3.51}$$

and the turbulence model equation is added to $R(\hat{Q})$.

In order to solve the above equations using a quasi-Newton approach, this set of nonlinear equations needs to be linearized. This is done by first performing a Taylor series expansion of R about a solution state $Q^{(n)}$, and then dropping higher-order terms in the expansion:

$$R\left(\hat{Q}^{(n+1)}\right) \approx R\left(\hat{Q}^{(n)}\right) + \mathcal{A}^{(n)}\Delta\hat{Q}^{(n)} = 0 \tag{3.52}$$

where

$$\Delta\hat{Q}^{(n)} = \hat{Q}^{(n+1)} - \hat{Q}^{(n)} \tag{3.53}$$

and the Jacobian matrix of R (also called the flow Jacobian) is given by:

$$\mathcal{A}^{(n)} = \left(\frac{\partial R}{\partial \hat{Q}}\right)^{(n)} \tag{3.54}$$

Equation 3.52 can be rewritten as:

$$\mathcal{A}^{(n)} \Delta \hat{Q}^{(n)} = -R\left(\hat{Q}^{(n)}\right) \tag{3.55}$$

The solution to the system yields ΔQ, which is used to update the flow variables at the next step:

$$\hat{Q}^{(n+1)} = \hat{Q}^{(n)} + \Delta \hat{Q}^{(n)} \tag{3.56}$$

Newton's method is desirable due to its quadratic rate of convergence. However, this is rarely achieved during startup due to the fact that Newton's method is only a good approximation of $R^{(n+1)}$ for small ΔQ close to the final solution. It is much more practical to use an implicit Euler time-stepping method, which combines stability and speed:

$$\left[\frac{I}{\Delta t} + \mathcal{A}\right]^{(n)} \Delta \hat{Q}^{(n)} = -R^{(n)} \tag{3.57}$$

where Δt is the time-step and I is an identity matrix.

Note that a fixed time-step in an implicit Euler formulation yields only a linear convergence rate. However, Newton's method can be recovered as the time-step is iteratively increased such that $\Delta t \rightarrow \infty$ near convergence.

In the above equations, $\mathcal{A}$ is an exact linearization of the fluxes and turbulence model. A first-order approximate Jacobian, $\mathcal{A}_1$ can also be used to solve the same system of equations. This is beneficial since instead of storing information from thirteen blocks at each node, only the nearest neighbours will be saved (seven in total). Called the approximate-Newton method, this approach can improve speed by being more diagonally dominant and reduce memory requirements via a reduction in the size of the stencil. The fourth and second-difference dissipation coefficients are combined using:

$$d_p^{(2)} = d^{(2)} + \sigma d^{(4)} \tag{3.58}$$

where $d_p^{(2)}$ is the combined dissipation term. The parameter σ will be determined through a numerical study in Chapter 5.

3.2.1 Linearization of the Interior Scheme

Inviscid Flux Jacobian

For the linearization of the inviscid interior scheme, see Nichols [24].

Viscous Flux Jacobian

The cross-derivative terms are removed when forming the Jacobian matrix. Without these terms, the viscous Jacobian is given by:

$$\partial_{\hat{Q}}\left(\beta\right) = J^{-1}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ m_{21} & \alpha_1\partial_\varsigma(\rho^{-1}) & \alpha_2\partial_\varsigma(\rho^{-1}) & \alpha_3\partial_\varsigma(\rho^{-1}) & 0 \\ m_{31} & \alpha_2\partial_\varsigma(\rho^{-1}) & \alpha_4\partial_\varsigma(\rho^{-1}) & \alpha_5\partial_\varsigma(\rho^{-1}) & 0 \\ m_{41} & \alpha_3\partial_\varsigma(\rho^{-1}) & \alpha_5\partial_\varsigma(\rho^{-1}) & \alpha_6\partial_\varsigma(\rho^{-1}) & 0 \\ m_{51} & m_{52} & m_{53} & m_{54} & m_{55} \end{bmatrix} J \qquad (3.59)$$

with

$$\begin{aligned}
m_{21} &= -\alpha_1\partial_\varsigma(u/\rho) - \alpha_2\partial_\varsigma(v/\rho) - \alpha_3\partial_\varsigma(w/\rho) \\
m_{31} &= -\alpha_2\partial_\varsigma(u/\rho) - \alpha_4\partial_\varsigma(v/\rho) - \alpha_5\partial_\varsigma(w/\rho) \\
m_{41} &= -\alpha_3\partial_\varsigma(u/\rho) - \alpha_5\partial_\varsigma(v/\rho) - \alpha_6\partial_\varsigma(w/\rho) \\
m_{51} &= \alpha_7\partial_\varsigma[-(e/\rho^2) + (u^2+v^2+w^2)/\rho] \\
&\quad -\alpha_1\partial_\varsigma(u^2/\rho) - \alpha_4\partial_\varsigma(v^2/\rho) - \alpha_6\partial_\varsigma(w^2/\rho) \\
&\quad -2\alpha_2\partial_\varsigma(uv/\rho) - 2\alpha_3\partial_\varsigma(uw/\rho) - 2\alpha_5\partial_\varsigma(vw/\rho) \\
m_{52} &= -\alpha_7\partial_\varsigma(u/\rho) - m_{21} \\
m_{53} &= -\alpha_7\partial_\varsigma(v/\rho) - m_{31} \\
m_{54} &= -\alpha_7\partial_\varsigma(w/\rho) - m_{41} \\
m_{55} &= \alpha_7\partial_\varsigma(\rho^{-1}) \\
\alpha_1 &= (\mu+\mu_t)\left[(4/3)\varsigma_x^2 + \varsigma_y^2 + \varsigma_z^2\right] \\
\alpha_2 &= \frac{1}{3}(\mu+\mu_t)\varsigma_x\varsigma_y \\
\alpha_3 &= \frac{1}{3}(\mu+\mu_t)\varsigma_x\varsigma_z \\
\alpha_4 &= (\mu+\mu_t)\left[\varsigma_x^2 + (4/3)\varsigma_y^2 + \varsigma_z^2\right] \\
\alpha_5 &= \frac{1}{3}(\mu+\mu_t)\varsigma_y\varsigma_z \\
\alpha_6 &= (\mu+\mu_t)\left[\varsigma_x^2 + \varsigma_y^2 + (4/3)\varsigma_z^2\right] \\
\alpha_7 &= \gamma\left(\mu\mathcal{P}r^{-1} + \mu_t\mathcal{P}r_t^{-1}\right)\left(\varsigma_x^2 + \varsigma_y^2 + \varsigma_z^2\right)
\end{aligned}$$

where $\beta = E_v$ for $\varsigma = \xi$, $\beta = F_v$ for $\varsigma = \eta$ and $\beta = G_v$ for $\varsigma = \zeta$.

Linearization of the Turbulence Model

The turbulence model is included into the flow Jacobian in the following manner:

$$\mathcal{A}_{turb_{j,k,m}} = \begin{bmatrix} a_{11} & \cdots & \cdots & \cdots & \cdots & a_{16} \\ \vdots & \ddots & & & & a_{26} \\ \vdots & & \ddots & & & a_{36} \\ \vdots & & & \ddots & & a_{46} \\ \vdots & & & & \ddots & a_{56} \\ a_{61} & a_{62} & a_{63} & a_{64} & a_{65} & a_{66} \end{bmatrix}_{j,k,m} \tag{3.60}$$

where the entries in the 5×5 block in the upper left corner of the matrix correspond to contributions from the Navier-Stokes equations, and the sixth row and column are from the turbulence model.

Note that entries a_{16} through a_{56} are zero, while the off-diagonal terms a_{61} through a_{65} result from the vorticity terms in the Spalart-Allmaras equation.

The differentiation of the turbulence model can be done without much difficulty, with all the terms corresponding to the model in two dimensions. The differentiation of the vorticity term can be found in Appendix B.

3.2.2 Linearization of the Boundary Conditions

Boundary conditions are treated implicitly in order for the code to converge quadratically. The linearization of the viscous boundary conditions in three dimensions is completely analogous to those which are done in two dimensions by Pueyo [27]. The boundary conditions from Section 3.1.4 can be presented in the following form:

$$\mathcal{B}(\mathcal{R}) = 0 \tag{3.61}$$

where $\mathcal{R}$ is the set of flow variables:

$$\mathcal{R} = \begin{bmatrix} \rho \\ u \\ v \\ w \\ p \\ \tilde{\nu} \end{bmatrix} \tag{3.62}$$

Newton's method as applied to this set of equations yields:

$$P\Delta\mathcal{R} = -\mathcal{B}^{(n)} \tag{3.63}$$

where

$$P = \left(\frac{\partial\mathcal{B}}{\partial\mathcal{R}}\right)^{(n)} \tag{3.64}$$

The selection of the flow variables in $\mathcal{R}$ makes necessary a transformation of the update $\Delta\mathcal{R}$ back into the working flow variables $\hat{Q}$. A transformation matrix $M = \frac{\partial B}{\partial\mathcal{R}}$, is thus defined:

$$\Delta\hat{Q} = M\Delta\mathcal{R} \tag{3.65}$$

where M is given as:

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ u & \rho & 0 & 0 & 0 & 0 \\ v & 0 & \rho & 0 & 0 & 0 \\ w & 0 & 0 & \rho & 0 & 0 \\ \frac{u^2+v^2+w^2}{2} & \rho u & \rho v & \rho w & \frac{1}{\gamma-1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.66}$$

Equation3.64 can then be written as:

$$P\underbrace{M^{-1}J\Delta\hat{Q}}_{\Delta\mathcal{R}} = -\mathcal{B}^{(n)} \tag{3.67}$$

where

$$M^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -\frac{u}{\rho} & \frac{1}{\rho} & 0 & 0 & 0 & 0 \\ -\frac{v}{\rho} & 0 & \frac{1}{\rho} & 0 & 0 & 0 \\ -\frac{w}{\rho} & 0 & 0 & \frac{1}{\rho} & 0 & 0 \\ (\gamma-1)\frac{u^2+v^2+w^2}{2} & -(\gamma-1)u & -(\gamma-1)v & -(\gamma-1)w & \gamma-1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.68}$$

Solid Boundary

For the linearization of the inviscid boundary conditions, see Nichols [24]. The linearization at the solid-wall boundary condition for a viscous flow can be written as:

$$\left[PM^{-1}J\Delta\hat{Q}\right]_{j,k,m_{min/max}} - \left[PM^{-1}J\Delta\hat{Q}\right]_{j,k,m_{max\pm1}} = -\mathcal{B}^{(n)} \tag{3.69}$$

The Jacobians at the surface of the body and one-off the surface for a no-slip condition are:

$$\left(\frac{\partial\mathcal{B}}{\partial\mathcal{R}}\right)_{j,k,m_{wall}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ u & \rho & 0 & 0 & 0 & 0 \\ v & 0 & \rho & 0 & 0 & 0 \\ w & 0 & 0 & \rho & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{j,k,m_{wall}} \tag{3.70}$$

$$\left(\frac{\partial\mathcal{B}}{\partial\mathcal{R}}\right)_{j,k,m_{wall\pm1}} = - \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}_{j,k,m_{wall\pm1}} \tag{3.71}$$

Far-field Boundary

For the inviscid inflow far-field boundary conditions, refer to Nichols [24]. Linearization of the viscous outflow far-field boundaries can be represented in the same general form as that of the viscous solid boundary in Equation 3.69, where instead, the Jacobians are:

$$\left(\frac{\partial\mathcal{B}}{\partial\mathcal{R}}\right)_{j,k,m_{min/max}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ u & \rho & 0 & 0 & 0 & 0 \\ v & 0 & \rho & 0 & 0 & 0 \\ w & 0 & 0 & \rho & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{j,k,m_{out}} \tag{3.72}$$

$$\left(\frac{\partial\mathcal{B}}{\partial\mathcal{R}}\right)_{j,k,m_{min+1/max-1}} = - \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ u & \rho & 0 & 0 & 0 & 0 \\ v & 0 & \rho & 0 & 0 & 0 \\ w & 0 & 0 & \rho & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{j,k,m_{out-1}} \tag{3.73}$$

3.3 Solution to the Linear Problem

From the spatial discretization and the application of Newton's method, there now arises a linear system of equations. It is desirable to solve these equations in an efficient manner.

An obvious method would be to directly solve the system. But due to the size of the linear system, this may be very expensive. A more cost-effective solution would be to use an iterative solver such as Generalized Minimal Residual (GMRES), BiConjugate Gradients or BiConjugate Gradient Stabilized. GMRES approximates the solution to the linear system by building a solution in a Krylov subspace using a minimal residual.

3.3.1 Jacobian-free GMRES

A highly desirable property of GMRES is that it does not require the explicit formation of the Jacobian matrix. Instead, only a matrix-vector product is used. The matrix-vector product itself can be approximated using a first-order Fréchet derivative of the form:

$$\mathcal{A} \cdot v \approx \frac{R(\hat{Q} + \epsilon v) - R(\hat{Q})}{\epsilon} \tag{3.74}$$

where ϵ is a small scalar perturbation parameter tuned to provide an accurate approximation of the matrix-vector product. The size of this parameter is determined following the work done by Nielsen et al. in [26] and is given by:

$$\epsilon \simeq \frac{\sqrt{\epsilon_m}}{\bar{v}} \tag{3.75}$$

where ϵ_m is the value of machine zero for the hardware used in the computation and $\bar{v}$ is the RMS value of v.

3.3.2 Inexact-Newton Method

The exact solution to the system is expensive to compute due to the size of the matrix. The solution to the linear system at each Newton iteration can be solved up to a certain accuracy, controlled by convergence parameter η, in the following manner:

$$\|R^{(n)} + \mathcal{A}^{(n)} \Delta \hat{Q}^{(n)}\| \leq \eta^{(n)} \|R^{(n)}\| \tag{3.76}$$

Note that as $\eta^{(n)} \to 0$ for all n, Newton's method is recovered, and the system is solved exactly. A well chosen η value will maintain a good balance between under and over-solving the system. This value is determined through a numerical study documented in Chapter 5.

3.3.3 Preconditioner

The flow matrix may become numerically stiff when a large spread in the eigenvalues is encountered. This decreases the efficiency of the GMRES solver as the equations become more difficult to solve. The purpose of a preconditioner is then to create a system where the eigenvalue spectrum is clustered about one. The preconditioning matrix can be applied to the left or the right of the flow matrix, but right preconditioning is chosen since the residual is left unchanged. This is preferred since the residual is required for checking the convergence criterion at each iteration.

Thus, the preconditioned system is written as follows:

$$\left[\frac{I}{\Delta t} + \mathcal{A}\right] \mathcal{M}^{-1}\mathcal{M}\Delta\hat{Q} = -R \tag{3.77}$$

where $\mathcal{M}$ is the preconditioning matrix. Note that the preconditioned system $\mathcal{A}\mathcal{M}^{-1}$ should be much better conditioned than $\mathcal{A}$, and $\mathcal{M}^{-1}$ should also closely approximate $\mathcal{A}^{-1}$ while being much more efficient to compute. The preconditioner is formed using an incomplete lower-upper factorization based on the approximate flow Jacobian $\mathcal{A}_1$.

3.3.4 Incomplete LU Factorization

The preconditioner is formed using an incomplete lower-upper factorization. This method generates two sparse matrices: a lower triangular matrix $\mathcal{L}$ and an upper triangular matrix $\mathcal{U}$:

$$\mathcal{A}_1 = \mathcal{L} \cdot \mathcal{U} \approx \tilde{\mathcal{L}} \cdot \tilde{\mathcal{U}} = \mathcal{M} \tag{3.78}$$

where $\mathcal{L}$ and $\mathcal{U}$ are the exact incomplete LU factorization of $\mathcal{A}_1$ while $\tilde{\mathcal{L}}$ and $\tilde{\mathcal{U}}$ are approximate factorizations.

The fill factor k allows the user to select the accuracy of the $\mathcal{M}$ matrix relative to the approximate flow Jacobian. The higher the integer value of k, the more representative the matrix is of $\mathcal{A}_1$. This parameter will be determined experimentally in Chapter 5.

The construction of the approximate factors $\tilde{\mathcal{L}}$ and $\tilde{\mathcal{U}}$ must be done in such a way that:

$$\tilde{\mathcal{U}}^{-1} \cdot \tilde{\mathcal{L}}^{-1} = \mathcal{M}^{-1} \approx \mathcal{A}_1^{-1} \tag{3.79}$$

while being much cheaper to compute than a direct inversion of $\mathcal{A}_1$.

3.3.5 Reverse Cuthill-McKee Nodal Reordering

The natural ordering of the terms in the flow Jacobian produces a sparse matrix with a large bandwidth. The goal of the Reverse Cuthill-McKee method [5] is to reorder the equations in a way such that the matrix entries are clustered near the diagonal, resulting in a reduction in the bandwidth. This is important for decreasing the storage requirements and improving the effectiveness of the ILU factorization algorithm [27].

3.4 Algorithm Startup

Two methods are used to converge the solution in TYPHOON: an approximate-Newton method which uses a first-order approximate Jacobian, and a Jacobian-free method. The former is used during the startup phase of the convergence, while the latter is used to converge the code to machine zero.

Startup is the most important phase of the flow solve. This is because the initial flow field is far from the converged solution, and the Newton method works well only near the region of the solution. A modified Newton method (Implicit Euler) is used to obtain an intermediate solution from which the Jacobian-free inexact-Newton method can proceed.

The use of an approximate Jacobian along with the inexact solution of the linear system of equations at each non-linear iteration results in the approximate-Newton method. This is used as a startup for the flow solver. Although slower than the Jacobian-free inexact-Newton method, it is far more stable during the initial phases of convergence.

The switch from the approximate-Newton method to the Jacobian-free inexact-Newton method is determined through the parameter $R_{d_{tol}}$, which is found by conducting a numerical study. The results of this study are shown in Chapter 5.

3.4.1 Local Time-Stepping

A local time-stepping method is used in TYPHOON. The time-step, Δt as first shown in Equation 3.57, is added to the preconditioner to strengthen its diagonal, and therefore, stabilize and accelerate convergence.

Mean Flow Time-step

The pseudo-time term for the mean flow equations during the approximate-Newton phase is determined locally according to work done by Pulliam in [29], and is given by:

$$\Delta t^{(n)}_{j,k,m} = \frac{\Delta t^{(n)}_{ref}}{1 + \sqrt{J_{j,k,m}}} \tag{3.80}$$

where $J_{j,k,m}$ is the local metric Jacobian, and Δt_{ref} is a global reference time-step that is increased as the solution develops.

The reference time-step is based on a geometric formula used in HURRICANE [14, 15] and is given as follows:

$$\Delta t^{(0)}_{ref} = A, \qquad \Delta t^{(n)}_{ref} = B \cdot \Delta t^{(n-1)}_{ref} \tag{3.81}$$

For the laminar grids used, $A = 0.25$ and $B = 1.125$ while the values for turbulent grids, $A = 1.00$ and $B = 1.300$. For inviscid grids, which typically have smaller aspect ratios, values as reported by Nichols are $A = 1.00$ and $B = 1.7$ [24].

Once an intermediate solution to the flow has been obtained, the Jacobian-free inexact-Newton method is employed. The Switched Evolution Relaxation method developed by Mulder and van Leer [21] is used and is given by the following:

$$\Delta t^{(n)}_{ref} = \max\left[\frac{\alpha}{\left(R^{(n)}_d\right)^{\beta}},\ \Delta t_{min}\right] \tag{3.82}$$

where α and β are constants typically set to 1.0 and 1.3 repectively, and Δt_{min} is the minimum time-step allowed whose value is 50.

The parameter dictating which solution method to use is the relative residual defined by:

$$R^{(n)}_d = \frac{||R^{(n)}||}{||R^{(0)}||} \tag{3.83}$$

where $R^{(n)}$ is the residual at outer iteration n, and $R^{(0)}$ is the starting residual.

Turbulent Time-step

Following the work done in two-dimensions by Chisholm [3] and in three-dimensions by Wong [37], the turbulent time step is determined by:

$$\Delta t^{(n)}_{turb} = \begin{cases} \Delta t^{(n)}_{j,k,m} & \text{if } |\delta_{j,k,m}| < \delta_{max_{j,k,m}} \\ \\ |\Delta t_{lim}| & \text{otherwise} \end{cases} \tag{3.84}$$

where the estimate of the local update variable $\delta_{j,k,m}$ is calculated using:

$$\delta_{j,k,m} = -\frac{[R(6)]_{j,k,m}}{[\mathcal{A}_1(6,6)]_{j,k,m}} \tag{3.85}$$

and where $R\left[(6)\right]_{j,k,m}$ is the turbulent variable at each given node in the residual equation, and $[\mathcal{A}_1(6,6)]_{j,k,m}$ is the sixth diagonal component in the 6×6 block of the Jacobian matrix $\mathcal{A}_1$ at node j,k,m.

The maximum update variable $\delta_{max_{j,k,m}}$ is calculated locally using:

$$\delta_{max_{j,k,m}} = r \cdot \max(\tilde{\nu}_{j,k,m}, 1.0) \tag{3.86}$$

where r is a limiting variable used to minimize the update. This value is typically 0.3. The limiting time-step Δt_{lim} is determined by solving:

$$\left(\frac{J_{j,k,m}}{\Delta t_{lim_{j,k,m}}} + [\mathcal{A}_1(6,6)]_{j,k,m} \right) \delta_{j,k,m_{max}} = -\left[R(6)\right]_{j,k,m} \tag{3.87}$$

At each Newton iteration, the turbulence variable is checked for negative values and clipped in the following manner:

$$\tilde{\nu} = \begin{cases} 1 \times 10^{-14} & \text{if on a solid wall} \\ \\ \tilde{\nu}_{\infty} & \text{otherwise} \end{cases} \tag{3.88}$$

where $\tilde{\nu}_1$ is the freestream eddy viscosity parameter, typically set at 0.001. This clipping method has been found by Chisholm [3] and Wong [37] to improve the stability of the solver.

Chapter 4

Grid Generation

The manner in which meshes are generated plays a critical role in the convergence of the flow solution. The grid generation tool that was used was ANSYS Inc's ICEM CFD v.11.

4.1 Blocking Strategy

The current blocking strategy is unchanged from that presented by Nichols in [24]. The arrangement of the twelve blocks is shown in Figure 4.1. The surface mesh consists of clustered edges and unsmoothed nodes (see Figure 4.2).

4.2 Problem Definition

The intersection of the eight blocks at the finely clustered nodes of the wingtips poses a major challenge to the grid generator. The problem manifests itself as skewing or wrinkling when fine clustering is needed (Figure 4.2), for example during laminar or turbulent mesh generation. Skewing or wrinkling does not necessarily cause the metric Jacobian to become inverted, although this is often the case. Only careful inspection of the pre-mesh in the GUI can determine this.

The inability to finely cluster nodes at the wing tip results in greatly stretched cells near the wingtip and leading edge intersection. This in turn affects the convergence of the solver, in particular, during startup.

The determinant in ICEM is the main indicator of mesh quality. Although it is similar to the metric Jacobian in TYPHOON, the values can differ greatly.

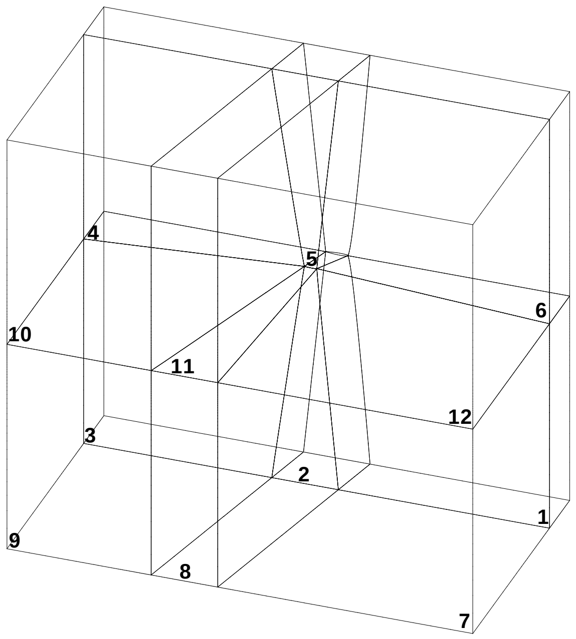


Figure 4.1: TYPHOON's 12-block configuration

The algorithm for calculation of the determinant in ICEM involves the computation of all 27 metric Jacobians in a 27-node hexahedron at each cell volume. From this, the maximum value is reported. In contrast, the metric Jacobian in TYPHOON is calculated at each grid node. Thus, ICEM may report negative determinants indicating inverted cells when the mesh may actually be usable. The reverse can also occur, where non-negative determinants show up as inverted nodes in TYPHOON.

4.3 Smoothing

ICEM employs various grid smoothing mechanisms for structured grids. The multiblock method performs elliptical smoothing and is specially optimized for use in blade configurations. Indeed, use of the smoother on a multiblock grid with a single element airfoil produces pinching at the trailing edge which results in poor flow solutions (Figure 4.4).

Orthogonality smoothing in ICEM is another elliptical smoother and produces better

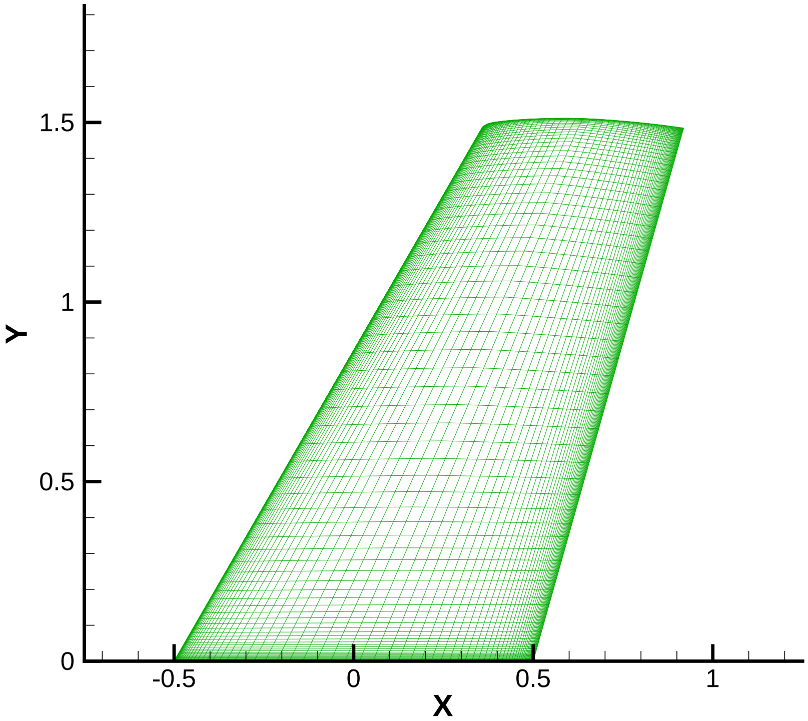
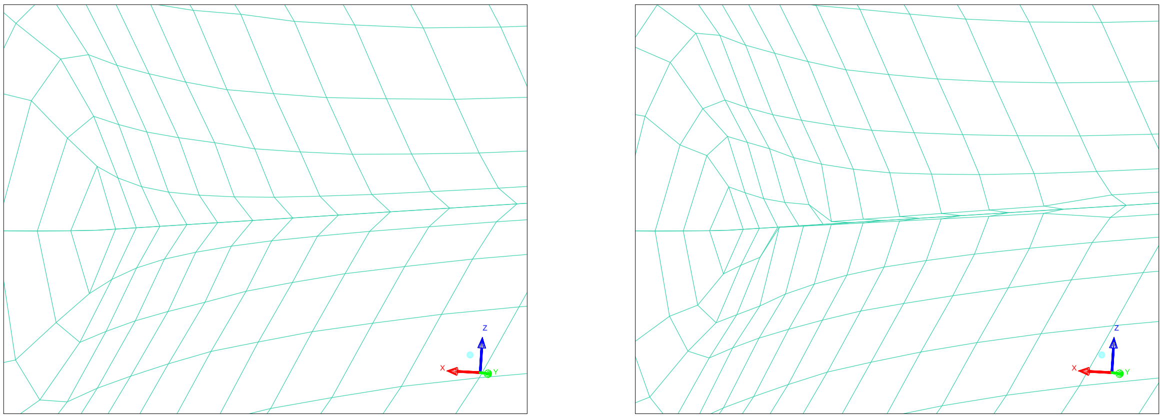


Figure 4.2: Surface mesh on an ONERA M6 wing

results (see Figure 4.4). This method is generally desirable since the spatial discretization is the most accurate on orthogonal grids, and the transformation of the equations into curvilinear coordinates produces the fewest terms [6]. The surface grid must first be frozen to aid the stability of the smoother. The stabilization factor should be chosen as low as possible to improve orthogonality of the off-wall nodes. The number of iterations should also be minimized since the algorithm will inevitably attempt to improve the mesh near the wall surface at the expense of the viscous off-wall spacing.

Parameters for orthogonal smoothing are shown in Table 4.1. In addition, wing faces should be frozen and the “define edges” option should be selected so that the node distribution on the edges on the wing is also prevented from changing.



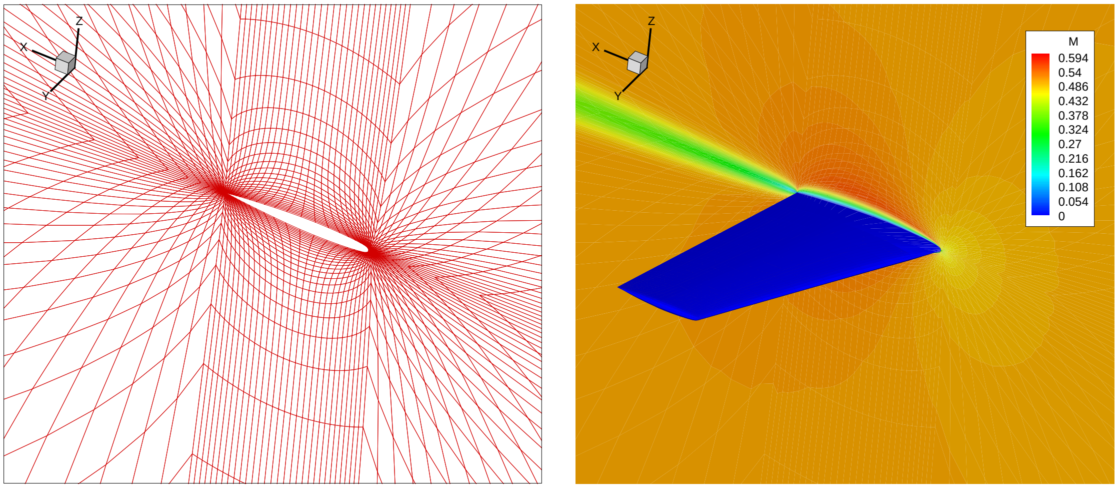
(a) uninverted - $1.3\times10^{-3}c$

(b) inverted - $1.0\times10^{-3}c$

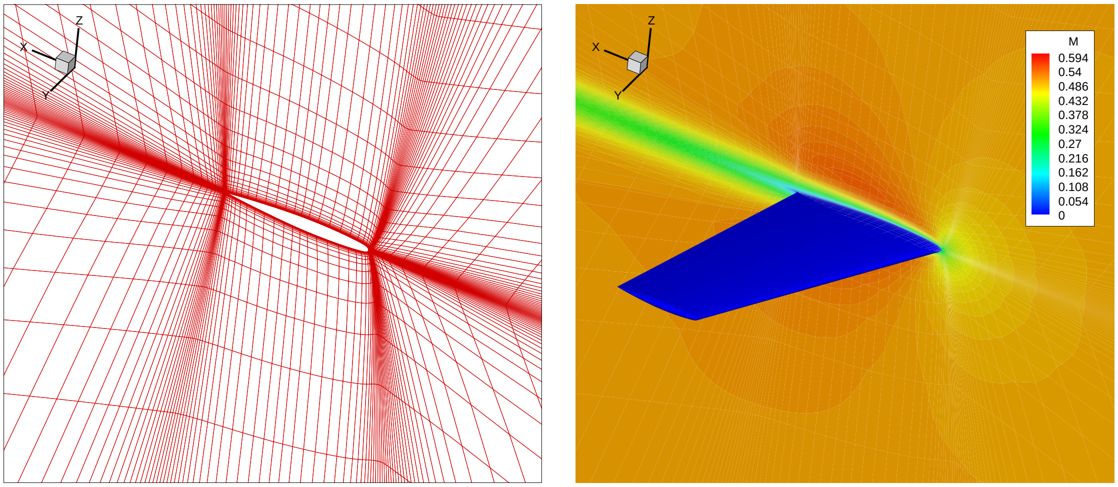
Figure 4.3: Mesh inversion at the wingtip and leading edge intersection due to changes in wingtip spacing

Parameter	Surface Value	Volume Value
Iterations	1	10
Grid Expansion	0.1	2
Stabilize Factor	1	2
Use Ortho. Dist.	N/A	N/A
Smooth Type	Laplace	Laplace

Table 4.1: Orthogonal smoothing parameters in ICEMCFD



(a) Grid 1 (b) Mach contours



(c) Grid 2 (d) Mach contours

Figure 4.4: Mesh smoothed using multiblock method (a) and resulting flow solution (b); Mesh smoothed using orthogonality method (c) and resulting flow solution (d).

Chapter 5

Algorithm Optimization

A word of caution to begin this chapter: the convergence rates were found to be quite dependent on the grid. The parameters that are optimized in this section were found to be optimal for the grids that the author was able to generate. From experience, it has been shown that grids with average aspect ratios closer to unity converge more rapidly.

The parameters here are optimized under the assumption that each is independent of the others. This is an idealization - determination of a true optimal set will be an expensive iterative process that has been forgone here.

5.1 Test Cases

A summary of the flow conditions used in the parametric study is shown in Table 5.1. Flow conditions in test cases 1 and 2 are laminar while those in cases 3 and 4 are turbulent. A different grid was used on each of the four flow solves. A multiblock configuration of twelve blocks about an ONERA M6 wing is used with the parameters prescribed in Table 5.2.

Case Number	Mach number	Reynolds Number	Angle of Attack (°)
1	0.5	600	3.0
2	0.84	600	3.0
3	0.5	2.88×10^{6}	3.0
4	0.84	2.88×10^{6}	3.0

Table 5.1: ONERA M6 wing test cases

Grid	Grid Size	Chords to Far Field	Off-wall Spacing $10^{-5}c$	LE Cluster $10^{-3}c$	TE Cluster $10^{-3}c$	Root Cluster $10^{-3}c$	Tip Cluster $10^{-3}c$	Nodes on Wing Surface
A	173,604	10	1	0.05	0.3	50	1.3	1568
B	173,604	10	1	0.5	0.1	2	1.3	1568
C	176,400	10	0.5	1.0	1.0	1	1.3	2310
D	236,716	10	0.5	2.5	10.0	2.5	1.3	3362

Table 5.2: Grids used in the ONERA M6 wing parametric study

Symbol	Parameter
k	ILU fill level
σ	Preconditioning parameter
η	Inexact Newton convergence parameter
$R_{d_{tol}}$	Approximate-Newton convergence parameter

Table 5.3: List of parameters in the numerical study

5.2 Parametric Study

A numerical study of the four parameters shown in Table 5.3 was conducted.

5.2.1 ILU Fill Level (k)

The ILU fill level (k) determines the number of non-zero elements in the matrix to keep during the factorization. The higher the level, the more accurate the representation of the original matrix. Optimality for this parameter is defined as the value which minimizes the convergence time to a given residual reduction factor (two orders for the approximate-Newton phase and eight orders for the Jacobian-free phase).

The results for the laminar flows are shown in Figures 5.1 through 5.4. The optimal fill level for both the approximate-Newton startup phase, and the Jacobian-free convergence was found to be one. Although fill level zero may be faster in certain cases than fill level one (Figure 5.3), it may have difficulty converging, or may require a reduction in time-step. Note that the cases for ILU(0) and ILU(3) did not converge due to the development of negative pressures at the trailing edge wingtip (see Figure 5.1).

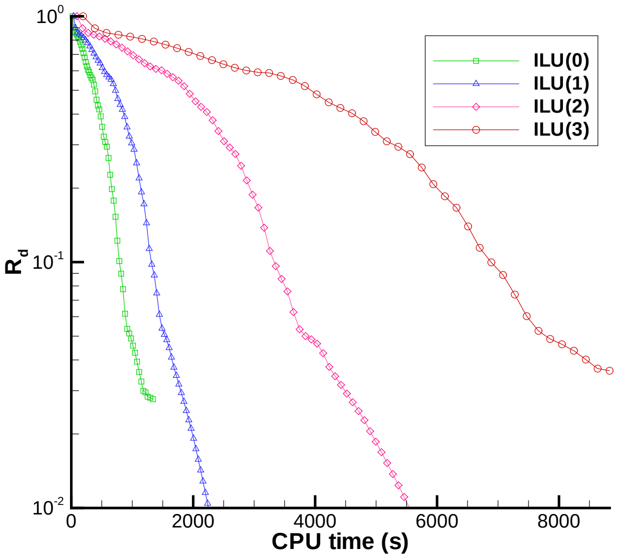


Figure 5.1: Laminar subsonic ILU fill level parameter (k) optimization; approximate-Newton phase (M=0.5, $\alpha = 3.0°$, Re=600, grid A)

Figures 5.5 through 5.8 show that for turbulent flows, an ILU fill level of one provides the optimal CPU time to convergence. Fill level zero has difficulty converging for the transonic case. Note the effect of negative $\tilde{\nu}$ clipping on the residual in Figures 5.6 and 5.8. The clipping performs well in stabilizing the turbulence model, preventing the flow solution from diverging. Negative values typically occur at the points near the wing tip and the trailing edge of the airfoil.

5.2.2 Preconditioning Parameter (σ)

The preconditioning parameter σ combines the second and fourth-difference dissipation terms in the preconditioner, as discussed in Section 3.2. In the laminar regime, these were are found to be optimal at values of 4.0 for transonic flows, and 6.0 for subsonic flows (see Figures 5.9 and 5.10). Note that the flow solver had difficulty converging the flow for values below six in the approximate-Newton phase of the subsonic flow solves. The best turbulent values were found to be 3.0 for both subsonic and transonic flows (see Figures 5.11 and 5.12).

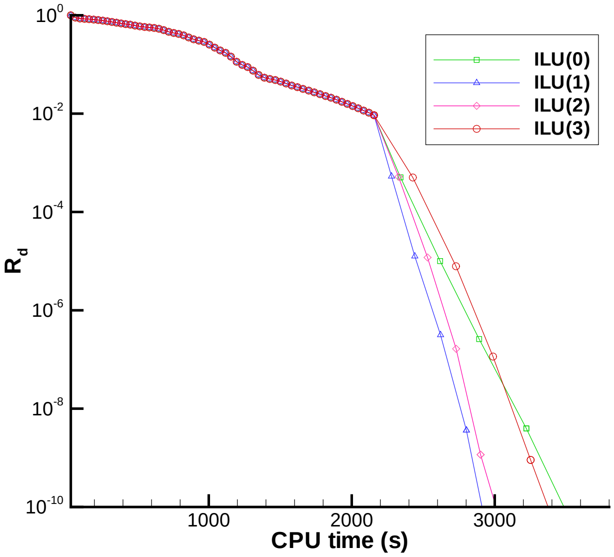


Figure 5.2: Laminar subsonic ILU fill level parameter (k) optimization; Jacobian-free phase (M=0.5, $\alpha = 3.0°$, Re=600, grid A)

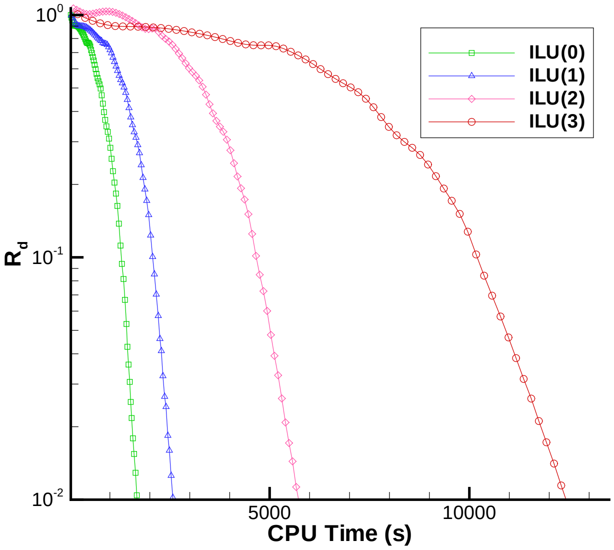


Figure 5.3: Laminar transonic ILU fill level parameter (k) optimization; approximate-Newton phase (M=0.84, $\alpha = 3.0°$, Re=600, grid B)

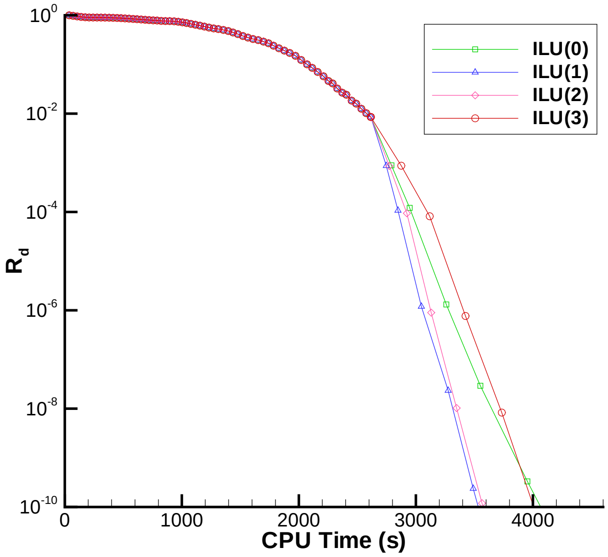


Figure 5.4: Laminar transonic ILU fill level parameter (k) optimization; Jacobian-free phase (M=0.84, $\alpha = 3.0°$, Re=600, grid B)

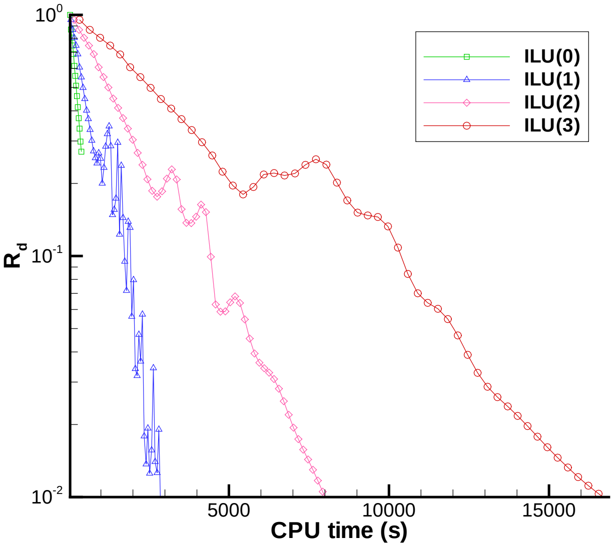


Figure 5.5: Turbulent subsonic ILU fill level parameter (k) optimization; approximate-Newton phase (M=0.5, $\alpha = 3.0°$, Re=2.88×10^{6}, grid C)

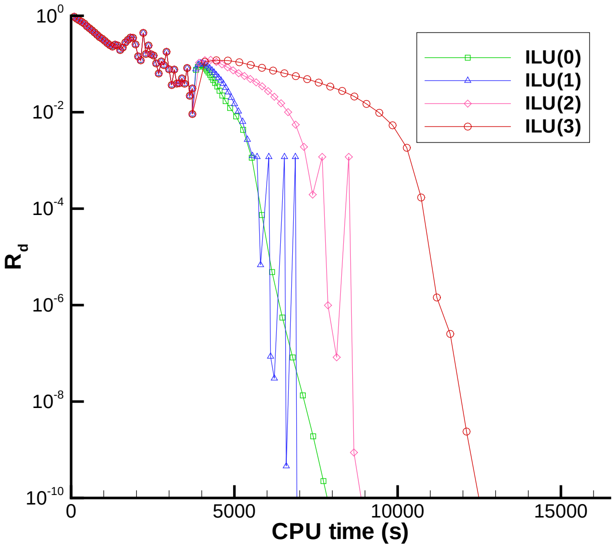


Figure 5.6: Turbulent subsonic ILU fill level parameter (k) optimization; Jacobian-free phase (M=0.5, $\alpha = 3.0°$, Re=2.88×10^6, grid D)

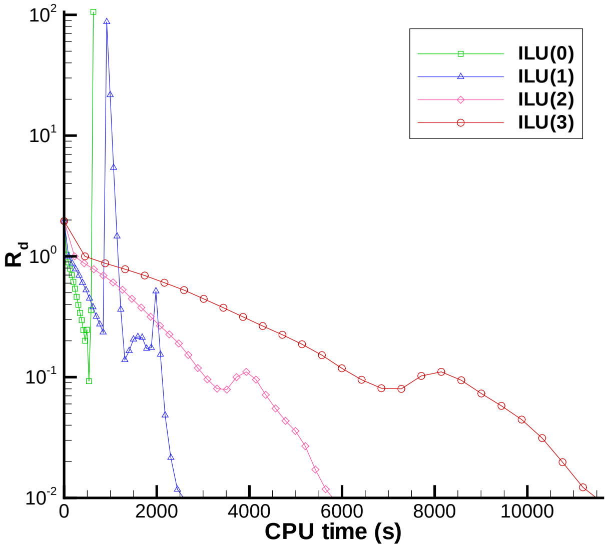


Figure 5.7: Turbulent transonic ILU fill level parameter (k) optimization; approximate-Newton phase (M=0.84, $\alpha = 3.0°$, Re=2.88×10^6, grid C)

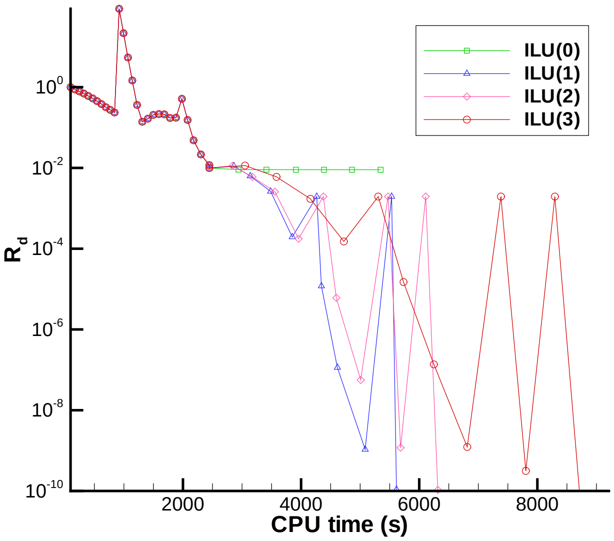


Figure 5.8: Turbulent transonic ILU fill level parameter (k) optimization; Jacobian-free phase (M=0.84, $\alpha = 3.0°$, Re=2.88×10^6, grid D)

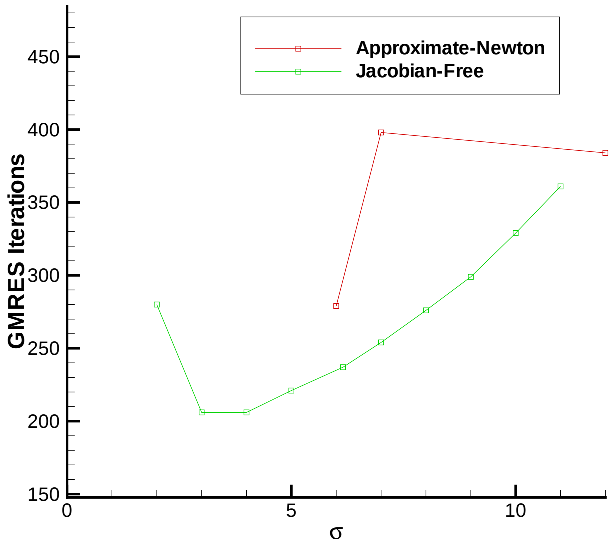


Figure 5.9: Laminar subsonic preconditioning parameter (σ) optimization (M=0.5, $\alpha =$ 3.0°, Re=600, grid A)

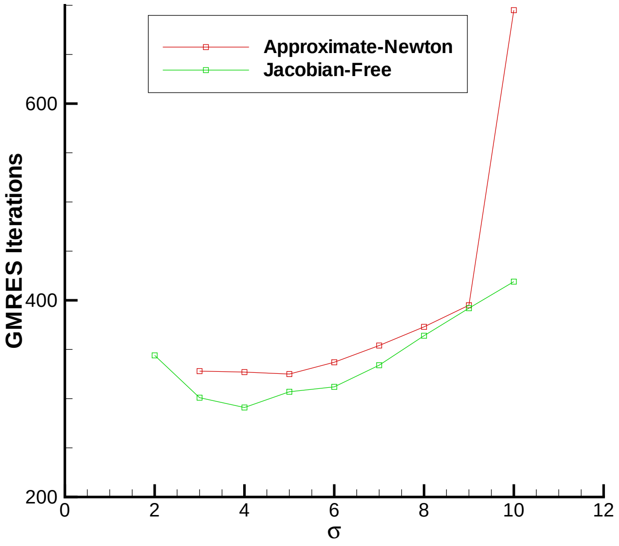


Figure 5.10: Laminar transonic preconditioning parameter (σ) optimization (M=0.84, α = 3.0°, Re=600, grid B)

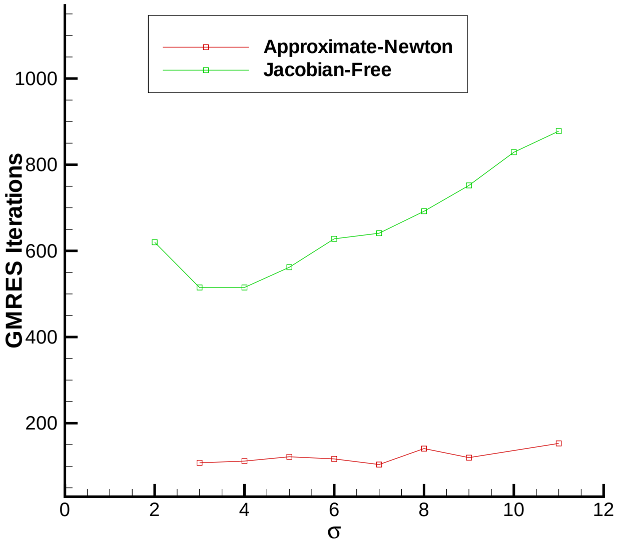


Figure 5.11: Turbulent subsonic preconditioning parameter (σ) optimization (M=0.5, α = 3.0°, Re=2.88×10^6, grid C)

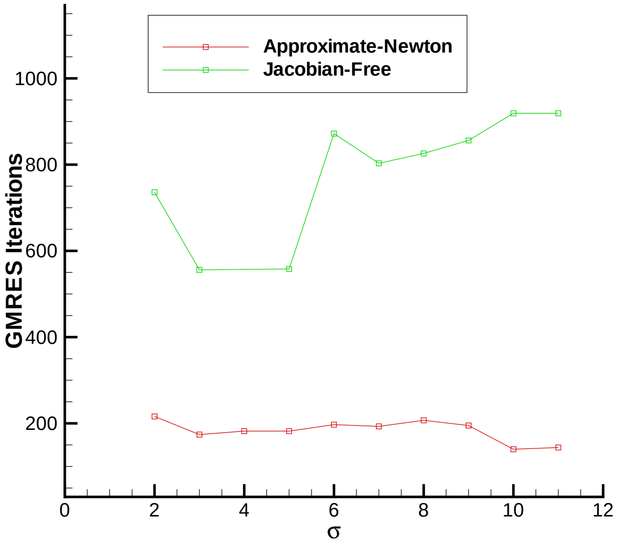


Figure 5.12: Turbulent transonic preconditioning parameter (σ) optimization (M=0.84, $\alpha = 3.0°$, Re=2.88×10^6, grid D)

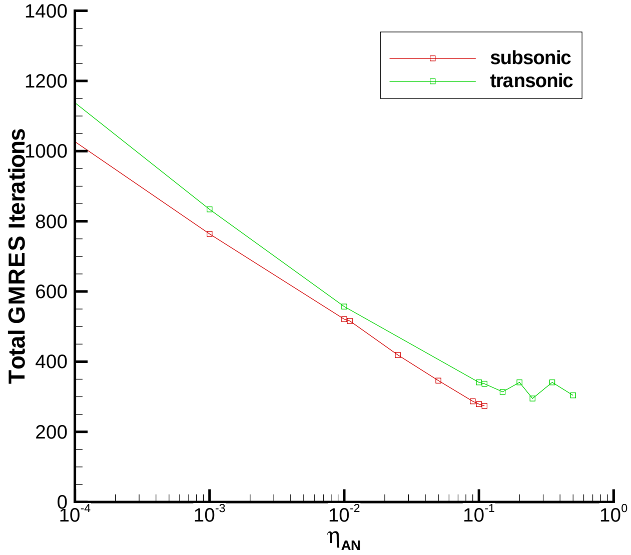


Figure 5.13: Laminar inexact-Newton parameter (η_{AN}) optimization; approximate-Newton phase (grid A)

5.2.3 Inexact-Newton Parameter (η)

The inexact-Newton parameter dictates the accuracy to which the linear system is solved. For laminar flows, the parameter was found to be the most robust for values of approximately 0.1 during the approximate-Newton startup phase. As seen in Figure 5.13, η_{AN} values less than 0.1 may tend to lead to convergence difficulties. A value between 0.01 and 0.1 for η_{JF} was found to minimize the number of GMRES iterations (Figure 5.14). Turbulent flows required values of 0.1 for both the approximate-Newton and Jacobian-free phases. This was true for subsonic and transonic flows (Figures 5.15 and 5.16).

5.2.4 Approximate-Newton Convergence Parameter ($R_{d_{tol}}$)

This threshold parameter governs the transition from the approximate-Newton startup method to the Jacobian-free method. The main indicator used is the relative residual parameter defined in in Equation 3.83. A well-chosen reduction parameter will reduce the amount of time the algorithm takes to converge while preventing divergence in the solution.

For laminar flows, a relative residual reduction of two orders of magnitude is recom-

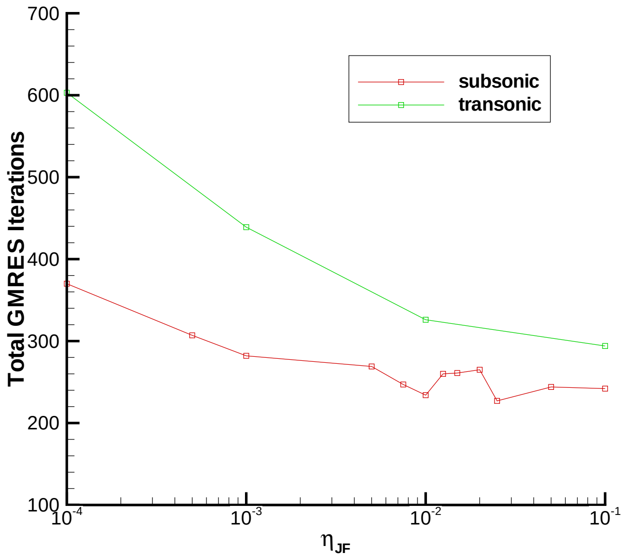


Figure 5.14: Laminar inexact-Newton parameter (η_{JF}) optimization; Jacobian-free phase (grid A)

mended for subsonic flows and factor of thirty for transonic flows (see Figure 5.17). For turbulent flows, the optimal values were found to be three orders for subsonic flows, and two orders for transonic flows (see Figure 5.18).

5.3 Summary of Optimal Parameters

The optimal parameters in TYPHOON for laminar and turbulent flows are listed in Table 5.4.

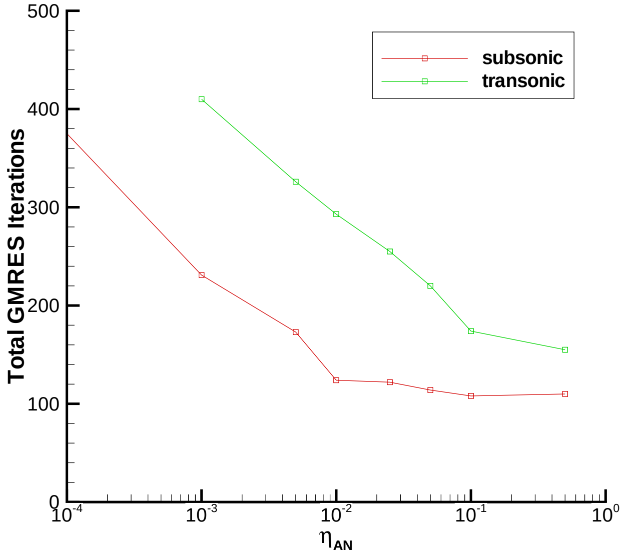


Figure 5.15: Turbulent inexact-Newton parameter (η_{AN}) optimization; approximate-Newton phase (grid C)

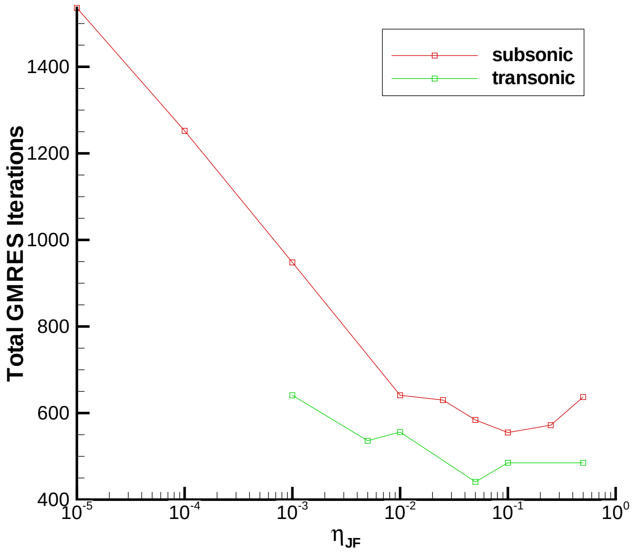


Figure 5.16: Turbulent inexact-Newton parameter (η_{JF}) optimization; Jacobian-free phase (grid D)

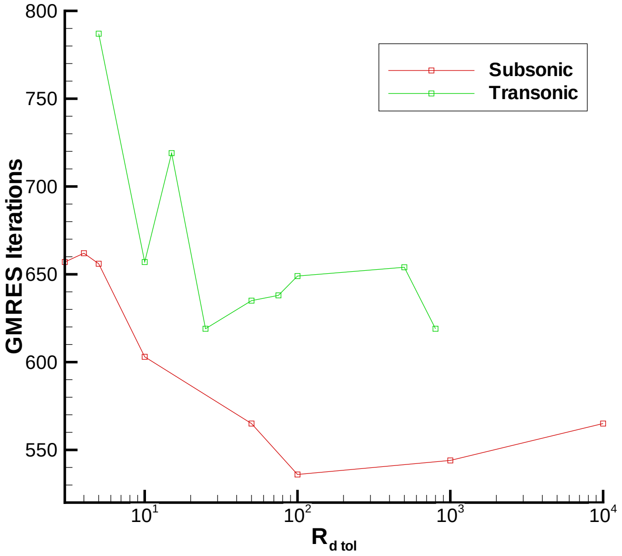


Figure 5.17: Laminar approximate-Newton residual reduction tolerance parameter ($R_{d_{tol}}$) - grids A and B

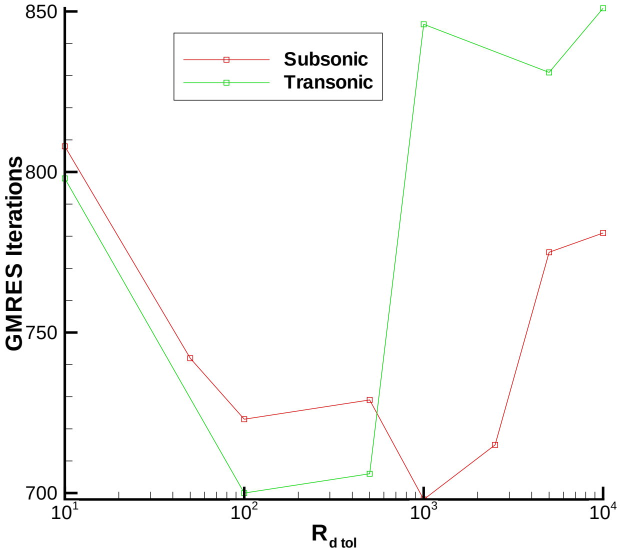


Figure 5.18: Turbulent approximate-Newton residual reduction tolerance parameter ($R_{d_{tol}}$) - grids C and D

Parameter	Laminar		Turbulent	
	Approx-Newton	Jacobian-free	Approx-Newton	Jacobian-free
ILU fill (k)	1	1	1	1
Inexact-Newton (η)	0.1	0.01	0.1	0.1
Preconditioner (σ)	4.0	6.0	3.0	3.0
	Subsonic	Transonic	Subsonic	Transonic
Residual Tol ($R_{d_{tol}}$)	100	30	1000	100

Table 5.4: Summary of results from parametric study

Chapter 6

Results and Validation

All results in this section were computed on the University of Toronto's High Performance Aerospace Computing Facility (HPACF), specifically, on the Hewlett-Packard ES45 AlphaServers and 1000MHz EV68CB Alpha Processors.

6.1 2D Validation

6.1.1 Blasius Solution

The Blasius equation is an exact solution of the boundary-layer equations for a steady incompressible two-dimensional flow over a flat plate at zero degree angle of attack. It was shown by Blasius that under these conditions, the boundary layer equations could be reformulated into a single third-order ordinary differential equation simply by selecting an appropriate coordinate transformation. The ODE can then be solved using a numerical solution procedure, such as the Runge-Kutta method [35].

The velocity profile obtained from TYPHOON is compared against the velocity profile obtained from the solution to the Blasius equation. The flow conditions are M=0.20 and Re=1000. The mesh used contains 152,106 nodes and consists of a near-wall 101×201-node block to more accurately capture the boundary layer (see Figure 6.1.1). Leading edge, trailing edge, and off-wall spacings are all $1\times10^{-5}c$. The results are plotted in nondimensional form following [20] and are shown in Figure 6.1. TYPHOON exhibits the basic shape closely with slight overshoot, but at most, the error between the Blasius solution and TYPHOON is about 2%.

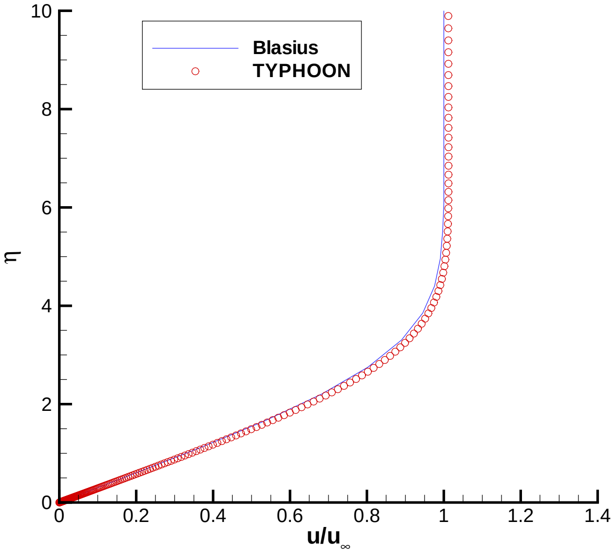


Figure 6.1: Comparison of TYPHOON surface velocity profile against Blasius solution

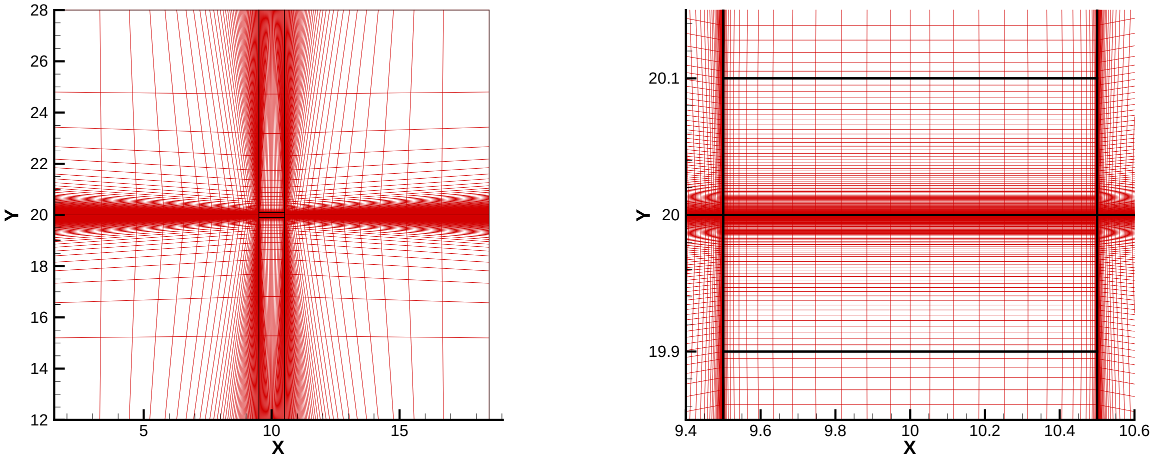
Case	Flow Condition	Mach Number	Reynolds Number	Angle of Attack (°)
1	Subsonic	0.63	600	2.0
2	Subsonic	0.63	2.88×10^6	2.0
3	Transonic	0.8	600	1.25
4	Transonic	0.8	2.88×10^6	1.25

Table 6.1: Flow conditions for the 2D validation test cases

6.1.2 NACA0012 Airfoil

The airfoil selected for validating the code is the NACA0012. Table 6.1 shows the grids and flow conditions on which the code was validated. The laminar results are compared against results obtained from OPTIMA-MB, a well-tested two-dimensional flow solver.

The six-block H-grids for the cases in this section were generated using AMBER2D, an in-house mesh generation program. The grids are extruded into six slices for use in TYPHOON. Off-wall spacing for the laminar cases is $5{\times}10^{-6}c$ and for the turbulent cases is $1{\times}10^{-6}c$ while the number of chords to the farfield is 20. Leading edge and trailing edge clustering is $10^{-4}c$ for all four grids. Clustering was performed on the upper surface



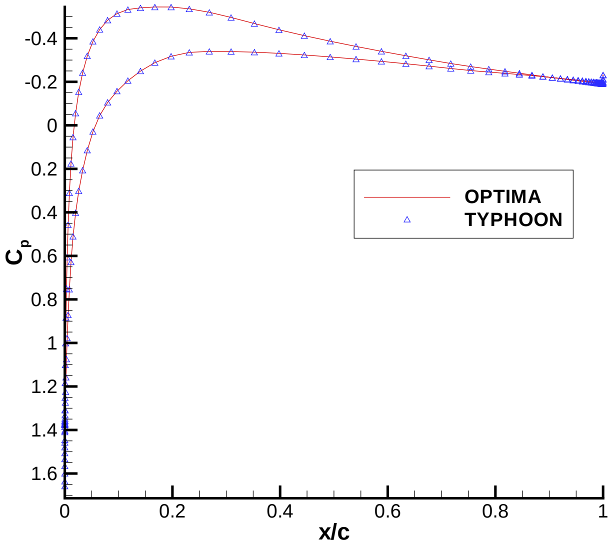
(a) complete grid (b) inner blocks

Figure 6.2: The computational grid used in the Blasius solution comparison. Note that every second node has been removed for the purposes of this graphic. Mesh lines are in red, while block boundaries are in black

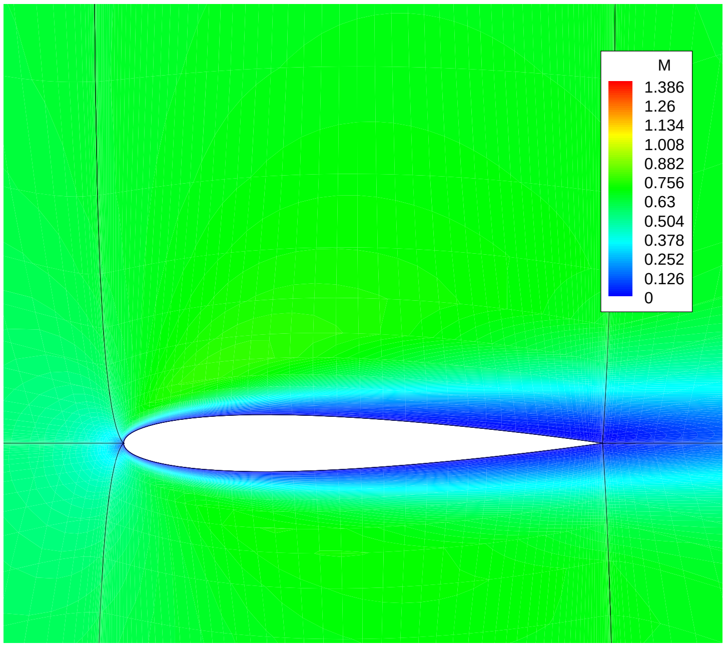
at 0.55c of the turbulent transonic airfoil in order to more accurately capture the shock. The size of the grid is 11,280 nodes for the laminar cases, 15,390 nodes for the subsonic turbulent case, and 24,888 for the transonic turbulent case. The results for the laminar cases are shown in Figures 6.3 and 6.4, while those for the turbulent cases are shown in Figures 6.5 and 6.6. The correlation between the two codes is quite good for all cases with slight discrepancies in the transonic turbulent case likely attributed to the use of the full Navier-Stokes equations in TYPHOON as opposed to the thin-layer Navier-Stokes equations in OPTIMA-MB.

6.2 3D Validation

The validation for three-dimensional flows is performed on the ONERA M6 wing. It is a benchmark validation geometry for turbulent flow solvers with a widely-compared set of experimental data collected by Charpin and Schmitt [31]. The CPU temporal unit of measure is the equivalent right-hand-side evaluation defined as the total CPU time divided by the time for one right-hand-side evaluation. This normalizes the convergence

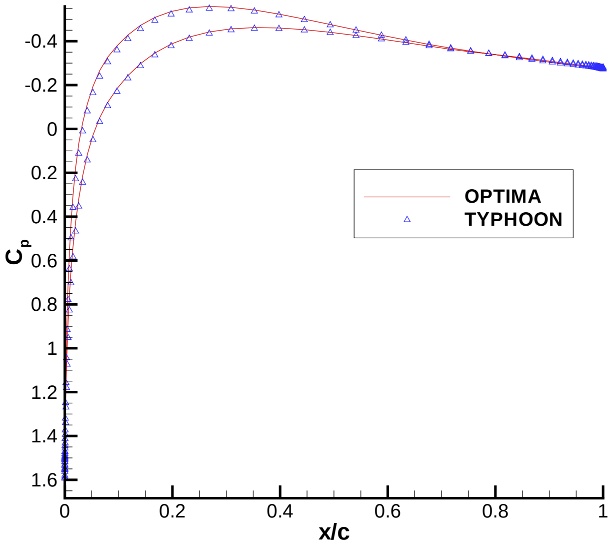


(a) C_p distribution

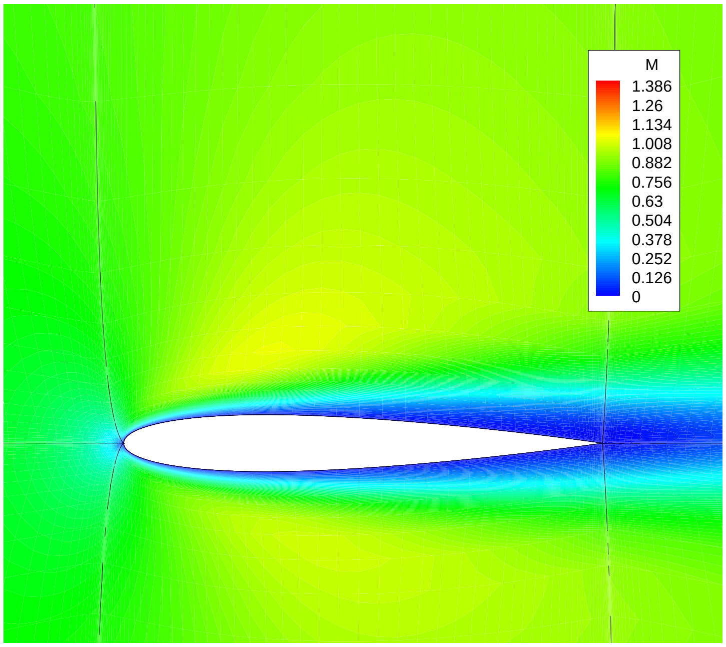


(b) Mach contours

Figure 6.3: TYPHOON laminar subsonic flow over a NACA0012 airfoil (case 1)

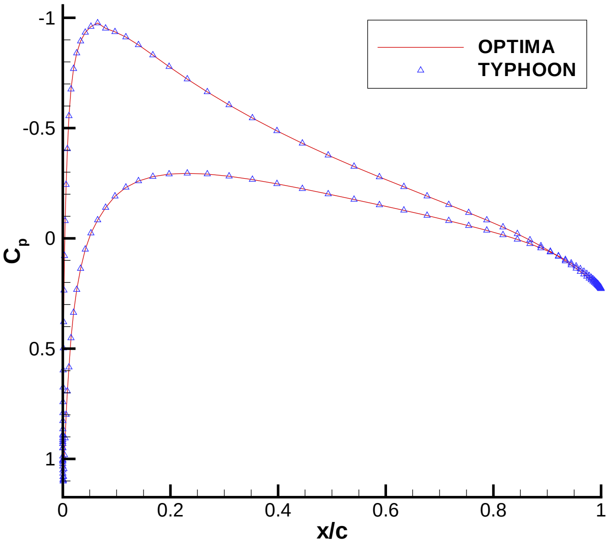


(a) C_p distribution

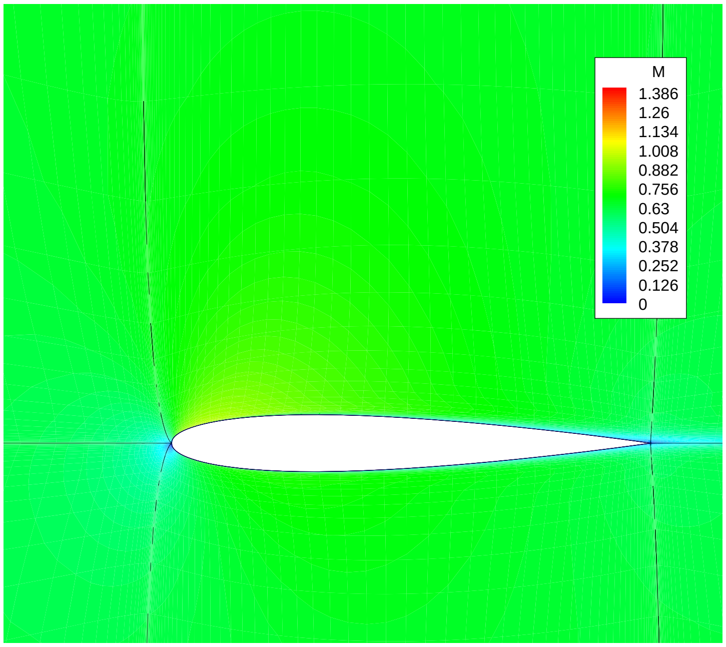


(b) Mach contours

Figure 6.4: TYPHOON laminar transonic flow over a NACA0012 airfoil (case 2)

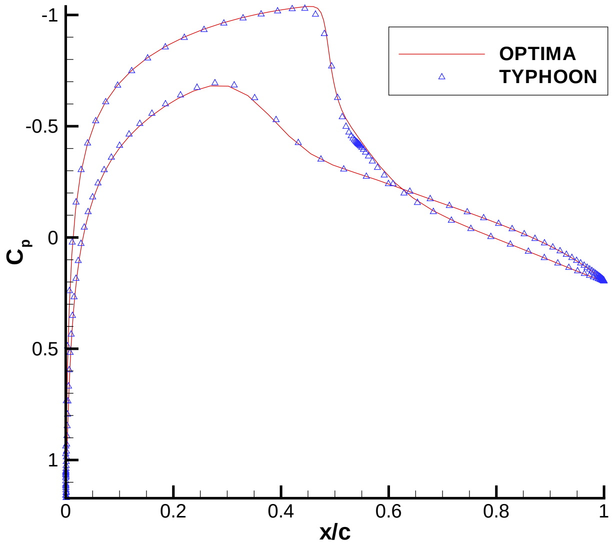


(a) C_p distribution

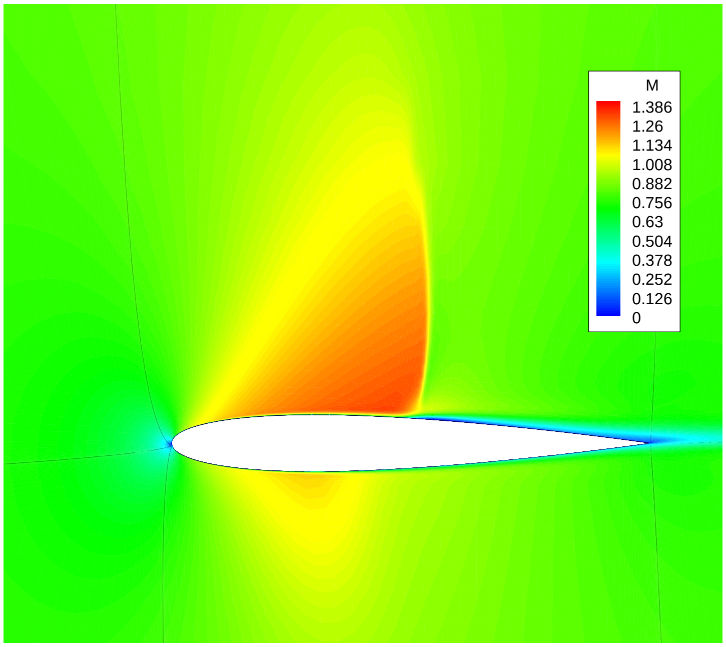


(b) Mach contours

Figure 6.5: TYPHOON turbulent subsonic flow over a NACA0012 airfoil (case 3)



(a) C_p distribution



(b) Mach contours

Figure 6.6: TYPHOON turbulent transonic flow over a NACA0012 airfoil (case 4)

Grid Size	Chords to Far Field	Off-wall Spacing ($10^{-3}c$)	LE Spacing ($10^{-3}c$)	TE Spacing ($10^{-3}c$)	Root Spacing ($10^{-3}c$)	Tip Spacing ($10^{-3}c$)	Nodes on Wing Surf
105,042	10	0.02	0.63	0.31	2.3	3.0	882
261,000	10	0.01	0.25	0.13	0.85	1.3	1250
488,880	10	0.01	0.25	0.13	0.87	1.3	2450
859,360	10	0.01	0.25	0.13	0.87	1.3	4182

Table 6.2: Summary of grid characteristics used in the 3D laminar convergence study

Grid Size	C_L	C_D
100k	0.1538	0.0700
250k	0.1425	0.0816
500k	0.1281	0.0749
800k	0.1408	0.0712

Table 6.3: TYPHOON laminar lift and drag grid convergence results for the ONERA M6 at M=0.5, Re=600, α=3.0°

histories to produce data that is independent of the CPU and computer architecture used.

6.2.1 Grid Convergence Studies

Laminar Flow

The flow conditions for the laminar study are M=0.5, Re=600, and α=3.0°. The flows were all converged to a residual of 10^{-15}. A summary of the grid characteristics is shown in Table 6.2. The coefficients of lift (C_L) and drag (C_D) for each of the grids is tabulated in Table 6.3, while the convergence histories in equivalent right-hand-side evaluations are shown in Figure 6.7.

Turbulent Flow

The flow conditions for the turbulent study are M=0.6998, Re=11.74×10^6, and α=0.04°. The TYPHOON results are compared against the experimental results obtained by

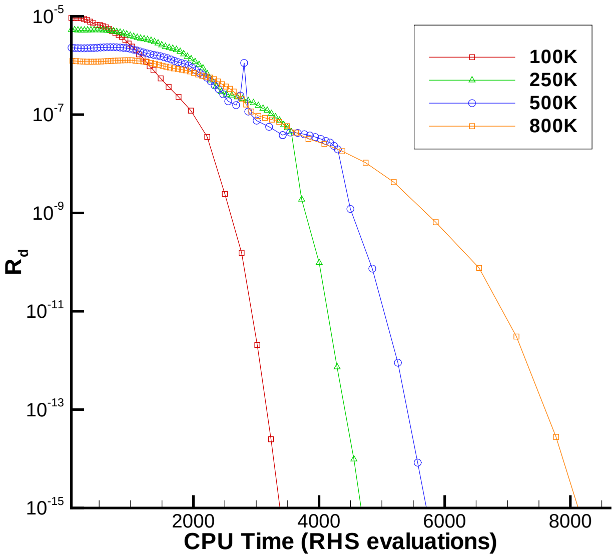


Figure 6.7: TYPHOON convergence histories for laminar grid convergence study on the ONERA M6 wing

Charpin and Schmitt in [31]. A summary of the grid characteristics is shown in Table 6.4 and flow results at six spanwise locations are shown in Figure 6.9. Note that the leading edge and tip clustering are virtually unchanged between the three grids tested, resulting from the grid generation issues discussed in Chapter 4. The convergence histories in equivalent RHS evaluations is shown in Figure 6.10. A pressure contour plot is depicted in Figure 6.11 for the 500k node grid. Convergence histories for the three grids are plotted as a function of equivalent right-hand-side evaluations in Figure 6.10. Note the effect of the turbulence model clipping as seen in the convergence histories.

The results show that the grids used tended to cause TYPHOON to underpredict the pressure coefficient at the leading edge, while over predicting it at the trailing edge. Increasing grid points in the flow field seems to improve the solution slightly, but the inability to refine spacing at the leading edge and wing tip has prevented further study of the effects of grid refinement.

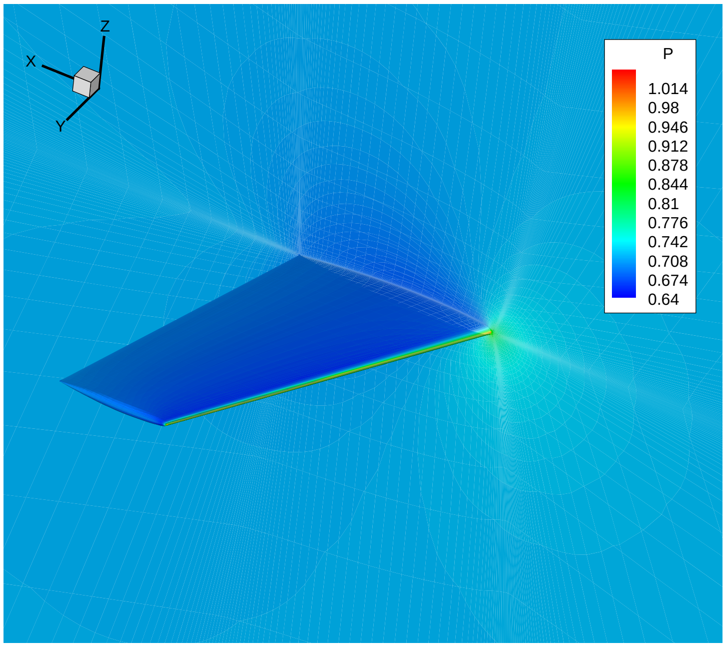
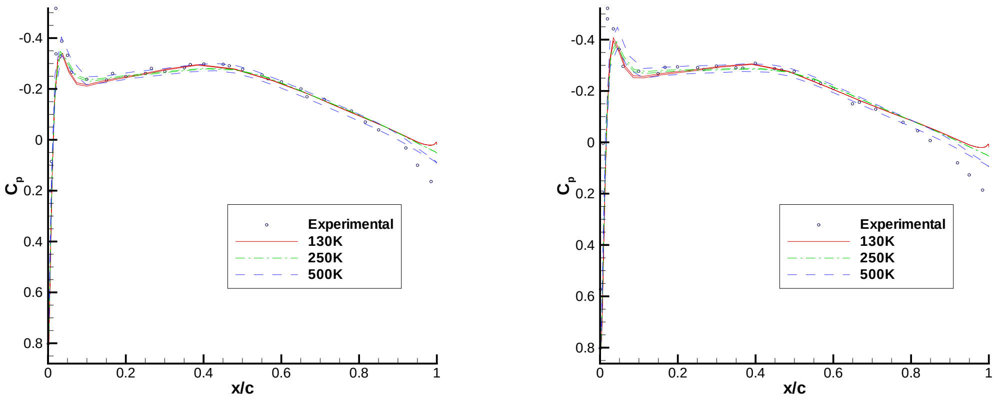

Z
X
Y
P
1.014
0.98
0.946
0.912
0.878
0.844
0.81
0.776
0.742
0.708
0.674
0.64

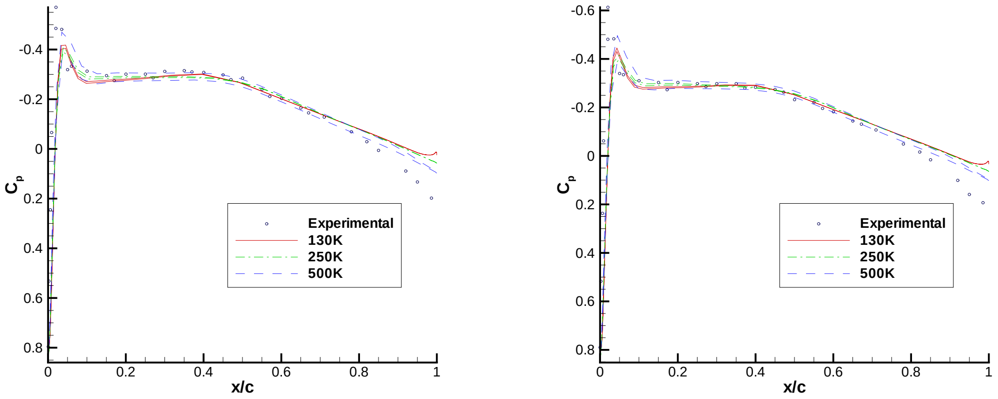
Figure 6.8: Pressure contours of an ONERA M6 wing at M=0.5, Re=600, α=3.0° (800k-node grid)

Grid Size	Chords to Far Field	Off-wall Spacing ($10^{-3}c$)	LE Spacing ($10^{-3}c$)	TE Spacing ($10^{-3}c$)	Root Spacing ($10^{-3}c$)	Tip Spacing ($10^{-3}c$)	Nodes on Wing Surf
133,152	10	0.02	0.5	0.25	2.5	2.5	3362
261,000	10	0.01	0.25	0.5	0.85	1.3	3362
488,880	10	0.03	0.25	1.0	1.0	1.3	2450

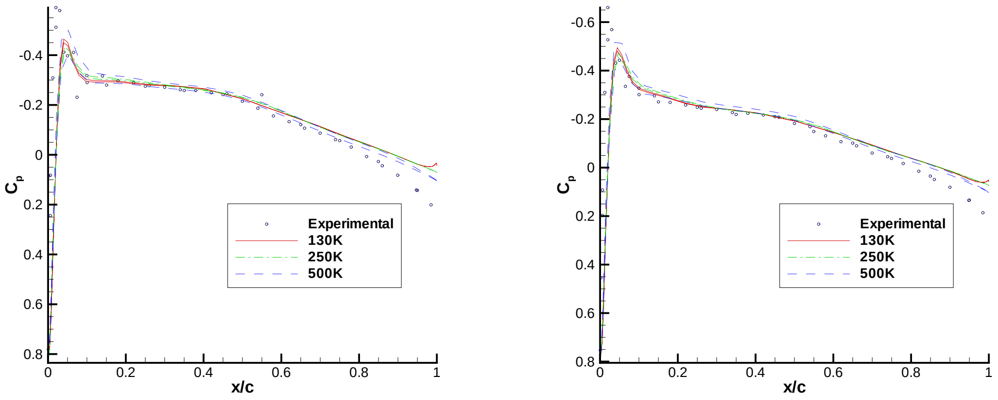
Table 6.4: Summary of grid characteristics used in 3D turbulent convergence study



(a) 20% span (b) 44% span



(c) 65% span (d) 80% span



(e) 90% span (f) 96% span

Figure 6.9: ONERA M6 wing C_p distribution, $M = 0.6998, \alpha = 0.04^{\circ}$, $\mathcal{R}e$= $11.74{\times}10^6$

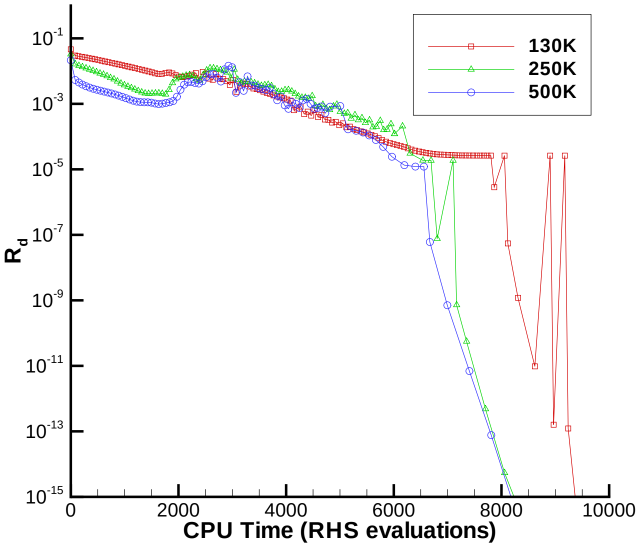


Figure 6.10: TYPHOON convergence histories for the 3D turbulent grid convergence study

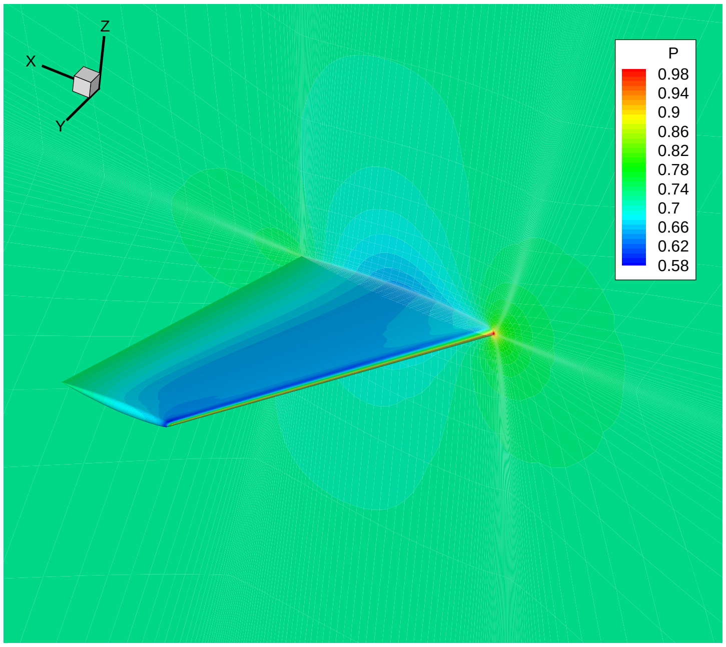

Z
X
Y
P
0.98
0.94
0.9
0.86
0.82
0.78
0.74
0.7
0.66
0.62
0.58

Figure 6.11: Pressure contours of an ONERA M6 wing at M=0.6998, Re=11.74×10^6, α=0.04° (500k-node grid)

Chapter 7

Conclusions

7.1 Summary

A 3D Navier-Stokes flow solver known as TYPHOON has been developed. TYPHOON uses structured grids and models turbulence through the one-equation Spalart-Allmaras model. The flow domain is decomposed into multiple blocks, and then transformed into the curvilinear domain. TYPHOON uses second-order finite-differencing and a second and fourth-difference dissipation scheme first presented by Jameson et al [10]. The discrete equations are solved using a quasi-Newton method. The linear system of equations arising from each Newton iteration is efficiently solved using a Krylov subspace method (GMRES). To improve the efficiency of the solver, the linear system is preconditioned using an incomplete lower/upper factorization of an approximate Jacobian matrix, and solved inexactly. A first-order approximation to the Jacobian is used at the initial startup phase of the solution convergence, switching later to a faster Jacobian-free method. This combination has been determined to provide stability and speed to the algorithm.

A numerical study has been conducted to optimize the parameters for the viscous and turbulent terms in the code. A two-dimensional validation study was performed and compared the current TYPHOON algorithm against the well-tested 2D OPTIMA flow solver. On a NACA0012 airfoil at various flight conditions both laminar and turbulent, the study shows that there is good correlation between the two.

A grid convergence study was also performed for an ONERA M6 airfoil at subsonic flow conditions for both laminar and transonic cases. Turbulent TYPHOON results for various grid sizes were compared against experimental results obtained by Charpin and

Schmitt [31] but due to grid spacing issues, convergence was not noticed.

7.2 Recommendations

It is recommended that the grid generation problems be resolved first. The inability to generate high resolution meshes affects both convergence and accuracy. One possible solution is to implement multiple-boundary condition faces into TYPHOON. This will open up new possibilities in blocking and meshing that may help to resolve tip clustering issues. Once this is done, more complex geometries such as wing-body configurations can then be attempted.

Grid sequencing has been shown by Chisholm and Zingg [4] to improve the stability of the turbulence model and should be implemented into TYPHOON. The addition of matrix dissipation will improve accuracy of the flow solution by reducing the amount of dissipation needed. Further on, the addition of tripping terms should be considered for solutions of mixed laminar and turbulent flows.

References

[1] J. D. Anderson, *Fundamentals of Aerodynamics, 3rd Ed.*, McGraw-Hill, Boston, 2001.

[2] G. A. Ashford, *An Unstructured Grid Generation and Adaptive Solution Technique for High Reynolds Number Compressible Flows*, PhD thesis, University of Michigan, 1996.

[3] T. T. Chisholm, *A Fully Coupled Newton-Krylov Solver with a One-Equation Turbulence Model*, PhD thesis, University of Toronto, 2007.

[4] T. T. Chisholm and D. W. Zingg, *A Fully-Coupled Newton-Krylov Solver for Turbulent Aerodynamic Flows*, Paper 333, ICAS 2002 Congress, Sept. 2002.

[5] E. Cuthill and J. McKee, *Reducing the Bandwidth of Sparse Symmetric Matricies*, Proceedings of the 1969 24th Association for Computing Machinery National Conference, New York, pp. 157-172, 1969.

[6] L. Eça, Orthogonal Generation Systems, in J. F. Thompson, B. K. Soni and N. P. Weatherill (Eds.), *Handbook of Grid Generation* (pp. 7-1 to 7-25), CRC Press LLC, Baca Raton (FL), 1999.

[7] N.T. Frink, *Tetrahedral Unstructured Navier-Stokes Method for Turbulent Flows*, AIAA Journal, 36 (1998), pp. 1975-1982.

[8] P. Geuzaine, *Newton-Krylov Strategy for Compressible Turbulent Flows on Unstructured Meshes*, AIAA Journal, 39 (2001), pp. 528-531.

[9] P. Godin, *Turbulence Modeling for High-Lift Multi-Element Airfoil Configurations*, PhD thesis, University of Toronto, 2004.

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

[11] D. C. Jespersen, *Parallelism and OVERFLOW*, NAS Technical Report NAS-98-013, October 1998.

[12] D. C. Jespersen, T. Pulliam and P. Buning, *Recent Enhancements to OVERFLOW (Navier-Stokes code)*, AIAA Paper 97-0644, January 1997.

[13] E.M. Lee-Rausch, *Transonic Drag Prediction on a DLR-F6 Transport Configuration Using Unstructured Grid Solvers*, AIAA Paper 2004-0554, Reno, NV, Jan. 2004.

[14] J. Lassaline, *A Navier-Stokes Equation Solver Using Agglomerated Multigrid Featuring Directional Coarsening and Line-Implicit Smoothing*, PhD Thesis, University of Toronto, 2003.

[15] T. Leung, *Parallel Implementation of a Newton-Krylov Flow Solver on Unstructured Grids*, Master's Thesis, University of Toronto, 2004.

[16] D. W. Levy, et al., *Summary of Data from the First AIAA CFD Drag Prediction Workshop*, AIAA Paper 2002-0841, Reno, NV, Jan. 2002.

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

[18] D. J. Mavriplis, *Grid Resolution of a Drag Prediction Workship Using the NSU3D Unstructured Mesh Solver*, AIAA Paper 2005-4729, Toronto, Canada, June 2005.

[19] G. May and A. Jameson, *Unstructured Algorithms for Inviscid and Viscous Flows Embedded in a Unified Solver Architecture: Flo3xx*, AIAA Paper 2005-0318, Reno, NV, Jan. 2005.

[20] G. May, E. Van der Weide, A. Jameson and L. Martinelli, *Drag Prediction of the DLR-F6 Configuration*, AIAA Paper 2004-0396, Reno, NV, Jan. 2004.

[21] W. Mulder and B. Van Leer, *Experiments With an Implicit Upwind Method for the Euler Equations*, Journal of Computational Physics, 59 (1985), pp. 232-246.

[22] M. Nemec, *Optimal Shape Design of Aerodynamic Configurations: A Newton-Krylov Approach*, PhD thesis, University of Toronto, 2003.

[23] M. Nemec and D. W. Zingg, *A Newton-Krylov Algorithm for Aerodynamic Design Using the Navier-Stokes Equations*, AIAA Journal, 40 (2002), pp. 1146-1154.

[24] J. Nichols, *A Three-Dimensional Multi-Block Newton-Krylov Flow Solver for the Euler Equations*, MASc thesis, University of Toronto, 2004.

[25] J. Nichols and D. W. Zingg *A Three-Dimensional Multi-Block Newton-Krylov Flow Solver for the Euler Equations*, AIAA Paper 2005-5231, 2005.

[26] E. Nielsen, K. Anderson, R. Walters and D. Keyes, *Application of Newton-Krylov Methodology to a Three-Dimensional Unstructured Euler Code*, AIAA Paper 95-1733, 1995.

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

[28] A. Pueyo and D. W. Zingg, *Efficient Newton-Krylov Solver for Aerodynamic Computations*, AIAA Journal, 36 (1998), pp. 1991-1997.

[29] T. H. Pulliam, *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*, NASA Ames Research Center, January 1986.

[30] Y. Saad and M. H. Schultz, *GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Problems*, SIAM Journal on Scientific and Statistical Computing, Vol. 7 (1986), pp.856-869.

[31] V. Schmitt and F. Charpin, *Pressure Distributions on the ONERA-M6 wing at Transonic Mach Numbers*, paper, Office National D'Etudes et de Recherches Aerospatiales, 1979.

[32] P. Spalart and S. Allmaras, *A One-Equation Turbulence Model for Aerodynamic Flows*, AIAA Paper 92-0439, January 1992.

[33] J. Steger, *Implicit Finite Difference Simulation of Flow About Arbitrary Geometries With Application to Airfoils*, AIAA Paper 77-665, June 1977.

[34] Transport Canada: Economic Analysis Directorate, *Transportation Canada Aviation Forecasts: 2003-2017*, Government Printing House, Ottawa, 2004.

[35] F. M. White, *Viscous Fluid Flow*, McGraw-Hill Book Company, New York, 1974.

[36] P. Wong, D. W. Zingg, *Three-Dimensional Aerodynamic Computations on Unstructured Grids Using a Newton-Krylov Approach*, AIAA Paper 2005-5231, Toronto, ON, June 2005.

[37] P. Wong, *A Compressible Navier-Stokes Flow Solver Using the Newton-Krylov Method on Unstructured Grids*, PhD thesis, University of Toronto, 2006.

Appendices

Appendix A

3D Curvilinear Coordinate Transformation

Cartesian coordinates can be described as a function of curvilinear coordinates ξ, η and ζ.

$$\begin{aligned} \tau &= t \\ \xi &= \xi(x, y, z, t) \\ \eta &= \eta(x, y, z, t) \\ \zeta &= \zeta(x, y, z, t) \end{aligned}$$

Metric relations can be acquired by expanding the Cartesian partial derivative operators in terms of the curvilinear coordinates using the chain rule.

$$\begin{bmatrix} \partial_t \\ \partial_x \\ \partial_y \\ \partial_z \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & \xi_t & \eta_t & \zeta_t \\ 0 & \xi_x & \xi_y & \xi_z \\ 0 & \eta_x & \eta_y & \eta_z \\ 0 & \zeta_x & \zeta_y & \zeta_z \end{bmatrix}}_{A} \begin{bmatrix} \partial_\tau \\ \partial_\xi \\ \partial_\eta \\ \partial_\zeta \end{bmatrix}$$

Conversely, one can take the curvilinear partial derivative operators and perform a similar expansion:

$$\begin{bmatrix} \partial_\tau \\ \partial_\xi \\ \partial_\eta \\ \partial_\zeta \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & t_\xi & t_\eta & t_\zeta \\ 0 & x_\xi & y_\xi & z_\xi \\ 0 & x_\eta & y_\eta & z_\eta \\ 0 & x_\zeta & y_\zeta & z_\zeta \end{bmatrix}}_{B} \begin{bmatrix} \partial_t \\ \partial_x \\ \partial_y \\ \partial_z \end{bmatrix}$$

$$\begin{bmatrix} \partial_t \\ \partial_x \\ \partial_y \\ \partial_z \end{bmatrix} = J \underbrace{\begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ 0 & b_{22} & b_{23} & b_{24} \\ 0 & b_{32} & b_{33} & b_{34} \\ 0 & b_{42} & b_{43} & b_{44} \end{bmatrix}}_{B^{-1}} \begin{bmatrix} \partial_\tau \\ \partial_\xi \\ \partial_\eta \\ \partial_\zeta \end{bmatrix}$$

where J, the inverse determinant of A (also called the metric Jacobian), can be written as:

$$J^{-1} = x_\xi y_\eta z_\zeta + y_\xi z_\eta x_\zeta + z_\xi x_\eta y_\zeta - x_\xi z_\eta y_\zeta - y_\xi x_\eta z_\zeta - z_\xi y_\eta x_\zeta$$

The components in B are given by:

$$\begin{aligned}
b_{11} &= x_\xi y_\xi z_\zeta + y_\xi z_\eta x_\zeta + z_\xi x_\eta y_\zeta - x_\xi z_\eta y_\zeta - y_\xi x_\eta z_\zeta - z_\xi y_\eta x_\zeta \\
b_{12} &= x_t z_\eta y_\zeta + y_t x_\eta z_\zeta + z_t y_\eta x_\zeta - x_t y_\eta z_\zeta - y_t z_\eta x_\zeta - z_t x_\eta y_\zeta \\
b_{13} &= x_t y_\xi z_\zeta + y_t z_\xi x_\zeta + z_t x_\xi y_\zeta - x_t z_\xi y_\zeta - y_t x_\xi z_\zeta - z_t y_\xi x_\zeta \\
b_{14} &= x_t z_\xi y_\eta + y_t x_\xi z_\eta + z_t y_\xi x_\eta - x_t y_\xi z_\eta - y_t z_\xi x_\eta - z_t x_\xi y_\eta \\
b_{22} &= y_\eta z_\zeta - z_\eta y_\zeta \\
b_{23} &= z_\xi y_\zeta - y_\xi z_\zeta \\
b_{24} &= y_\xi z_\eta - z_\xi y_\eta \\
b_{32} &= z_\eta x_\zeta - x_\eta z_\zeta \\
b_{33} &= x_\xi z_\zeta - z_\xi x_\zeta \\
b_{34} &= z_\xi x_\eta - x_\xi z_\eta \\
b_{42} &= x_\eta y_\zeta - y_\eta x_\zeta \\
b_{43} &= y_\xi x_\zeta - x_\xi y_\zeta \\
b_{44} &= x_\xi y_\eta - y_\xi x_\eta
\end{aligned}$$

By comparing the individual terms in A^{-1} with those in B, one can arrive at the metric relations with which we can write the Cartesian derivatives in terms of curvilinear derivatives.

$$\begin{aligned}
\partial_x &= J\left[(y_\eta z_\zeta - z_\eta y_\zeta)\partial_\xi + (z_\xi y_\zeta - y_\xi z_\zeta)\partial_\eta + (y_\xi z_\eta - z_\xi y_\eta)\partial_\zeta\right] \\
&= J\left(\xi_x \partial_\xi + \eta_x \partial_\eta + \zeta_x \partial_\zeta\right) \qquad \text{(A.1)}
\end{aligned}$$

$$\begin{aligned}\partial_y &= J\left[(z_\eta x_\zeta - x_\eta z_\zeta)\partial_\xi + (x_\xi z_\zeta - z_\xi x_\zeta)\partial_\eta + (z_\xi x_\eta - x_\xi z_\eta)\partial_\zeta\right] \\ &= J\left(\xi_y \partial_\xi + \eta_y \partial_\eta + \zeta_y \partial_\zeta\right) \qquad \text{(A.2)}\end{aligned}$$

$$\begin{aligned}\partial_z &= J\left[(x_\eta y_\zeta - y_\eta x_\zeta)\partial_\xi + (y_\xi x_\zeta - x_\xi y_\zeta)\partial_\eta + (x_\xi y_\eta - y_\xi x_\eta)\partial_\zeta\right] \\ &= J\left(\xi_z \partial_\xi + \eta_z \partial_\eta + \zeta_z \partial_\zeta\right) \qquad \text{(A.3)}\end{aligned}$$

Averaged values of the centered differences for the grid values (x_ξ, x_η, x_ζ, etc.) are used to calculate the metrics (ξ_x, ξ_y, ξ_x, etc.) such that metric invariants are satisfied and can be found in Section 2.2.1 of Nichols' thesis in [24].

A.1 3D Viscous Flux Terms in Curvilinear Coordinates

Using the above metric relations, we can describe the viscous flux vectors by the following expressions:

$$E_v = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ \tau_{xx}u + \tau_{xy}v + \tau_{xz}w + kT_x \end{bmatrix}, \quad F_v = \begin{bmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ \tau_{yz} \\ \tau_{yx}u + \tau_{yy}v + \tau_{yz}w + kT_y \end{bmatrix},$$

$$G_v = \begin{bmatrix} 0 \\ \tau_{zx} \\ \tau_{zy} \\ \tau_{zz} \\ \tau_{zx}u + \tau_{zy}v + \tau_{zz}w + kT_z \end{bmatrix}$$

The shear and normal stresses are given by:

$$
\begin{aligned}
\tau_{xx} &= 2(\mu+\mu_t)u_x - \frac{2}{3}(\mu+\mu_t)\left(u_x+v_y+w_z\right)\\
\tau_{xy} &= (\mu+\mu_t)\left(u_y+v_x\right)\\
\tau_{xz} &= (\mu+\mu_t)\left(u_z+w_x\right)\\
\tau_{yx} &= \tau_{xy}\\
\tau_{yy} &= 2(\mu+\mu_t)v_y - \frac{2}{3}(\mu+\mu_t)\left(u_x+v_y+w_z\right)\\
\tau_{yz} &= (\mu+\mu_t)\left(v_z+w_y\right)\\
\tau_{zx} &= \tau_{xz}\\
\tau_{zy} &= \tau_{yz}\\
\tau_{zz} &= 2(\mu+\mu_t)w_z - \frac{2}{3}(\mu+\mu_t)\left(u_x+v_y+w_z\right)
\end{aligned}
$$

By applying curvilinear coordinate transformation, the viscous flux vectors become:

$$
\begin{aligned}
\hat{E}_v &= J^{-1}\left(\xi_x E_v + \xi_y F_v + \xi_z G_v\right)\\
\hat{F}_v &= J^{-1}\left(\eta_x E_v + \eta_y F_v + \eta_z G_v\right)\\
\hat{G}_v &= J^{-1}\left(\zeta_x E_v + \zeta_y F_v + \zeta_z G_v\right)
\end{aligned}
$$

where the shear stresses in general curvilinear coordintes are given by:

$$
\begin{aligned}
\tau_{xx} &= (\mu+\mu_t)\left[\frac{4}{3}\left(\xi_x u_\xi + \eta_x u_\eta + \zeta_x u_\zeta\right) - \frac{2}{3}\left(\xi_y v_\xi + \eta_y v_\eta + \zeta_y v_\zeta + \xi_z w_\xi + \eta_z w_\eta + \zeta_z w_\zeta\right)\right]\\
\tau_{xy} &= (\mu+\mu_t)\left(\xi_y u_\xi + \eta_y u_\eta + \zeta_y u_\zeta + \xi_x v_\xi + \eta_x v_\eta + \zeta_x v_\zeta\right)\\
\tau_{xz} &= (\mu+\mu_t)\left(\xi_z u_\xi + \eta_z u_\eta + \zeta_z u_\zeta + \xi_x w_\xi + \eta_x w_\eta + \zeta_x w_\zeta\right)\\
\tau_{yy} &= (\mu+\mu_t)\left[\frac{4}{3}\left(\xi_y v_\xi + \eta_y v_\eta + \zeta_y v_\zeta\right) - \frac{2}{3}\left(\xi_x u_\xi + \eta_x u_\eta + \zeta_x u_\zeta + \xi_z w_\xi + \eta_z w_\eta + \zeta_z w_\zeta\right)\right]\\
\tau_{yz} &= (\mu+\mu_t)\left(\xi_z v_\xi + \eta_z v_\eta + \zeta_z v_\zeta + \xi_y w_\xi + \eta_y w_\eta + \zeta_y w_\zeta\right)\\
\tau_{zz} &= (\mu+\mu_t)\left[\frac{4}{3}\left(\xi_z w_\xi + \eta_z w_\eta + \zeta_z w_\zeta\right) - \frac{2}{3}\left(\xi_x u_\xi + \eta_x u_\eta + \zeta_x u_\zeta + \xi_y v_\xi + \eta_y v_\eta + \zeta_y v_\zeta\right)\right]
\end{aligned}
$$

Also, if Fourier's law of heat conduction is used, the kT_x, kT_y and kT_z terms in general curvilinear coordinates become:

$$
\begin{aligned}
kT_x &= (\mu \mathcal{P}r^{-1} + \mu_t \mathcal{P}r_t^{-1})(\gamma-1)^{-1}[\xi_x\partial_\xi(a^2) + \eta_x\partial_\eta(a^2) + \zeta_x\partial_\zeta(a^2)]\\
kT_y &= (\mu \mathcal{P}r^{-1} + \mu_t \mathcal{P}r_t^{-1})(\gamma-1)^{-1}[\xi_y\partial_\xi(a^2) + \eta_y\partial_\eta(a^2) + \zeta_y\partial_\zeta(a^2)]\\
kT_z &= (\mu \mathcal{P}r^{-1} + \mu_t \mathcal{P}r_t^{-1})(\gamma-1)^{-1}[\xi_z\partial_\xi(a^2) + \eta_z\partial_\eta(a^2) + \zeta_z\partial_\zeta(a^2)]
\end{aligned}
$$

$$\hat{E}_v = J^{-1}\begin{bmatrix} 0 \\ \xi_x\tau_{xx} + \xi_y\tau_{xy} + \xi_z\tau_{xz} \\ \xi_x\tau_{yx} + \xi_y\tau_{yy} + \xi_z\tau_{yz} \\ \xi_x\tau_{zx} + \xi_y\tau_{zy} + \xi_z\tau_{zz} \\ E_{v,5} \end{bmatrix}$$

$$\hat{F}_v = J^{-1}\begin{bmatrix} 0 \\ \eta_x\tau_{xx} + \eta_y\tau_{xy} + \eta_z\tau_{xz} \\ \eta_x\tau_{yx} + \eta_y\tau_{yy} + \eta_z\tau_{yz} \\ \eta_x\tau_{zx} + \eta_y\tau_{zy} + \eta_z\tau_{zz} \\ F_{v,5} \end{bmatrix}$$

$$\hat{G}_v = J^{-1}\begin{bmatrix} 0 \\ \zeta_x\tau_{xx} + \zeta_y\tau_{xy} + \zeta_z\tau_{xz} \\ \zeta_x\tau_{yx} + \zeta_y\tau_{yy} + \zeta_z\tau_{yz} \\ \zeta_x\tau_{zx} + \zeta_y\tau_{zy} + \zeta_z\tau_{zz} \\ G_{v,5} \end{bmatrix}$$

$$\begin{aligned} E_{v,5} &= \tau_{xx}u + \tau_{xy}v + \tau_{xz}w + \left(\frac{\mu\mathcal{P}r^{-1} + \mu_t\mathcal{P}r_t^{-1}}{\gamma - 1}\right)\left[\xi_z\partial_\xi(a^2) + \eta_z\partial_\eta(a^2) + \zeta_z\partial_\zeta(a^2)\right] \\ F_{v,5} &= \tau_{yx}u + \tau_{yy}v + \tau_{yz}w + \left(\frac{\mu\mathcal{P}r^{-1} + \mu_t\mathcal{P}r_t^{-1}}{\gamma - 1}\right)\left[\xi_y\partial_\xi(a^2) + \eta_y\partial_\eta(a^2) + \zeta_y\partial_\zeta(a^2)\right] \\ G_{v,5} &= \tau_{zx}u + \tau_{zy}v + \tau_{zz}w + \left(\frac{\mu\mathcal{P}r^{-1} + \mu_t\mathcal{P}r_t^{-1}}{\gamma - 1}\right)\left[\xi_z\partial_\xi(a^2) + \eta_z\partial_\eta(a^2) + \zeta_z\partial_\zeta(a^2)\right] \end{aligned}$$

A.2 Turbulence Model in Curvilinear Coordinates

Recall the turbulence model from Equation 2.7. We now expand the substantial derivative and drop the trip functions:

$$\begin{aligned} \frac{\partial\tilde{\nu}}{\partial t} + u\frac{\partial\tilde{\nu}}{\partial x} + v\frac{\partial\tilde{\nu}}{\partial y} + w\frac{\partial\tilde{\nu}}{\partial z} = \frac{c_{b1}}{Re}\tilde{S}\tilde{\nu} &+ \frac{1 + c_{b2}}{\tilde{\sigma}Re}\nabla\cdot\left[(\nu + \tilde{\nu})\nabla\tilde{\nu}\right] - \frac{c_{b2}}{\tilde{\sigma}Re}(\nu + \tilde{\nu})\nabla^2\tilde{\nu} \\ &- \frac{1}{Re}c_{w1}f_w\left(\frac{\tilde{\nu}}{d}\right)^2 \end{aligned} \tag{A.4}$$

We ignore all time terms since we are only concerned with steady-state solutions and write the spatial derivatives in curvilinear coordinates using Equations A.1 through A.3:

$$\begin{aligned}
u\frac{\partial\tilde{\nu}}{\partial x} &= Ju\left(\xi_x\frac{\partial\tilde{\nu}}{\partial\xi}+\eta_x\frac{\partial\tilde{\nu}}{\partial\eta}+\zeta_x\frac{\partial\tilde{\nu}}{\partial\zeta}\right)\\
v\frac{\partial\tilde{\nu}}{\partial y} &= Jv\left(\xi_y\frac{\partial\tilde{\nu}}{\partial\xi}+\eta_y\frac{\partial\tilde{\nu}}{\partial\eta}+\zeta_y\frac{\partial\tilde{\nu}}{\partial\zeta}\right)\\
w\frac{\partial\tilde{\nu}}{\partial z} &= Jw\left(\xi_z\frac{\partial\tilde{\nu}}{\partial\xi}+\eta_z\frac{\partial\tilde{\nu}}{\partial\eta}+\zeta_z\frac{\partial\tilde{\nu}}{\partial\zeta}\right)
\end{aligned}$$

Now rewriting the left hand side of Equation A.4:

$$J\underbrace{(\xi_x u+\xi_y v+\xi_z w)}_{U}\frac{\partial\tilde{\nu}}{\partial\xi}+J\underbrace{(\eta_x u+\eta_y v+\eta_z w)}_{V}\frac{\partial\tilde{\nu}}{\partial\eta}+J\underbrace{(\zeta_x u+\zeta_y v+\zeta_z w)}_{W}\frac{\partial\tilde{\nu}}{\partial\zeta}$$

On the right hand side, we must deal with the spatial derivatives within ∇ and $\tilde{S}$. The transformation of vorticity within $\tilde{S}$ is carried out in Appendix B. The derivatives in the del operator can be transformed in a rather straight forward manner and yield:

$$\begin{aligned}
\nabla &= \frac{\partial}{\partial x}+\frac{\partial}{\partial y}+\frac{\partial}{\partial z}\\
&= J\left(\xi_x\partial_\xi+\eta_x\partial_\eta+\zeta_x\partial_\zeta+\xi_y\partial_\xi+\eta_y\partial_\eta+\zeta_y\partial_\zeta+\xi_z\partial_\xi+\eta_z\partial_\eta+\zeta_z\partial_\zeta\right)
\end{aligned}$$

$$\begin{aligned}
\nabla^2 &= \frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}+\frac{\partial^2}{\partial z^2}\\
&= J^2\big[\xi_x\partial_\xi\left(\xi_x\partial_\xi+\eta_x\partial_\eta+\zeta_x\partial_\zeta\right)+\eta_x\partial_\eta\left(\xi_x\partial_\xi+\eta_x\partial_\eta+\zeta_x\partial_\zeta\right)+\zeta_x\partial_\zeta\left(\xi_x\partial_\xi+\eta_x\partial_\eta+\zeta_x\partial_\zeta\right)\\
&+ \xi_y\partial_\xi\left(\xi_y\partial_\xi+\eta_y\partial_\eta+\zeta_y\partial_\zeta\right)+\eta_y\partial_\eta\left(\xi_y\partial_\xi+\eta_y\partial_\eta+\zeta_y\partial_\zeta\right)+\zeta_y\partial_\zeta\left(\xi_y\partial_\xi+\eta_y\partial_\eta+\zeta_y\partial_\zeta\right)\\
&+ \xi_z\partial_\xi\left(\xi_z\partial_\xi+\eta_z\partial_\eta+\zeta_z\partial_\zeta\right)+\eta_z\partial_\eta\left(\xi_z\partial_\xi+\eta_z\partial_\eta+\zeta_z\partial_\zeta\right)+\zeta_z\partial_\zeta\left(\xi_z\partial_\xi+\eta_z\partial_\eta+\zeta_z\partial_\zeta\right)\big]
\end{aligned}$$

By dropping cross-derivative terms in the above equation, we arrive at the Spalart-Allmaras turbulence model in curvilinear coordinates:

$$J\left(U\frac{\partial\tilde{\nu}}{\partial\xi}+V\frac{\partial\tilde{\nu}}{\partial\eta}+W\frac{\partial\tilde{\nu}}{\partial\zeta}\right)=c_{b1}\tilde{S}\tilde{\nu}+\frac{1+c_{b2}}{\tilde{\sigma}}J^2T_1-\frac{c_{b2}}{\tilde{\sigma}}(\nu+\tilde{\nu})J^2T_2-c_{w1}f_w\left(\frac{\tilde{\nu}}{d}\right)^2$$

where

$$\begin{aligned}
T_1 &= \xi_x\frac{\partial}{\partial\xi}\left[(\nu+\tilde{\nu})\xi_x\frac{\partial\tilde{\nu}}{\partial\xi}\right]+\eta_x\frac{\partial}{\partial\eta}\left[(\nu+\tilde{\nu})\eta_x\frac{\partial\tilde{\nu}}{\partial\eta}\right]+\zeta_x\frac{\partial}{\partial\zeta}\left[(\nu+\tilde{\nu})\zeta_x\frac{\partial\tilde{\nu}}{\partial\zeta}\right]\\
&+ \xi_y\frac{\partial}{\partial\xi}\left[(\nu+\tilde{\nu})\xi_y\frac{\partial\tilde{\nu}}{\partial\xi}\right]+\eta_y\frac{\partial}{\partial\eta}\left[(\nu+\tilde{\nu})\eta_y\frac{\partial\tilde{\nu}}{\partial\eta}\right]+\zeta_y\frac{\partial}{\partial\zeta}\left[(\nu+\tilde{\nu})\zeta_y\frac{\partial\tilde{\nu}}{\partial\zeta}\right]\\
&+ \xi_z\frac{\partial}{\partial\xi}\left[(\nu+\tilde{\nu})\xi_z\frac{\partial\tilde{\nu}}{\partial\xi}\right]+\eta_z\frac{\partial}{\partial\eta}\left[(\nu+\tilde{\nu})\eta_z\frac{\partial\tilde{\nu}}{\partial\eta}\right]+\zeta_z\frac{\partial}{\partial\zeta}\left[(\nu+\tilde{\nu})\zeta_z\frac{\partial\tilde{\nu}}{\partial\zeta}\right]
\end{aligned}$$

$$
\begin{aligned}
T_2 &= \xi_x \frac{\partial \tilde{\nu}}{\partial \xi}\left(\xi_x \frac{\partial \tilde{\nu}}{\partial \xi}\right) + \eta_x \frac{\partial \tilde{\nu}}{\partial \eta}\left(\eta_x \frac{\partial \tilde{\nu}}{\partial \eta}\right) + \zeta_x \frac{\partial \tilde{\nu}}{\partial \zeta}\left(\zeta_x \frac{\partial \tilde{\nu}}{\partial \zeta}\right) \\
&+ \xi_y \frac{\partial \tilde{\nu}}{\partial \xi}\left(\xi_y \frac{\partial \tilde{\nu}}{\partial \xi}\right) + \eta_y \frac{\partial \tilde{\nu}}{\partial \eta}\left(\eta_y \frac{\partial \tilde{\nu}}{\partial \eta}\right) + \zeta_y \frac{\partial \tilde{\nu}}{\partial \zeta}\left(\zeta_y \frac{\partial \tilde{\nu}}{\partial \zeta}\right) \\
&+ \xi_z \frac{\partial \tilde{\nu}}{\partial \xi}\left(\xi_z \frac{\partial \tilde{\nu}}{\partial \xi}\right) + \eta_z \frac{\partial \tilde{\nu}}{\partial \eta}\left(\eta_z \frac{\partial \tilde{\nu}}{\partial \eta}\right) + \zeta_z \frac{\partial \tilde{\nu}}{\partial \zeta}\left(\zeta_z \frac{\partial \tilde{\nu}}{\partial \zeta}\right)
\end{aligned}
$$

Appendix B

Vorticity Differentiation

Vorticity:

$$\mathbf{S} = \left[\left(\frac{\partial w}{\partial y} - \frac{\partial v}{\partial z} \right) \mathbf{i} + \left(\frac{\partial u}{\partial z} - \frac{\partial w}{\partial x} \right) \mathbf{j} + \left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \right) \mathbf{k} \right]$$

In curvilinear coordinates:

$$\begin{aligned} \mathbf{S} = & J\left[(\xi_y \partial_\xi w + \eta_y \partial_\eta w + \zeta_y \partial_\zeta w) - (\xi_z \partial_\xi v + \eta_z \partial_\eta v + \zeta_z \partial_\zeta v) \right] \mathbf{i} \\ & + J\left[(\xi_z \partial_\xi u + \eta_z \partial_\eta u + \zeta_z \partial_\zeta u) - (\xi_x \partial_\xi w + \eta_x \partial_\eta w + \zeta_x \partial_\zeta w) \right] \mathbf{j} \\ & + J\left[(\xi_x \partial_\xi v + \eta_x \partial_\eta v + \zeta_x \partial_\zeta v) - (\xi_y \partial_\xi u + \eta_y \partial_\eta u + \zeta_y \partial_\zeta u) \right] \mathbf{k} \\ & = J\left(S_1 \mathbf{i} + S_2 \mathbf{j} + S_3 \mathbf{k} \right) \end{aligned}$$

Discretized using second-order centered-difference method:

$$\begin{aligned} S_1 = & \frac{1}{2}[\xi_y (w_{j+1,k,m} - w_{j-1,k,m}) + \eta_y (w_{j,k+1,m} - w_{j,k-1,m}) \\ & + \zeta_y (w_{j,k,m+1} - w_{j,k,m-1}) - \xi_z (v_{j+1,k,m} - v_{j-1,k,m}) \\ & - \eta_z (v_{j,k+1,m} - v_{j,k-1,m}) - \zeta_z (v_{j,k,m+1} - v_{j,k,m-1})] \\ S_2 = & \frac{1}{2}[\xi_z (u_{j+1,k,m} - u_{j-1,k,m}) + \eta_z (u_{j,k+1,m} - u_{j,k-1,m}) \\ & + \zeta_z (u_{j,k,m+1} - u_{j,k,m-1}) - \xi_x (w_{j+1,k,m} - w_{j-1,k,m}) \\ & - \eta_x (w_{j,k+1,m} - w_{j,k-1,m}) - \zeta_x (w_{j,k,m+1} - w_{j,k,m-1})] \\ S_3 = & \frac{1}{2}[\xi_x (v_{j+1,k,m} - v_{j-1,k,m}) + \eta_x (v_{j,k+1,m} - v_{j,k-1,m}) \\ & + \zeta_x (v_{j,k,m+1} - v_{j,k,m-1}) - \xi_y (u_{j+1,k,m} - u_{j-1,k,m}) \\ & - \eta_y (u_{j,k+1,m} - u_{j,k-1,m}) - \zeta_y (u_{j,k,m+1} - u_{j,k,m-1})] \end{aligned}$$

The flow variables are given by the vector:

$$\mathbf{Q}_{j,k,m} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho e \\ \tilde{\nu} \end{bmatrix}_{j,k,m}$$

Write vorticity in terms of flow variables Q (note that $\hat{Q}_1 = JQ_1$)

$$\begin{aligned} S_1(\mathbf{Q}) = &\frac{1}{2}\xi_y[(Q_4/\hat{Q}_1)_{j+1,k,m} - (Q_4/\hat{Q}_1)_{j-1,k,m}] \\ &+ \frac{1}{2}\eta_y[(Q_4/\hat{Q}_1)_{j,k+1,m} - (Q_4/\hat{Q}_1)_{j,k-1,m}] \\ &+ \frac{1}{2}\zeta_y[(Q_4/\hat{Q}_1)_{j,k,m+1} - (Q_4/\hat{Q}_1)_{j,k,m-1}] \\ &- \frac{1}{2}\xi_z[(Q_3/\hat{Q}_1)_{j+1,k,m} - (Q_3/\hat{Q}_1)_{j-1,k,m}] \\ &- \frac{1}{2}\eta_z[(Q_3/\hat{Q}_1)_{j,k+1,m} - (Q_3/\hat{Q}_1)_{j,k-1,m}] \\ &- \frac{1}{2}\zeta_z[(Q_3/\hat{Q}_1)_{j,k,m+1} - (Q_3/\hat{Q}_1)_{j,k,m-1}] \end{aligned}$$

$$\begin{aligned} S_2(\mathbf{Q}) = &\frac{1}{2}\xi_z[(Q_2/\hat{Q}_1)_{j+1,k,m} - (Q_2/\hat{Q}_1)_{j-1,k,m}] \\ &+ \frac{1}{2}\eta_z[(Q_2/\hat{Q}_1)_{j,k+1,m} - (Q_2/\hat{Q}_1)_{j,k-1,m}] \\ &+ \frac{1}{2}\zeta_z[(Q_2/\hat{Q}_1)_{j,k,m+1} - (Q_2/\hat{Q}_1)_{j,k,m-1}] \\ &- \frac{1}{2}\xi_x[(Q_4/\hat{Q}_1)_{j+1,k,m} - (Q_4/\hat{Q}_1)_{j-1,k,m}] \\ &- \frac{1}{2}\eta_x[(Q_4/\hat{Q}_1)_{j,k+1,m} - (Q_4/\hat{Q}_1)_{j,k-1,m}] \\ &- \frac{1}{2}\zeta_x[(Q_4/\hat{Q}_1)_{j,k,m+1} - (Q_4/\hat{Q}_1)_{j,k,m-1}] \end{aligned}$$

$$\begin{aligned}
S_3(\mathbf{Q}) = &\frac{1}{2}\xi_x[(Q_3/\hat{Q}_1)_{j+1,k,m} - (Q_3/\hat{Q}_1)_{j-1,k,m}] \\
&+ \frac{1}{2}\eta_x[(Q_3/\hat{Q}_1)_{j,k+1,m} - (Q_3/\hat{Q}_1)_{j,k-1,m}] \\
&+ \frac{1}{2}\zeta_x[(Q_3/\hat{Q}_1)_{j,k,m+1} - (Q_3/\hat{Q}_1)_{j,k,m-1}] \\
&- \frac{1}{2}\xi_y[(Q_2/\hat{Q}_1)_{j+1,k,m} - (Q_2/\hat{Q}_1)_{j-1,k,m}] \\
&- \frac{1}{2}\eta_y[(Q_2/\hat{Q}_1)_{j,k+1,m} - (Q_2/\hat{Q}_1)_{j,k-1,m}] \\
&- \frac{1}{2}\zeta_y[(Q_2/\hat{Q}_1)_{j,k,m+1} - (Q_2/\hat{Q}_1)_{j,k,m-1}]
\end{aligned}$$

Magnitude of the vorticity is written as: $\|\mathbf{S}\| = \sqrt{\mathbf{S_1^2 + S_2^2 + S_3^2}}$

Next, we take a derivative of S:

$$\left(\frac{\partial\|\mathbf{S}\|}{\partial\mathbf{Q}}\right)_{j,k,m} = \frac{1}{\sqrt{S_1^2 + S_2^2 + S_3^2}}\left(S_1\frac{\partial S_1}{\partial\mathbf{Q}} + S_2\frac{\partial S_2}{\partial\mathbf{Q}} + S_3\frac{\partial S_3}{\partial\mathbf{Q}}\right)$$

The derivatives of the three components will considered separately in the next three sections.

B.1 Vorticity Differention Terms for S_1

$$\left(\frac{\partial S_1}{\partial Q_1}\right)_{j+1,k,m} = -\frac{1}{2}\left[\xi_y\left(\frac{Q_4}{\hat{Q_1}^2}\right)_{j+1,k,m} - \xi_z\left(\frac{Q_3}{\hat{Q_1}^2}\right)_{j+1,k,m}\right]$$

$$\left(\frac{\partial S_1}{\partial Q_1}\right)_{j,k+1,m} = -\frac{1}{2}\left[\eta_y\left(\frac{Q_4}{\hat{Q_1}^2}\right)_{j,k+1,m} - \eta_z\left(\frac{Q_3}{\hat{Q_1}^2}\right)_{j,k+1,m}\right]$$

$$\left(\frac{\partial S_1}{\partial Q_1}\right)_{j,k,m+1} = -\frac{1}{2}\left[\zeta_y\left(\frac{Q_4}{\hat{Q_1}^2}\right)_{j,k,m+1} - \zeta_z\left(\frac{Q_3}{\hat{Q_1}^2}\right)_{j,k,m+1}\right]$$

$$\left(\frac{\partial S_1}{\partial Q_1}\right)_{j-1,k,m} = \frac{1}{2}\left[\xi_y\left(\frac{Q_4}{\hat{Q_1}^2}\right)_{j-1,k,m} - \xi_z\left(\frac{Q_3}{\hat{Q_1}^2}\right)_{j-1,k,m}\right]$$

$$\left(\frac{\partial S_1}{\partial Q_1}\right)_{j,k-1,m} = \frac{1}{2}\left[\eta_y\left(\frac{Q_4}{\hat{Q_1}^2}\right)_{j,k-1,m} - \eta_z\left(\frac{Q_3}{\hat{Q_1}^2}\right)_{j,k-1,m}\right]$$

$$\left(\frac{\partial S_1}{\partial Q_1}\right)_{j,k,m-1} = \frac{1}{2}\left[\zeta_y\left(\frac{Q_4}{\hat{Q_1}^2}\right)_{j,k,m-1} - \zeta_z\left(\frac{Q_3}{\hat{Q_1}^2}\right)_{j,k,m-1}\right]$$

$$\left(\frac{\partial S_1}{\partial Q_3}\right)_{j+1,k,m} = -\frac{1}{2}\xi_z\left(\frac{1}{\hat{Q_1}}\right)_{j+1,k,m}$$

$$\left(\frac{\partial S_1}{\partial Q_3}\right)_{j,k+1,m} = -\frac{1}{2}\eta_z\left(\frac{1}{\hat{Q_1}}\right)_{j,k+1,m}$$

$$\left(\frac{\partial S_1}{\partial Q_3}\right)_{j,k,m+1} = -\frac{1}{2}\zeta_z\left(\frac{1}{\hat{Q_1}}\right)_{j,k,m+1}$$

$$\left(\frac{\partial S_1}{\partial Q_3}\right)_{j-1,k,m} = \frac{1}{2}\xi_z\left(\frac{1}{\hat{Q_1}}\right)_{j-1,k,m}$$

$$\left(\frac{\partial S_1}{\partial Q_3}\right)_{j,k-1,m} = \frac{1}{2}\eta_z\left(\frac{1}{\hat{Q_1}}\right)_{j,k-1,m}$$

$$\left(\frac{\partial S_1}{\partial Q_3}\right)_{j,k,m-1} = \frac{1}{2}\zeta_z\left(\frac{1}{\hat{Q_1}}\right)_{j,k,m-1}$$

$$\left(\frac{\partial S_1}{\partial Q_4}\right)_{j+1,k,m} = \frac{1}{2}\xi_y\left(\frac{1}{\hat{Q_1}}\right)_{j+1,k,m}$$

$$\left(\frac{\partial S_1}{\partial Q_4}\right)_{j,k+1,m} = \frac{1}{2}\eta_y\left(\frac{1}{\hat{Q}_1}\right)_{j,k+1,m}$$

$$\left(\frac{\partial S_1}{\partial Q_4}\right)_{j,k,m+1} = \frac{1}{2}\zeta_y\left(\frac{1}{\hat{Q}_1}\right)_{j,k,m+1}$$

$$\left(\frac{\partial S_1}{\partial Q_4}\right)_{j-1,k,m} = -\frac{1}{2}\xi_y\left(\frac{1}{\hat{Q}_1}\right)_{j-1,k,m}$$

$$\left(\frac{\partial S_1}{\partial Q_4}\right)_{j,k-1,m} = -\frac{1}{2}\eta_y\left(\frac{1}{\hat{Q}_1}\right)_{j,k-1,m}$$

$$\left(\frac{\partial S_1}{\partial Q_4}\right)_{j,k,m-1} = -\frac{1}{2}\zeta_y\left(\frac{1}{\hat{Q}_1}\right)_{j,k,m-1}$$

$$\frac{\partial S_1}{\partial \mathbf{Q}} = \left[\begin{array}{cccccc} \frac{\partial S_1}{\partial Q_1} & \frac{\partial S_1}{\partial Q_2} & \frac{\partial S_1}{\partial Q_3} & \frac{\partial S_1}{\partial Q_4} & \frac{\partial S_1}{\partial Q_5} & \frac{\partial S_1}{\partial Q_6} \end{array}\right]$$

$$\frac{\partial S_1}{\partial \mathbf{Q}} = \left[\begin{array}{cccccc} \frac{\partial S_1}{\partial Q_1} & 0 & \frac{\partial S_1}{\partial Q_3} & \frac{\partial S_1}{\partial Q_4} & 0 & 0 \end{array}\right]$$

B.2 Vorticity Differention Terms for S_2

$$\left(\frac{\partial S_1}{\partial Q_1}\right)_{j+1,k,m} = -\frac{1}{2}\left[\xi_z\left(\frac{Q_2}{\hat{Q_1}^2}\right)_{j+1,k,m} - \xi_x\left(\frac{Q_4}{\hat{Q_1}^2}\right)_{j+1,k,m}\right]$$

$$\left(\frac{\partial S_1}{\partial Q_1}\right)_{j,k+1,m} = -\frac{1}{2}\left[\eta_z\left(\frac{Q_2}{\hat{Q_1}^2}\right)_{j,k+1,m} - \eta_x\left(\frac{Q_4}{\hat{Q_1}^2}\right)_{j,k+1,m}\right]$$

$$\left(\frac{\partial S_1}{\partial Q_1}\right)_{j,k,m+1} = -\frac{1}{2}\left[\zeta_z\left(\frac{Q_2}{\hat{Q_1}^2}\right)_{j,k,m+1} - \zeta_x\left(\frac{Q_4}{\hat{Q_1}^2}\right)_{j,k,m+1}\right]$$

$$\left(\frac{\partial S_1}{\partial Q_1}\right)_{j-1,k,m} = \frac{1}{2}\left[\xi_z\left(\frac{Q_2}{\hat{Q_1}^2}\right)_{j-1,k,m} - \xi_x\left(\frac{Q_4}{\hat{Q_1}^2}\right)_{j-1,k,m}\right]$$

$$\left(\frac{\partial S_1}{\partial Q_1}\right)_{j,k-1,m} = \frac{1}{2}\left[\eta_z\left(\frac{Q_2}{\hat{Q_1}^2}\right)_{j,k-1,m} - \eta_x\left(\frac{Q_4}{\hat{Q_1}^2}\right)_{j,k-1,m}\right]$$

$$\left(\frac{\partial S_1}{\partial Q_1}\right)_{j,k,m-1} = \frac{1}{2}\left[\zeta_z\left(\frac{Q_2}{\hat{Q_1}^2}\right)_{j,k,m-1} - \zeta_x\left(\frac{Q_4}{\hat{Q_1}^2}\right)_{j,k,m-1}\right]$$

$$\left(\frac{\partial S_2}{\partial Q_2}\right)_{j+1,k,m} = \frac{1}{2}\xi_z\left(\frac{1}{\hat{Q_1}}\right)_{j+1,k,m}$$

$$\left(\frac{\partial S_2}{\partial Q_2}\right)_{j,k+1,m} = \frac{1}{2}\eta_z\left(\frac{1}{\hat{Q_1}}\right)_{j,k+1,m}$$

$$\left(\frac{\partial S_2}{\partial Q_2}\right)_{j,k,m+1} = \frac{1}{2}\zeta_z\left(\frac{1}{\hat{Q_1}}\right)_{j,k,m+1}$$

$$\left(\frac{\partial S_2}{\partial Q_2}\right)_{j-1,k,m} = -\frac{1}{2}\xi_z\left(\frac{1}{\hat{Q_1}}\right)_{j-1,k,m}$$

$$\left(\frac{\partial S_2}{\partial Q_2}\right)_{j,k-1,m} = -\frac{1}{2}\eta_z\left(\frac{1}{\hat{Q_1}}\right)_{j,k-1,m}$$

$$\left(\frac{\partial S_2}{\partial Q_2}\right)_{j,k,m-1} = -\frac{1}{2}\zeta_z\left(\frac{1}{\hat{Q_1}}\right)_{j,k,m-1}$$

$$\left(\frac{\partial S_2}{\partial Q_4}\right)_{j+1,k,m} = -\frac{1}{2}\xi_x\left(\frac{1}{\hat{Q_1}}\right)_{j+1,k,m}$$

$$\left(\frac{\partial S_2}{\partial Q_4}\right)_{j,k+1,m} = -\frac{1}{2}\eta_x\left(\frac{1}{\hat{Q}_1}\right)_{j,k+1,m}$$

$$\left(\frac{\partial S_2}{\partial Q_4}\right)_{j,k,m+1} = -\frac{1}{2}\zeta_x\left(\frac{1}{\hat{Q}_1}\right)_{j,k,m+1}$$

$$\left(\frac{\partial S_2}{\partial Q_4}\right)_{j-1,k,m} = \frac{1}{2}\xi_x\left(\frac{1}{\hat{Q}_1}\right)_{j-1,k,m}$$

$$\left(\frac{\partial S_2}{\partial Q_4}\right)_{j,k-1,m} = \frac{1}{2}\eta_x\left(\frac{1}{\hat{Q}_1}\right)_{j,k-1,m}$$

$$\left(\frac{\partial S_2}{\partial Q_4}\right)_{j,k,m-1} = \frac{1}{2}\zeta_x\left(\frac{1}{\hat{Q}_1}\right)_{j,k,m-1}$$

$$\frac{\partial S_2}{\partial \mathbf{Q}} = \left[\begin{array}{cccccc} \frac{\partial S_2}{\partial Q_1} & \frac{\partial S_2}{\partial Q_2} & \frac{\partial S_2}{\partial Q_3} & \frac{\partial S_2}{\partial Q_4} & \frac{\partial S_2}{\partial Q_5} & \frac{\partial S_2}{\partial Q_6} \end{array}\right]$$

$$\frac{\partial S_2}{\partial \mathbf{Q}} = \left[\begin{array}{cccccc} \frac{\partial S_2}{\partial Q_1} & \frac{\partial S_2}{\partial Q_2} & 0 & \frac{\partial S_2}{\partial Q_4} & 0 & 0 \end{array}\right]$$

B.3 Vorticity Differention Terms for S_3

$$\left(\frac{\partial S_3}{\partial Q_1}\right)_{j+1,k,m} = -\frac{1}{2}\left[\xi_x\left(\frac{Q_3}{\hat{Q}_1^{\,2}}\right)_{j+1,k,m} - \xi_y\left(\frac{Q_2}{\hat{Q}_1^{\,2}}\right)_{j+1,k,m}\right]$$

$$\left(\frac{\partial S_3}{\partial Q_1}\right)_{j,k+1,m} = -\frac{1}{2}\left[\eta_x\left(\frac{Q_3}{\hat{Q}_1^{\,2}}\right)_{j,k+1,m} - \eta_y\left(\frac{Q_2}{\hat{Q}_1^{\,2}}\right)_{j,k+1,m}\right]$$

$$\left(\frac{\partial S_3}{\partial Q_1}\right)_{j,k,m+1} = -\frac{1}{2}\left[\zeta_x\left(\frac{Q_3}{\hat{Q}_1^{\,2}}\right)_{j,k,m+1} - \zeta_y\left(\frac{Q_2}{\hat{Q}_1^{\,2}}\right)_{j,k,m+1}\right]$$

$$\left(\frac{\partial S_3}{\partial Q_1}\right)_{j-1,k,m} = \frac{1}{2}\left[\xi_x\left(\frac{Q_3}{\hat{Q}_1^{\,2}}\right)_{j-1,k,m} - \xi_y\left(\frac{Q_2}{\hat{Q}_1^{\,2}}\right)_{j-1,k,m}\right]$$

$$\left(\frac{\partial S_3}{\partial Q_1}\right)_{j,k-1,m} = \frac{1}{2}\left[\eta_x\left(\frac{Q_3}{\hat{Q}_1^{\,2}}\right)_{j,k-1,m} - \eta_y\left(\frac{Q_2}{\hat{Q}_1^{\,2}}\right)_{j,k-1,m}\right]$$

$$\left(\frac{\partial S_3}{\partial Q_1}\right)_{j,k,m-1} = \frac{1}{2}\left[\zeta_x\left(\frac{Q_3}{\hat{Q}_1^{\,2}}\right)_{j,k,m-1} - \zeta_y\left(\frac{Q_2}{\hat{Q}_1^{\,2}}\right)_{j,k,m-1}\right]$$

$$\left(\frac{\partial S_3}{\partial Q_2}\right)_{j+1,k,m} = -\frac{1}{2}\xi_y\left(\frac{1}{\hat{Q}_1}\right)_{j+1,k,m}$$

$$\left(\frac{\partial S_3}{\partial Q_2}\right)_{j,k+1,m} = -\frac{1}{2}\eta_y\left(\frac{1}{\hat{Q}_1}\right)_{j,k+1,m}$$

$$\left(\frac{\partial S_3}{\partial Q_2}\right)_{j,k,m+1} = -\frac{1}{2}\zeta_y\left(\frac{1}{\hat{Q}_1}\right)_{j,k,m+1}$$

$$\left(\frac{\partial S_3}{\partial Q_2}\right)_{j-1,k,m} = \frac{1}{2}\xi_y\left(\frac{1}{\hat{Q}_1}\right)_{j-1,k,m}$$

$$\left(\frac{\partial S_3}{\partial Q_2}\right)_{j,k-1,m} = \frac{1}{2}\eta_y\left(\frac{1}{\hat{Q}_1}\right)_{j,k-1,m}$$

$$\left(\frac{\partial S_3}{\partial Q_2}\right)_{j,k,m-1} = \frac{1}{2}\zeta_y\left(\frac{1}{\hat{Q}_1}\right)_{j,k,m-1}$$

$$\left(\frac{\partial S_3}{\partial Q_3}\right)_{j+1,k,m} = \frac{1}{2}\xi_x\left(\frac{1}{\hat{Q}_1}\right)_{j+1,k,m}$$

$$\left(\frac{\partial S_3}{\partial Q_3}\right)_{j,k+1,m} = \frac{1}{2}\eta_x\left(\frac{1}{\hat{Q}_1}\right)_{j,k+1,m}$$

$$\left(\frac{\partial S_3}{\partial Q_3}\right)_{j,k,m+1} = \frac{1}{2}\zeta_x\left(\frac{1}{\hat{Q}_1}\right)_{j,k,m+1}$$

$$\left(\frac{\partial S_3}{\partial Q_4}\right)_{j-1,k,m} = -\frac{1}{2}\xi_x\left(\frac{1}{\hat{Q}_1}\right)_{j-1,k,m}$$

$$\left(\frac{\partial S_3}{\partial Q_4}\right)_{j,k-1,m} = -\frac{1}{2}\eta_x\left(\frac{1}{\hat{Q}_1}\right)_{j,k-1,m}$$

$$\left(\frac{\partial S_3}{\partial Q_4}\right)_{j,k,m-1} = -\frac{1}{2}\zeta_x\left(\frac{1}{\hat{Q}_1}\right)_{j,k,m-1}$$

$$\frac{\partial S_3}{\partial \mathbf{Q}} = \left[\begin{array}{cccccc} \frac{\partial S_3}{\partial Q_1} & \frac{\partial S_3}{\partial Q_2} & \frac{\partial S_3}{\partial Q_3} & \frac{\partial S_3}{\partial Q_4} & \frac{\partial S_3}{\partial Q_5} & \frac{\partial S_3}{\partial Q_6} \end{array}\right]$$

$$\frac{\partial S_3}{\partial \mathbf{Q}} = \left[\begin{array}{cccccc} \frac{\partial S_3}{\partial Q_1} & \frac{\partial S_3}{\partial Q_2} & \frac{\partial S_3}{\partial Q_3} & 0 & 0 & 0 \end{array}\right]$$

Appendix C

TYPHOON Input Files

C.1 Subsonic Turbulent Flow

The following input file is used in the turbulent grid convergence study in Section 6.2.1:

```
MP-OPT
  1
WEIGHT  FSMACH  CL_TAR  WFL  CD_TAR    RE      DVALFA   ALPHA
  1.0   0.6998   0.25   1.0  0.023   11.74e6   FALSE    0.04d0
&OPTIMA
   OPT_METH=1
&END
&TYPHOON
   MIN_RES = 1E-15,

   VISCOUS = true, VISXI = true,  VISETA = true,
   VISZETA = true, VISJAC = true,  VISCROSS = false,

   TURBULNT = true, WRITETURB=true,
   FRECHET = false, ASHFORD = false,

   RESTART = false, BODYBC = true, FARBC = true,

   INORD = 2, IREORD = 2, IREAD=2, NHALO = 2,

   TURBDELAY = 1,

   IDMODEL=1,
   DIS2X   = 0.00,  DIS4X = 0.02,
   DIS2Y   = 0.00,  DIS4Y = 0.02,
   DIS2Z   = 0.00,  DIS4Z = 0.02,

   ISTREAM = 1, IGROUND = 3,

   NK_ITS = 1000,   BLUNT = FALSE,  NK_LFIL  = 1, NK_PFRZ  = 300,
   NK_PDC = 5.d0, NK_IMGMR = 40, NK_ITGMR = 100, GMR_TOL = 1d-2,
   NK_TIME = 4, ISTARTUP = 2, RD_TOL = 10.0, STRTIT = 1,
   RETINF = 0.001,
&END

&EXTRA
   grid_file_prefix = 'grid',
   output_file_prefix = 'results',
   restart_file_prefix = 'results'
&END
```

C.2 Transonic Turbulent Flow

The following grid file was used for the transonic turbulent flows in the parametric optimization study in Section 5.2:

```
MP-OPT
  1
WEIGHT  FSMACH  CL_TAR  WFL  CD_TAR    RE      DVALFA   ALPHA
  1.0    0.84    0.25    1.0  0.023    2.88e6   FALSE    3.0d0
&OPTIMA
   OPT_METH=1
&END
&TYPHOON
   MIN_RES =  1E-15,

   VISCOUS = true, VISXI = true,  VISETA = true,
   VISZETA = true, VISJAC = true,  VISCROSS = false,

   TURBULNT = true, WRITETURB=true,
   FRECHET = false, ASHFORD = false,
   RESTART = false, BODYBC = true, FARBC = true,

   INORD = 2, IREORD = 2, IREAD=2, NHALO = 2,

   TURBDELAY = 1,

   IDMODEL=1,
   DIS2X   = 1.00,  DIS4X = 0.02,
   DIS2Y   = 1.00,  DIS4Y = 0.02,
   DIS2Z   = 1.00,  DIS4Z = 0.02,

   ISTREAM = 1, IGROUND = 3,

   NK_ITS = 1000,   BLUNT = FALSE,  NK_LFIL  = 1, NK_PFRZ  = 300,
   NK_PDC = 3.d0, NK_IMGMR = 40, NK_ITGMR = 100, GMR_TOL = 1d-2,
   NK_TIME = 3, ISTARTUP = 2, RD_TOL = 1000, STRTIT = 1,
   RETINF = 0.001,
&END

&EXTRA
   grid_file_prefix = 'grid',
   output_file_prefix = 'results',
   restart_file_prefix = 'results'
&END
```