2004

Numerical methods for the unsteady incompressible Navier-Stokes equations and their application to the Direct Numerical Simulation of turbulent flows

Frank Herbert Muldoon
Louisiana State University and Agricultural and Mechanical College, fmuldoo@me.lsu.edu

Follow this and additional works at: https://digitalcommons.lsu.edu/gradschool_dissertations
Part of the Mechanical Engineering Commons

Recommended Citation
https://digitalcommons.lsu.edu/gradschool_dissertations/1360

This Dissertation is brought to you for free and open access by the Graduate School at LSU Digital Commons. It has been accepted for inclusion in LSU Doctoral Dissertations by an authorized graduate school editor of LSU Digital Commons. For more information, please contact gradetd@lsu.edu.
NUMERICAL METHODS FOR THE UNSTEADY INCOMPRESSIBLE NAVIER-STOKES EQUATIONS AND THEIR APPLICATION TO THE DIRECT NUMERICAL SIMULATION OF TURBULENT FLOWS

A Dissertation

submitted to the Graduate Faculty of the Louisiana State University and Agricultural and Mechanical College in partial fulfillment of the requirements for the degree of Doctor of Philosophy

in
The Department of Mechanical Engineering

by
Frank Herbert Muldoon
B.S., Mechanical Engineering, Louisiana State University, 1996
May, 2004
Acknowledgements

I would like to thank my family who have been supportive of me in this work. I appreciate the support and advice of my advisor Professor Sumanta Acharya. I would like to thank Professor Robert Dorroh, Professor John Tyler, Dr. Dimitris Nikitopoulos and Professor Arlo Landolt for serving on my committee. Also, the many discussions I have had with fellow students Raymond Jones and Mayank Tyagi have been quite helpful. I would also like to thank Dr. Dimitris Nikitopoulos for helping and advising me in the field of computational fluid dynamics as an undergraduate.

This work received support from the Office Of Naval Research (Propulsion Program, Gabriel Roy, Program Officer) and the High Performance Computing Office (Annette May S/AAA), the Graduate Student Research Program administered by NASA-Stennis, NASA-Glenn, AFOSR, the Board of Regents of Louisiana State University in conjunction with the Louisiana Educational Quality Support Fund and the Biological Computation and Visualization Center (BCVC).

The simulations were run on an IBM SP2 (LSU), IBM SP4 (ARSC), IBM SP3 (NAVO), IBM SP4 (NAVO), Linux PIII cluster (Maui Supercomputing Center), Linux Xeon clusters (SuperMike and Helix at LSU) a dual processor Compaq Dec Alpha at the Computational Fluid Dynamics Lab at Louisiana State University and a Linux PIII single processor machine.
# Table of Contents

Acknowledgements ..................................................................................................................................................................................... iii

List of Tables ................................................................................................................................................................................................... viii

List of Figures ................................................................................................................................................................................................... ix

Abstract .............................................................................................................................................................................................................. xix

Chapter 1 Introduction ...................................................................................................................................................................... 1

Chapter 2 Governing Equations and Discretization .................................................................................................... 5
  2.1 Governing Equations ................................................................................................................................................................ 5
  2.2 Linear Finite Difference Schemes .................................................................................................................................. 6
  2.3 Monotonic Convection Schemes ..................................................................................................................................... 6
  2.3.1 Non-monotonic Schemes .............................................................................................................................................. 7
  2.3.2 Monotonic High Order Scheme ................................................................................................................................. 8
  2.3.3 High Order Time Integration ................................................................................................................................... 10
  2.3.4 Clipping ............................................................................................................................................................................. 11
  2.3.5 Results of One-dimensional Profile ............................................................................................................................ 12
  2.3.6 Results of Film Cooling Problem .............................................................................................................................. 15

Chapter 3 Pressure Velocity Coupling ................................................................................................................................ 21
  3.1 Introduction .................................................................................................................................................................................. 21
  3.2 Nomenclature .............................................................................................................................................................................. 24
  3.3 Governing Equations and Discretization ............................................................................................................... 25
    3.3.1 Governing Equations ................................................................................................................................................. 25
    3.3.2 Spatial Discretization ...................................................................................................................................................... 25
    3.3.3 Temporal Discretization ........................................................................................................................................... 26
  3.4 Artificial Compressibility .................................................................................................................................................. 26
    3.4.1 Velocity ........................................................................................................................................................................... 27
    3.4.2 Pressure ........................................................................................................................................................................ 29
    3.4.3 Algorithm ................................................................................................................................................................... 29
  3.5 SCGS ................................................................................................................................................................................................. 30
  3.6 New Algorithm, SCGS-PP ............................................................................................................................................... 33
    3.6.1 Pressure ........................................................................................................................................................................ 33
    3.6.2 Velocity ........................................................................................................................................................................ 35
    3.6.3 Algorithm ................................................................................................................................................................... 35
  3.7 New Algorithm, SCGS-PPV ........................................................................................................................................... 35
    3.7.1 Algorithm ................................................................................................................................................................... 38
  3.8 Colored SCGS .............................................................................................................................................................................. 38
    3.8.1 Implicit Convection Diffusion Time Integration ......................................................................................................... 39
  3.9 Fourier Mode Analysis ........................................................................................................................................................ 40
    3.9.1 Model Problem ............................................................................................................................................................ 41
    3.9.2 Results of the Model Problem .................................................................................................................................... 42
Chapter 4  Immersed Boundary Method ................................................................. 74
  3.10 Computational Work ........................................................................................................ 46
  3.11 Results .............................................................................................................................. 49
    3.11.1 DNS of Channel Flow ................................................................. 49
    3.11.2 DNS of Driven Cavity ................................................................. 61
    3.11.3 DNS of Flow Over a Backstep .................................................. 65
  3.12 Conclusion ...................................................................................................................... 69

Chapter 5  Tetra .............................................................................................................. 100
  5.1 Parallelization .................................................................................................................. 100
    5.1.1 HPF .................................................................................... 100
    5.1.2 MPI .............................................................................. 102
  5.2 Performance ................................................................................................................... 103
  5.3 I/O ................................................................................................................................. 106
  5.4 Lagrangian Particles ....................................................................................................... 107
    5.4.1 Parallelization ........................................................................ 108
    5.4.2 Implementation ................................................................... 109
  5.5 Code Validation .............................................................................................................. 110
    5.5.1 Constructing Artificial Flow Fields ........................................ 110
  5.6 Varying Grid Spacing ..................................................................................................... 113
    5.6.1 Order in the Solution .................................................................. 113
    5.6.2 Order in the Residual .............................................................. 114

Chapter 6  Budgets of $k-\varepsilon$ Equations .............................................................. 119
  6.1 Introduction ..................................................................................................................... 119
  6.2 Turbulence Modeling ....................................................................................................... 120
    6.2.1 Exact $k-\varepsilon$ Equations ....................................................... 121
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 3.1</td>
<td>Number of floating point operations</td>
<td>48</td>
</tr>
<tr>
<td>Table 3.2</td>
<td>Wall clock time</td>
<td>48</td>
</tr>
<tr>
<td>Table 3.3</td>
<td>Channel time step and numerical parameters, explicit time integration</td>
<td>52</td>
</tr>
<tr>
<td>Table 3.4</td>
<td>Channel time step and numerical parameters, implicit time integration</td>
<td>57</td>
</tr>
<tr>
<td>Table 3.5</td>
<td>Driven cavity time step and numerical parameters, explicit time integration</td>
<td>62</td>
</tr>
<tr>
<td>Table 3.6</td>
<td>Driven cavity time step and numerical parameters, implicit time integration</td>
<td>64</td>
</tr>
<tr>
<td>Table 5.1</td>
<td>Machines to which Tetra has been ported</td>
<td>105</td>
</tr>
<tr>
<td>Table 5.2</td>
<td>Performance of Tetra on different machines</td>
<td>106</td>
</tr>
<tr>
<td>Table 5.3</td>
<td>Grids used in performance test of Tetra to keep the number of finite difference points the same for each process</td>
<td>106</td>
</tr>
<tr>
<td>Table 6.1</td>
<td>Channel flow grid spacing and time step table</td>
<td>127</td>
</tr>
<tr>
<td>Table B.1</td>
<td>Running average terms in $k$ equation</td>
<td>276</td>
</tr>
<tr>
<td>Table B.2</td>
<td>Running average terms in $\varepsilon$ equation</td>
<td>278</td>
</tr>
</tbody>
</table>
List of Figures

Figure 2.1 Clipping ................................................................. 12
Figure 2.2 Profile convected 100 grid points at a CFL number of .125 14
Figure 2.3 Profile convected 500 grid points at a CFL number of .125 15
Figure 2.4 RMS error of Eq. A.35 at CFL number of .125 16
Figure 2.5 Maximum scalar as a function of time, (a) 286 × 80 × 70 grid, (b) 574 × 161 × 142 grid 18
Figure 2.6 Maximum kinetic energy as a function of time, (a) 286 × 80 × 70 grid, (b) 574 × 161 × 142 grid 18
Figure 2.7 Maximum scalar as a function of time, (a) 286 × 80 × 70 grid, (b) 574 × 161 × 142 grid 19
Figure 2.8 Maximum kinetic energy as a function of time, (a) 286 × 80 × 70 grid, (b) 574 × 161 × 142 grid 19
Figure 2.9 Maximum scalar as a function of time, (a) 286 × 80 × 70 grid, (b) 574 × 161 × 142 grid 20
Figure 2.10 Monotonic version of Eq. A.35 on different grids 20
Figure 3.1 Staggered grid (circled points show the points used in SCGS-PPV) 36
Figure 3.2 Colored grid scheme .............................................. 39
Figure 3.3 Isosurfaces of ρ_u on a 16 × 16 × 16 grid 43
Figure 3.4 Isosurfaces of ρ_p on a 16 × 16 × 16 grid 44
Figure 3.5 Maximum and average smoothing factors on a 16 × 16 × 16 grid 45
Figure 3.6 Residual history on a 16 × 16 × 16 grid 47
Figure 3.7 Grid that colored SCGS cannot be used on 47
Figure 3.8 Channel schematic .................................................. 50
Figure 3.9 Average residual of the continuity equation, channel flow, ∆t = .00005 53
Figure 3.10 Average residual of the continuity equation, channel flow, ∆t = .00015 54
Figure 3.11  Mean $v$, channel flow, $\Delta t = .00005$.  

Figure 3.12  Channel flow, explicit scheme, spatially averaged $v$ in $xz$ plane, (a) Artificial compressibility, (b) SCGS-PP, SCGS-PPV and colored SCGS.  

Figure 3.13  Average residual of the continuity equation, channel flow, $\Delta t = .0005$.  

Figure 3.14  Average residual of the continuity equation, channel flow, $\Delta t = .0015$.  

Figure 3.15  Mean $v$, channel flow, $\Delta t = .0005$.  

Figure 3.16  Channel flow, implicit scheme, spatially averaged $v$ in $xz$ plane, (a) Artificial compressibility, (b) SCGS-PP, SCGS-PPV and colored SCGS.  

Figure 3.17  Driven cavity geometry and boundary conditions.  

Figure 3.18  Average residual of the continuity equation, driven cavity, $\Delta t = .0003$.  

Figure 3.19  Mass flow through $yz$ plane at $x = .5h$, driven cavity.  

Figure 3.20  Average residual of the continuity equation, driven cavity, $\Delta t = .003$.  

Figure 3.21  Backstep schematic.  

Figure 3.22  Averaged $u$ velocity at different $x/h$ locations.  

Figure 3.23  Averaged $v$ velocity at different $x/h$ locations.  

Figure 3.24  $\overline{u'v'}$ at different $x/h$ locations.  

Figure 3.25  $\overline{u'u'}$ at different $x/h$ locations.  

Figure 4.1  Staggered grid.  

Figure 4.2  Immersed boundary.  

Figure 4.3  Stencil for interpolation of immersed boundary points.  

Figure 4.4  Notation used for equations centered at mass conservation cells.  

Figure 4.5  Notation used for equations centered at velocity points.  

Figure 4.6  Mass conservation cells that may need to be constraints.  

Figure 4.7  Lagrange multiplier grid arrangement.  

Figure 4.8  Grid coloring scheme.  

Figure 4.9  Staggered grid.
Figure 4.9  Vector fields for flow around sphere at $Re = 100$ ........................................ 92
Figure 4.10 Pressure contours for flow around sphere at $Re = 100$ ................................. 93
Figure 4.11 Cylinder in crossflow, $Re = 40$ (even grid) .................................................. 98
Figure 5.1  Domain decomposition ............................................................... 103
Figure 5.2  Performance of Tetra on a $128 \times 128 \times 128$ grid, (a) Speedup, (b) wall clock time ................................................................. 106
Figure 5.3 Normalized time of Tetra on Marcellus and Helix, (a) $32 \times 32 \times 32$ base grid, (b) $64 \times 64 \times 64$ base grid ................................................................. 107
Figure 5.4 Wall clock time of Tetra on Marcellus and Helix, (a) $32 \times 32 \times 32$ base grid, (b) $64 \times 64 \times 64$ base grid ................................................................. 107
Figure 5.5 Order of accuracy of discretization of Equations 2.1, 2.2 and 2.3 on an even grid ................................................................. 116
Figure 5.6 Order of accuracy of discretization of Equations 2.1, 2.2 and 2.3 on an uneven grid ................................................................. 117
Figure 5.7 Order of accuracy of discretization of Equations 2.1 and 2.3 on an even grid using monotonic convection scheme ................................................................. 117
Figure 5.8 Order of accuracy of discretization of Equations 2.1 and 2.3 on an uneven grid using monotonic convection scheme ................................................................. 118
Figure 5.9 Different measures of error used to determine the order of accuracy of discretization of the $x$ component of Eq. 2.1 on an uneven grid ................................................................. 118
Figure 6.1 Average residual of continuity equation for channel flow ......................................... 129
Figure 6.2 Average value of scalar in $xz$ plane for channel flow ......................................... 129
Figure 6.3 Mean velocity components and scalar, averaged over different time periods, channel flow, $128 \times 64 \times 64$ grid ................................................................. 131
Figure 6.4 Turbulent kinetic energy and turbulent scalar fluctuations, averaged over different time periods, channel flow, $128 \times 64 \times 64$ grid ................................................................. 131
Figure 6.5 Quantities from the exact $k-\varepsilon$ equations, averaged over different time periods, channel flow, $128 \times 64 \times 64$ grid ................................................................. 132
Figure 6.6 Mean velocity components and scalar on different grids, channel flow .................. 133
Figure 6.7 Non-normal turbulent Reynolds stresses and turbulent kinetic energy on different
grids, channel flow .......................................................... 134

Figure 6.8  Quantities from the exact $k−\varepsilon$ equations on different grids, channel flow ...... 135

Figure 6.9  Turbulent diffusion on different grids, channel flow ................................. 135

Figure 6.10 Residual of $k$ equation (Eq. 6.4) on different grids for channel flow .......... 136

Figure 6.11 Residual of $\varepsilon$ equation (Eq. 6.5) on different grids for channel flow .......... 137

Figure 6.12 Residual history (a) average, (b) maximum, channel flow, $128 \times 64 \times 64$ grid ................................. 139

Figure 6.13 Time averaged $u$ (a) and $v$ (b) at different residual levels, channel flow, $128 \times 64 \times 64$ grid ................................................................. 139

Figure 6.14 Time averaged $w$ (a) and scalar (b) at different residual levels, channel flow, $128 \times 64 \times 64$ grid ................................................................. 140

Figure 6.15 $k$ (a) and scalar fluctuations (b) at different residual levels, channel flow, $128 \times 64 \times 64$ grid ................................................................. 140

Figure 6.16 Convection (a) and production of $k$ (b) at different residual levels, channel flow, $128 \times 64 \times 64$ grid ................................................................. 141

Figure 6.17 Turbulent diffusion (a) and residual (b) from exact $k$ equation at different residual levels, channel flow, $128 \times 64 \times 64$ grid ................................................................. 141

Figure 6.18 Convection (a) and production of dissipation (b) at different residual levels, channel flow, $128 \times 64 \times 64$ grid ................................................................. 142

Figure 6.19 Turbulent diffusion (a) and residual (b) from exact dissipation equation at different residual levels, channel flow, $128 \times 64 \times 64$ grid ................................................................. 142

Figure 6.20 $k, \overline{w'u'}$, $\overline{u'w'}$, $\overline{v'w'}$, channel flow ................................................................. 143

Figure 6.21 $s's'$, $s'l'$, $s'v'$, $s'w'$, channel flow ................................................................. 144

Figure 6.22 Eddy viscosity and turbulent Prandtl number, channel flow ................................. 145

Figure 6.23 Eddy viscosity from Eq. 6.8, channel flow ................................................................. 146

Figure 6.24 Eddy viscosity using only $-\overline{u'w'}$ and $\frac{\partial u'}{\partial x_2}$, channel flow ................................. 146

Figure 6.25 Eddy viscosity computed from Eq. 6.8 and modeled eddy viscosity, channel flow ................................. 147

Figure 6.26 Eddy viscosity, wall units, channel flow ................................................................. 147
Figure 6.27 Turbulent Prandtl number from Eq. 6.10, channel flow .......................... 148
Figure 6.28 Simplified turbulent Prandtl number, channel flow ............................ 148
Figure 6.29 Turbulent Prandtl number, wall units, channel flow ............................ 149
Figure 6.30 All terms in exact $k$ equation (Eq. 6.4) computed from DNS, channel flow ................................................................. 150
Figure 6.31 All terms in exact $\varepsilon$ equation (Eq. 6.4) computed from DNS, channel flow ................................................................. 151
Figure 6.32 Production term from Eq. 6.4 vs modeled production term, channel flow .... 152
Figure 6.33 Turbulent diffusion term from Eq. 6.4 vs modeled term, channel flow ......... 153
Figure 6.34 All terms in modeled $k$ equation Eq. 6.6, channel flow ............................ 154
Figure 6.35 Production term from Eq. 6.5 vs modeled term, channel flow ..................... 155
Figure 6.36 Destruction term from Eq. 6.5 vs modeled term, channel flow ..................... 156
Figure 6.37 Turbulent diffusion term from Eq. 6.5 vs modeled term, channel flow ......... 157
Figure 6.38 All terms in modeled $\varepsilon$ equation Eq. 6.7, channel flow ............................ 157
Figure 6.39 Inclined jet in crossflow with delivery tube and plenum, 3D schematic, film cooling jet ................................................................. 163
Figure 6.40 Instantaneous contours of the velocity in the direction of the jet inclination angle .................................................................................. 164
Figure 6.41 Inclined jet in crossflow with prescribed exit conditions 3D schematic, film cooling jet ................................................................. 165
Figure 6.42 Jet exit boundary condition as interpolated onto $574 \times 161 \times 142$ grid, film cooling jet ................................................................. 165
Figure 6.43 One-dimensional spatial energy spectrum in $x$ direction, film cooling jet ........ 167
Figure 6.44 One-dimensional spatial energy spectrum in $z$ direction, film cooling jet ........ 167
Figure 6.45 Instantaneous contours of scalar, film cooling jet ........................................ 168
Figure 6.46 Instantaneous contours of the magnitude of the gradient of scalar, film cooling jet ................................................................. 169
Figure 6.47 Instantaneous contours of the magnitude of the velocity, film cooling jet ........ 169
| Figure 6.67 | Three-dimensional view of production and dissipation terms in exact $k$ equation (Eq. 6.4), film cooling jet | 184 |
| Figure 6.68 | All terms of exact $\varepsilon$ equation (Eq. 6.5), $yz$ plane at $\approx 5d$, film cooling jet | 185 |
| Figure 6.69 | Three-dimensional view of turbulent diffusion terms in exact $\varepsilon$ equation (Eq. 6.5), film cooling jet | 186 |
| Figure 6.70 | Three-dimensional view of production and destruction terms in exact $\varepsilon$ equation (Eq. 6.5), film cooling jet | 186 |
| Figure 6.71 | Production term from exact $k$ equation (Eq. 6.4) vs models, film cooling jet | 188 |
| Figure 6.72 | Turbulent diffusion term from exact $k$ equation (Eq. 6.4) vs models, film cooling jet | 189 |
| Figure 6.73 | Three-dimensional view of residual of modeled $k$ equation (Eq. 6.6), film cooling jet | 189 |
| Figure 6.74 | Three-dimensional view of production term from exact $\varepsilon$ equation (Eq. 6.5) vs modeled term, film cooling jet | 191 |
| Figure 6.75 | Production and turbulent diffusion in exact $\varepsilon$ equation (Eq. 6.5) in $xz$ plane on the wall, film cooling jet | 192 |
| Figure 6.76 | Three-dimensional view of destruction term from exact $\varepsilon$ equation (Eq. 6.5) vs model, film cooling jet | 192 |
| Figure 6.77 | Turbulent diffusion term from exact $\varepsilon$ equation (Eq. 6.5) vs models, film cooling jet | 193 |
| Figure 6.78 | Spatially averaged eddy viscosity, eddy viscosity from standard $k-\varepsilon$ model and damping function, film cooling jet, $574 \times 161 \times 142$ grid | 195 |
| Figure 6.79 | Comparison of $\frac{1}{C_p \varepsilon}$, Eq. 6.13 and Eq. 6.12, film cooling jet | 195 |
| Figure 7.1 | Schematic of mixing jet configuration | 203 |
| Figure 7.2 | Residual history, $410 \times 280 \times 130$ grid, (a) unpulsed (b) $St = .4$ | 206 |
| Figure 7.3 | Scalar history, $410 \times 280 \times 130$ grid, (a) unpulsed (b) $St = .4$ | 207 |
| Figure 7.4 | Mixing jet particle seed locations | 210 |
| Figure 7.5 | Instantaneous scalar, unpulsed jet | 211 |
| Figure 7.6 | Time averaged scalar in $xy$ plane at $z = 0$ | 211 |
Figure 7.7  Time averaged scalar in $yz$ plane at $x = 4d, 6d$ ........................................ 212
Figure 7.8  One dimensional spatial energy spectrum of $u$ (in $x$ direction), $St = .4$ ........ 213
Figure 7.9  Instantaneous isosurface of scalar colored by magnitude of the gradient of the scalar, unpulsed ................................................................. 215
Figure 7.10  Particles released from pink locations in Fig. 7.4, colored by residence time, unpulsed ................................................................. 215
Figure 7.11  Wake vortex visualization by particle traces, unpulsed ............................. 216
Figure 7.12  Wake vortex visualization by particle traces, unpulsed ............................. 217
Figure 7.13  Temporal frequency spectra of $v$ at $x = -.5, z = 0$, unpulsed case .............. 218
Figure 7.14  Asymmetry visualization by particle traces, unpulsed ............................. 219
Figure 7.15  Isosurface of mean scalar colored by $k$, unpulsed ................................. 221
Figure 7.16  Isosurface of mean scalar colored by $\overline{s^2}$, unpulsed ............................. 221
Figure 7.17  Isosurface of $\overline{s^2}$ colored by $\|\nabla s\|$, unpulsed ............................. 221
Figure 7.18  Velocity vectors of mean field colored by mean scalar at $x = 3d, x = 5d, x = 8d, x = 15d$, roughly every 12th vector shown, unpulsed ...................... 222
Figure 7.19  Instantaneous isosurface of scalar colored by magnitude of the gradient of the scalar, $St = .2$ ................................................................. 224
Figure 7.20  Instantaneous isosurface of scalar colored by magnitude of the gradient of the scalar, $St = .4$ ................................................................. 225
Figure 7.21  Instantaneous isosurface of scalar colored by magnitude of the gradient of the scalar, $St = .6$ ................................................................. 225
Figure 7.22  Vortex merging, $St = .2$ ................................................................. 226
Figure 7.23  Particles released from pink locations in Fig. 7.4, colored by residence time, $St = .4$ ................................................................. 226
Figure 7.24  Vortex merging, $St = .6$ ................................................................. 227
Figure 7.25  Wake vortex schematic ................................................................. 227
Figure 7.26  Particles released .05$d$ from wall colored by residence time, (a) $St = .2$, (b) $St = .4$ ................................................................. 229
Figure 7.27 Wake vortex visualization by particle tracking, \( St = .2 \) .......................... 230
Figure 7.28 Wake region visualization by particle tracking, \( St = .2 \) .......................... 231
Figure 7.29 Wake vortex visualization by particle tracking, \( St = .2 \) .......................... 232
Figure 7.30 Wake vortex visualization by particle tracking, \( St = .2 \) .......................... 232
Figure 7.31 Wake region visualization by particle tracking, \( St = .4 \) .......................... 233
Figure 7.32 Wake region visualization by particle traces, \( St = .6 \) .......................... 234
Figure 7.33 Temporal frequency spectra of \( v \) at \( x = -.5, z = 0, St = .2 \) .................. 236
Figure 7.34 Temporal frequency spectra of \( v \) at \( x = -.5, z = 0, St = .4 \) .................. 236
Figure 7.35 Temporal frequency spectra of \( v \) at \( x = -.5, z = 0, St = .6 \) .................. 237
Figure 7.36 Jet trajectory in \( y \), unpulsed and pulsed .......................... 237
Figure 7.37 Jet spreading in \( y \), unpulsed and pulsed .......................... 238
Figure 7.38 Jet trajectory in \( z \), unpulsed and pulsed .......................... 238
Figure 7.39 Jet spreading in \( z \), unpulsed and pulsed .......................... 239
Figure 7.40 \( S_\infty(x) \), unpulsed and pulsed .......................... 239
Figure 7.41 Isosurface of mean scalar colored by \( k \), \( St = .2 \) .......................... 241
Figure 7.42 Isosurface of mean scalar colored by \( k \), \( St = .4 \) .......................... 241
Figure 7.43 Isosurface of mean scalar colored by \( k \), \( St = .6 \) .......................... 241
Figure 7.44 Isosurface of mean scalar colored by \( \overline{s's'} \), \( St = .2 \) .......................... 242
Figure 7.45 Isosurface of mean scalar colored by \( \overline{s's'} \), \( St = .4 \) .......................... 242
Figure 7.46 Isosurface of mean scalar colored by \( \overline{s's'} \), \( St = .6 \) .......................... 242
Figure 7.47 Isosurface of \( \overline{s's'} \) colored by magnitude of the gradient of \( \overline{s} \), \( St = .2 \) ............ 243
Figure 7.48 Isosurface of \( \overline{s's'} \) colored by magnitude of the gradient of \( \overline{s} \), \( St = .4 \) ............ 243
Figure 7.49 Isosurface of \( \overline{s's'} \) colored by magnitude of the gradient of \( \overline{s} \), \( St = .6 \) ............ 243
Figure 7.50 Velocity vectors of mean field colored by mean scalar at \( x = 3d, x = 5d, x = 8d, x = 15d \), roughly every 12th vector shown, \( St = .2 \) ............ 244
Figure 7.51  Velocity vectors of mean field colored by mean scalar at $x = 3d, x = 5d, x = 8d, x = 15d$, roughly every 12th vector shown, $St = .4$ .......................... 245

Figure 7.52  Velocity vectors of mean field colored by mean scalar at $x = 3d, x = 5d, x = 8d, x = 15d$, roughly every 12th vector shown, $St = .6$ .......................... 246

Figure 7.53  $\overline{w}$ at $z = 0$ ................................................................. 247

Figure 7.54  $\overline{w'w'}$ at $z = 0$ ................................................................. 248

Figure 7.55  $\overline{w'v'}$ at $z = 0$ ................................................................. 248

Figure 7.56  $\overline{u'v'}$ at $z = 0$ ................................................................. 248

Figure 7.57  Velocity vectors of mean field colored by mean scalar at $x = 15d$, different symmetry breaking initial conditions, unpulsed ........................................... 250

Figure A.1  Staggered grid schematic .......................................................... 263

Figure A.2  Grid locations of scalar ............................................................. 263

Figure A.3  Second and fourth order interpolation stencils for velocity as needed for the convective terms in Eq. 2.1 ................................................................. 266

Figure A.4  Interpolation stencil for velocity as needed for the convective terms in Eq. 2.1; a fourth order stencil is shown near a non-periodic boundary ...................... 267

Figure A.5  Interpolation stencils used to obtain the velocity at the grid points at which the scalar is solved, (fourth order stencils are shown). .............................. 268

Figure F.1  Wake vortex visualization by particle traces, $St = .2$ ......................... 286

Figure F.2  Wake vortex visualization by particle traces, $St = .4$ ......................... 287

Figure F.3  Wake vortex visualization by particle traces, $St = .4$ ......................... 288
Abstract

Two new methods for the efficient parallel computation of the unsteady incompressible Navier-Stokes equations are presented. Such efficient methods are desired for large scale parallel computations of unsteady turbulent flows such as Direct Numerical Simulations (DNS). The performance of the new methods has a distinct advantage over the artificial compressibility method, in that the methods exhibit robust convergence for a variety of flow problems without extensive need for tuning computational parameters. These methods and others have been implemented in a computer program designed for massively parallel computer architectures, written by the author and used to obtain all results in this work.

A DNS of a film-cooling jet is performed in order to evaluate the accuracy of the modeled expressions in the $k - \varepsilon$ turbulence model. Using the results of the DNS, the terms in the exact and modeled $k - \varepsilon$ equations are computed. These terms are examined to see where the models fail for these flows. DNS budgets for $k$ and $\varepsilon$ in a film cooling jet flow are presented to provide turbulence modelers with information as to where the models used to replace the exact $k - \varepsilon$ equations need improvement for this particular type of flow.

A DNS of a pulsed jet is performed to analyze the effect of external pulsing on the flow structures and the resulting mixing of the jet with the crossflow. As the problem is inherently unsteady, the key to the successful prediction of such flows is the ability to resolve the dynamics of all important flow structures resulting from the interaction of the unsteady pulsed jet with the crossflow. In the present work massless particles are released into the flow at various locations. These particles are colored by their seed locations or residence time, greatly aiding the understanding of the dynamics of the flow. A new origin for the formation of the wake vortices has been discovered for both pulsed and unpulsed jets. Pulsing is shown to drastically change
the jet spreading and penetration and to increase the mixing of the jet with the crossflow. A significant asymmetry affecting primarily the wake vortices has been found for certain cases.
Chapter 1 Introduction

The incompressible Navier-Stokes equations are commonly used to model incompressible turbulent flows (although there is no proof of existence and uniqueness of solutions). Analytical solutions to these equations exist only for extremely simple flows. For the vast majority of flows for which there is no analytical solution, it may be possible to obtain a solution through the use of numerical methods. A numerical solution will require a discretization or gridding of the spatial and time domain in which the Navier-Stokes equations are to be solved. This discretization or gridding must contain spacing on the order of the smallest important features in the flow field. This is known as a Direct Numerical Simulation (DNS), although the term is commonly applied to turbulent flows; this is an arbitrary distinction. A DNS can entail an enormous computational effort when a solution of a turbulent flow is sought. This is a result of the fact that the smallest eddies in a turbulent flow shrink in size as the Reynolds number increases, thereby requiring very small grid spacings in the discretization or gridding. As an example, a DNS of turbulent flow around a car is estimated to require $9 \times 10^{11}$ years of computing time and $2 \times 10^{11}$ Gigabytes of storage. For this reason, the vast majority of solutions to the turbulent flow problems are not obtained by solving the Navier-Stokes equations. Instead, a modified version of the Navier-Stokes equations is used in which the unsteady terms are dropped and the effect of turbulence is modeled. These modified equations are known as the Reynolds Averaged Navier Stokes (RANS) equations. The downside to the Reynolds Averaged Navier-Stokes equations is that they contain a number of correlations of fluctuating (turbulent) quantities that arise from the unsteady terms. These correlations must be modeled in some manner. Unfortunately, it is not possible to derive models for these correlations from first principles. As a result, models for these correlations are developed that make various assumptions about the flow and/or are designed to reproduce a certain type of
simple flow well. When these models are applied to a different or more complicated flow, the results are often grossly in error. The basic problem is that turbulence models are essentially complicated curve fits. In order to develop these complicated curve fits, large amounts of data concerning the structure of turbulence are very useful. Much of this data is difficult or impossible to acquire from experiments. DNS can provide this data and can also be very useful in enabling the comparison of the various terms of the turbulence models with the results of DNS.

The pressure velocity coupling in the solution of the unsteady incompressible Navier Stokes equations has long been a computationally expensive part of the solution process. The basic problem is that of determining an equation to use to solve for the pressure. Within the framework of finite differences there have historically been four approaches. The first and most common have been methods that involve the solution of a Poisson equation for pressure. Such methods include the fractional step method (Chorin, 1968) and the SIMPLE type methods (Patankar, 1980). The second is the method of artificial compressibility (Chorin, 1967). This method involves the addition of an artificial time derivative to the equation set. The third is what is known as direct methods. These methods involve a global coupled solution of the entire discretized system. Direct methods are infeasible due to the computational expense of direct solutions of matrices. The fourth is Symmetric Coupled Gauss Seidel (SCGS) (Vanka, 1986). The DNS of most flows is extremely computationally expensive in terms of spatial and temporal resolution requirements. For this reason, solution algorithms that work on vector, and, more importantly, on distributed memory parallel machines are required. This requirement eliminates SCGS due to its inherently serial nature. This work presents two new methods of solving the Navier Stokes equations on distributed parallel computers.

One of the most difficult and expensive tasks of Computational Fluid Dynamics has always been the generation of a grid around or inside the object being modeled. This has always involved
a high degree of user interaction. If the object is moving or deforming then the necessary regridding is a challenging and computationally expensive problem. The immersed boundary method (Peskin, 1977) (Goldstein et al., 1993) (Fadlun et al., 2000) (Beyer, 1992) has the potential of simplifying these problems associated with the grid. In the present work a new method is presented for insuring a divergence free solution in conjunction with the immersed boundary method.

The methods outlined in the present work have been implemented in a computer program, called Tetra, designed for parallel computers and written by the author. In the present work Tetra is applied to a number of problems, some of which are solved in order to verify the numerical method and its implementation in software on parallel computers. Two problems are solved using Tetra so as to provide insight into different aspects of the flow physics of these particular problems. One of the problems, a film cooling flow, is solved in order to evaluate turbulence models commonly used to solve this flow problem. This flow is of great interest in the design of turbine blades. Current turbulence models used by industry experience large errors in attempting to model this flow. In order to examine the reasons why these models fail, all the correlations that appear in the exact $k - \varepsilon$ equations are computed for this flow. These correlations are then compared with the models used by the standard $k - \varepsilon$ turbulence model to represent them. To help improve turbulence models for this flow, a new damping function is presented for the eddy viscosity that is designed for film cooling flows. The other problem, concerning pulsed jets in a crossflow, is examined so as to understand the effect of pulsing on the flow physics. A dramatic difference in the jet spreading and structure is observed between various pulsing frequencies. This difference is of interest in the context of improving mixing of fuel and air in combustors. Due to the inherent unsteadiness of the pulsing, this can only be examined by a numerical method (such as DNS or Large Eddy Simulation (LES)) that solves the unsteady Navier-Stokes equations.
The results of the DNS are used to make movies of the evolution of the flow structures. With the aid of these movies, the dynamics of the flow resulting from different applied pulsings are examined in depth.
2.1 Governing Equations

Turbulent flows contain a range of spatial and temporal scales. The range of these scales is determined by the Reynolds number ($Re$) and the Prandtl number ($Pr$). A turbulent flow is characterized by the presence of many different sizes of structures or eddies. For instance, a turbulent flow field has large eddies that contain smaller eddies that contain still smaller eddies, etc. This process continues until the smallest sized eddies are reached. At this size the eddies do not break down into smaller eddies, but instead are destroyed by viscous dissipation which turns their kinetic energy into heat. A Direct Numerical Simulation (DNS) is the numerical solution of the mass, momentum and energy conservation equations, with a resolution sufficient to capture all the important scales or sizes of eddies present in the particular flow. This means that the numerical solution of the flow field must have a resolution able to capture the viscous dissipation of the small eddies. The spatial resolution requirements are given as proportional to $Re^{9/4}$ in (Pope, 2000), who also gives an estimate of the number of floating point operations to be proportional to $Re^{3}$. As a result of the computational expense implied by these large exponents, DNS is restricted to flows at relatively small Reynolds numbers.

\[
\begin{align*}
\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} &= -\frac{\partial p}{\partial x_i} + \frac{1}{Re} \frac{\partial^2 u_i}{\partial x_j \partial x_j} \\
\frac{\partial u_i}{\partial x_i} &= 0 \\
\frac{\partial s}{\partial t} + u_j \frac{\partial s}{\partial x_j} &= \frac{1}{Re Pr} \frac{\partial^2 s}{\partial x_j \partial x_j}
\end{align*}
\]

(2.1) (2.2) (2.3)

In the present work, the non-conservative form of the unsteady incompressible three-dimensional Navier-Stokes equations (Eq. 2.1 and Eq. 2.2) is used to model the flow. In addition, an equation describing the evolution of a passive scalar (Eq. 2.3) is solved for flows in which heat transfer or mixing is of interest. These equations are solved without the use of a model
for turbulence. Equations 2.1, 2.2 and 2.3 are solved in non-dimensional form, in which the properties of the fluid and the passive scalar are described by the Reynolds number ($Re$) and the Prandtl number ($Pr$).

2.2 Linear Finite Difference Schemes

See Appendix A for details on the linear finite difference and time integration schemes used in the present work.

2.3 Monotonic Convection Schemes

While high order schemes theoretically have a faster rate of convergence as the grid spacing is reduced when approximating a derivative, they have a property that can adversely affect the obtaining of a solution. This property is known as non-monotonicity. Physically, the process of convection will simply transport a profile or wave around without changing its shape. The physical process of convection should not create any new local maxima or minima in a profile or change those already in existence. A numerical scheme possessing this property is termed a monotonic scheme (Oran & Boris, 2001). The numerical solution of certain physical problems requires that certain variables stay positive. One example is reaction rates that become very unstable if the chemical species concentration becomes negative. Another is compressible flow problems in which the density should never be less than zero. Even if solving an equation governing the evolution of a quantity that has no such restrictions, a numerical convection scheme that is not monotonic can be unstable if it allows the creation of large new maxima or minima. (Godunov, 1959) gives a proof (known as Godunov’s Theorem) that there is no linear scheme that is monotonic and higher than first-order accurate. The essential problem, if linear schemes are used, is that high order accuracy and physically correct solutions are mutually exclusive properties for general flow problems.

Monotonic convection schemes have received extensive attention from a large number of researchers, primarily in the context of compressible flow with shocks and reacting flows. A
number of schemes have been developed that address the issue of high order accuracy and physically permissible solutions including Total Variation Diminishing (TVD) (Harten, 1983), Flux-Corrected Transport (FCT) (Oran & Boris, 2001), Essentially Non-Oscillatory (ENO) (Harten et al., 1987) and Weighted Average Flux (WAF) (Touro, 1999). The present work presents a straightforward optimization-based method of limiting the convective terms to attempt to maintain monotonicity.

2.3.1 Non-monotonic Schemes

2.3.1.1 Upwind Scheme

Consider the one-dimensional wave equation (Eq. 2.4).

\[ \frac{\partial u}{\partial t} + c \frac{\partial u}{\partial x} = 0 \]  

(2.4)

On a discrete grid, local maxima or minima are defined as \( \max(u_{i-1}, u_i, u_{i+1}) \) and \( \min(u_{i-1}, u_i, u_{i+1}) \). Consider the following second-order accurate upwind scheme along with explicit first-order time integration applied to Eq. 2.4, at a CFL number \( \frac{\Delta t c}{\Delta x} \leq 1 \).

\[
\begin{align*}
    u_{i+1}^n & = u_i^n - c \frac{u_{i-2}^n - 4u_{i-1}^n + 3u_{i}^n}{2\Delta x} \Delta t \\
    & = -c \frac{u_{i-2}^n - 4u_{i-1}^n}{2\Delta x} \Delta t + \left(1 - \frac{3\Delta t c}{2\Delta x}\right)u_{i}^n \quad \text{for } c \geq 0
\end{align*}
\]  

(2.5)

Consider the following profile at time \( n \): \( u_{i-2}^n = a + p, \ u_{i-1}^n = a + q, \ u_i^n = a \), where \( p > q > 0 \). This profile has a local maximum of \( a + q \) and a local minimum of \( a \). Apply the following profile to Eq. 2.5 at time \( n \): \( u_{i-2}^n = a + p, \ u_{i-1}^n = a + q, \ u_i^n = a \), where \( p > q > 0 \). This profile has a local maximum of \( a + q \) and a local minimum of \( a \). Solving Eq. 2.5 yields:

\[
    u_{i}^{n+1} = a - \frac{\Delta t c(p - 4q)}{2\Delta x}
\]

If \( p > 4q \) a new local minimum in the solution will occur as the scheme will attempt to make \( u_{i}^{n+1} \leq a \) regardless of the size of \( \Delta t \). If \( 0 < p < 4q \), which may result in a new local maximum, it will be possible to make \( u_{i}^{n+1} \leq a + q \) by making \( \Delta t \) smaller. Therefore it can be seen that, depending on the values of the profile and possibly the time step, this finite difference scheme (Eq. 2.5) may not represent the physical process of convection. The non-monotonic behavior of the
scheme occurs when the profile has large gradients. When the profile is "smooth" this does not occur. In general, the higher the formal accuracy of a scheme, the smaller the range of gradients it can resolve without becoming non-monotonic.

2.3.1.2 Centered Scheme

Discretizing Eq. 2.4 using a sixth-order central difference scheme (Eq. A.35) along with first-order time integration results in:

\[ u^{n+1}_i = u^n_i - c \left(-a_1 u_{i-3} + a_2 u_{i-2} + a_3 u_{i-1} + a_4 u_i + a_5 u_{i+1} + a_6 u_{i+2} + a_7 u_{i+3}\right) \Delta t \]

On an even grid the coefficients in Eq. A.35 have the following values:

\[
\begin{align*}
    a_1 &= -\frac{1}{60\Delta x}, \\
    a_2 &= \frac{9}{60\Delta x}, \\
    a_3 &= -\frac{45}{60\Delta x}, \\
    a_4 &= 0, \\
    a_5 &= -\frac{45}{60\Delta x}, \\
    a_6 &= \frac{9}{60\Delta x}, \\
    a_7 &= -\frac{1}{60\Delta x}
\end{align*}
\]

Using these relations results in:

\[ u^{n+1}_i = u^n_i - c \left(\frac{-u_{i-3} + 9u_{i-2} - 45u_{i-1} + 0u_i + 45u_{i+1} - 9u_{i+2} + u_{i+3}}{60\Delta x}\right) \Delta t \]

If the following profile, which has a local maximum and minimum of 1, is used.

\[ u^n_{i-3} = 1 + \varepsilon, u^n_{i-2} = u^n_{i-1} = u^n_i = u^n_{i+1} = u^n_{i+2} = u^n_{i+3} = 1 \]

The result is:

\[ u^{n+1}_i = 1 - c \left(\frac{-(1 + \varepsilon) + 9 - 45 + 45 - 9 + 1}{60\Delta x}\right) \Delta t = 1 - c \left(\frac{\varepsilon}{60\Delta x}\right) \Delta t \]

If \( \varepsilon > 0 \), a new local minimum in the solution will occur as the scheme will make \( u^{n+1}_i < 1 \). If \( \varepsilon < 0 \), a new local maximum in the solution will occur as the scheme will make \( u^{n+1}_i > 1 \). In both cases this will occur for any \( \Delta t \neq 0 \).

2.3.2 Monotonic High Order Scheme

Define a monotonic scheme as one that does not allow the process of convection to cause the value of a quantity at a grid index to be greater or less than itself or its neighbors at the current time step in one dimension. Note that this restriction is intertwined with the time integration scheme used. In particular, a first-order explicit time integration scheme with a CFL number \( \leq 1 \) is assumed. Any scheme can be made monotonic by the following.
Minimize \( a^2 \) in \( u_i^{n+1} = -\Delta t \left( c \frac{\partial u}{\partial x} \vert_{\text{high}} + a \right) + u_i^n \), where \( \frac{\partial u}{\partial x} \vert_{\text{high}} \) is any scheme of order greater than one.

subject to the constraints that:

\[ u_i^{n+1} \leq \max \left( \text{neighbors} \right) \]

\[ u_i^{n+1} \geq \min \left( \text{neighbors} \right) \]

There are three possibilities to this optimization problem. The first is that the constraints are satisfied with the minimum value of \( a^2 = 0 \). The second is that \(-\Delta t \left( \frac{\partial u}{\partial x} \vert_{\text{high}} \right) + u_i^n \geq \max \left( \text{neighbors} \right) \). In this case the minimum value of \( a^2 \) that meets the constraints is \( \frac{\max(\text{neighbors}) - u_i^n}{-\Delta t} = a \rightarrow \left( \frac{\max(\text{neighbors}) - u_i^n}{-\Delta t} \right)^2 = a^2 \). This minimum value of \( a^2 \) will be that which gives a result of \( \max \left( \text{neighbors} \right) \) when the modified convection algorithm is used with the given time integration scheme. The third is that \(-\Delta t \left( \frac{\partial u}{\partial x} \vert_{\text{high}} \right) + u_i^n \leq \min \left( \text{neighbors} \right) \). In this case the minimum value of \( a^2 \) that meets the constraints is \( \frac{\min(\text{neighbors}) - u_i^n}{-\Delta t} = a \rightarrow \left( \frac{\min(\text{neighbors}) - u_i^n}{-\Delta t} \right)^2 = a^2 \). This minimum value of \( a^2 \) will be that which gives a result of \( \min \left( \text{neighbors} \right) \) when the modified convection algorithm is used with the given time integration scheme. The value of \( a \), determined by the following procedure, will ensure the convection scheme is monotonic in one dimension if the given time integration algorithm is used. It is not possible to define a monotonic condition in more than one dimension or with a greater than first-order time integration scheme.

The choice made for the neighbors will determine the constraints imposed. Consider the case of a positive value for \( c \) and the first-order upwind scheme. The result is:

\[ u_i^{n+1} = -\Delta t \left( c \left( \frac{u_i^n - u_{i-1}^n}{\Delta x} \right) \right) + u_i^n = \frac{\Delta t c}{\Delta x} u_{i-1}^n + u_i^n \left( 1 - \frac{\Delta t c}{\Delta x} \right) \]

This means that for a CFL number \( 0 \leq \frac{\Delta t c}{\Delta x} \leq 1 \), the system is always bounded by the upstream pair at the previous time step. The value of \( u_{i+1}^n \) has no effect on the result, which is as expected physically. This scheme will always be bounded by its upstream pair at the previous time step.
This indicates that the appropriate choice of neighbors is the upwind pair.

Regardless of which choice is made for the neighbors, this monotonic scheme can be thought of as a derivative limiter. Given a CFL number and a high order derivative, the high order derivative will be limited so that if used with the given time integration scheme it will not produce values that are outside the maximum and minimum of the points chosen as neighbors (in this case the upstream pair at the previous time step). A higher-order scheme that has been made monotonic by the above procedure does not violate Godunov’s theorem. Godunov’s theorem states that it is impossible to construct a monotonic linear scheme with an order of accuracy greater than one. The high order scheme has become a nonlinear scheme through the application of the minimization procedure. Another way of stating this is that the scheme is non-linear as it is a function of the convected profile. What this minimization procedure actually does is add viscosity (i.e. limits the derivatives) in regions of high gradients. In regions of low gradients no viscosity is added (i.e. the derivatives are not limited).

2.3.3 High Order Time Integration

So far, only first-order time integration schemes have been considered. In DNS, higher-order explicit time integration schemes are frequently used in order to reduce phase errors. As before, a scheme can be made monotonic by considering the following minimization problem:

Minimize \( \alpha^2 \) in \( u_i^{n+1} = [-\Delta t (\alpha_1 \frac{\partial u}{\partial x}|_{\text{high}} + \alpha_2 C_{n-1} + \alpha_3 C_{n-2} + \ldots + \alpha_m C_{n-m+1} + a) + u_i^n] \),

where \( \frac{\partial u}{\partial x}|_{\text{high}} \) is any scheme of order greater than one and \( \alpha \) represents the time integration weights of an \( m \) order explicit time integration scheme.

subject to the constraints that:

\[
\begin{align*}
  u_i^{n+1} &\leq \max (\text{neighbors}) \\
  u_i^{n+1} &\geq \min (\text{neighbors})
\end{align*}
\]

Here \( C_{n-1}, C_{n-2} + \ldots + C_{n-m} \) represent the convection term at previous time steps. It is referred to as \( C_{n-1}, C_{n-2} \ldots C_{n-m+1} \) and not \( \frac{\partial u^{n-1}}{\partial x}|_{\text{high}}, \frac{\partial u^{n-2}}{\partial x}|_{\text{high}}, \ldots \frac{\partial u^{n-m+1}}{\partial x}|_{\text{high}} \) because it is
assumed to have been made monotonic at the previous time levels. A definition of a monotonic scheme cannot be established because of the terms $C_{n-1}, C_{n-2}, \ldots, C_{n-m+1}$. One problem is that the convection terms $C_{n-1}, C_{n-2}, \ldots, C_{n-m}$ are not obtained from solving Eq. 2.4, but instead are obtained from solving the Navier-Stokes equations (Eq. 2.1 and Eq. 2.2) or the scalar equation (Eq. 2.3) at previous time steps. As a result, there is no relationship between $C_{n-1}, C_{n-2}, \ldots, C_{n-m}$ and Eq. 2.4 (on which monotonicity is defined). Another problem is that the very definition of monotonicity is based on the properties of a first-order spatial scheme and first-order explicit time integration scheme applied to Eq. 2.4. This can be seen by applying higher-order explicit time integration and the first-order upwind spatial scheme to Eq. 2.4.

$$u_{i}^{n+1} - u_{i}^{n} = -\Delta t \left( \alpha_{1} C \left( \frac{u_{i}^{n} - u_{i+1}^{n}}{\Delta x} \right) + \alpha_{2} C \left( \frac{u_{i}^{n} - u_{i-1}^{n}}{\Delta x} \right) \right) + \cdots + \alpha_{m} C \left( \frac{u_{i}^{n-m+1} - u_{i-1}^{n-m+1}}{\Delta x} \right)$$

$$u_{i}^{n+1} = \alpha_{1} \frac{\Delta t c}{\Delta x} u_{i-1}^{n} + \left( 1 - \alpha_{1} \frac{\Delta t c}{\Delta x} \right) u_{i}^{n} + \alpha_{2} \frac{\Delta t c}{\Delta x} \left( u_{i}^{n-1} - u_{i-1}^{n-1} \right) + \cdots + \alpha_{m} \frac{\Delta t c}{\Delta x} \left( u_{i}^{n-m+1} - u_{i-1}^{n-m+1} \right)$$

This results in terms such as $\alpha_{m} \frac{\Delta t c}{\Delta x} \left( u_{i}^{n-m+1} - u_{i-1}^{n-m+1} \right)$ which appear as source terms (in the first-order upwind spatial and first-order explicit time integration scheme, $u_{i}^{n+1}$ depends only on the upwind pair at the previous time step). These terms may cause $u_{i}^{n+1}$ to fail to be bounded by its upwind pair and therefore would allow the creation of new local extrema. However, this does not mean that a monotonic scheme obtained from using the minimization procedure with a first-order time integration scheme cannot be used with a high order time integration scheme. As the minimization procedure is nothing more than a type of derivative limiter for a spatial difference scheme, it can be applied to any difference scheme. Results show that while the use of the monotonic scheme with a high order time integration scheme allows the creation of new maxima and minima, these are substantially smaller than without the monotonic scheme.

### 2.3.4 Clipping

Monotonic schemes have a problem regarding maxima or minima as can be seen by considering
the convection of a profile or wave by Eq. 2.4. The initial conditions (at time $n$) are as shown in Fig. 2.1, with the maximum in the profile ($u_i^n$) at a grid point. The solution is advanced half a grid point to time level $n + 1$ by a perfect method. At this point in time the maximum is no longer representable on the discrete grid. The maximum extrema is now $u_{i+1}^{n+1}$. The solution is advanced another half a grid point to time level $n + 2$ using a high order monotonic scheme. The solution for $u_{i+1}^{n+2}$ will be less than or equal to $u_{i+1}^{n+1}$ as $u_{i+1}^{n+1}$ is the limiting extrema representable on the discrete grid. This is what is known as clipping; it is a result of the fact that the maximum or minimum in a profile or wave is not representable on a discrete grid at all times. Clipping causes extrema to be progressively reduced as the profile is convected in time. Note that a non-monotonic higher-order method might predict a value for $u_{i+1}^{n+2}$ which is greater than $u_{i+1}^{n+1}$. Note also that if one were to advance half a grid point to time level $n + 1$ by the high order monotonic scheme it would be quite possible to generate a value for $u_{i+1}^{n+1}$ which is greater than the exact value. This is because the limiting extrema is $u_i^n$.

### 2.3.5 Results of One-dimensional Profile

A test of a convection scheme can be made by convecting a one-dimensional profile. This is done by solving Eq. 2.4 on an evenly spaced one-dimensional grid using a constant time step. As
periodic boundary conditions are used, the profile can convect over an unlimited distance. The initial conditions are a continuous function represented on the discrete grid. The exact solution is known if the profile is convected an integer number of cells from one grid point to another on the even grid. It is possible to define the profile by a continuous function and convect it a non-integer number of time steps from one grid point to another on an even grid. In this case the definition of the exact solution is clear; however the definition of the error is not clear as the numerical scheme will be expected to generate intermediate values of which it has no knowledge due to the finite resolution of the grid. This issue is avoided by comparing the numerical solution with the exact solution only at those time steps at which the exact solution has convected an integer number of cells. Fig. 2.2 compares the results of the exact solution, the linear sixth-order accurate central difference scheme given by Eq. A.35 and the monotonic version of Eq. A.35 after the profile has been convected 100 grid points. The three plots represent the different time integration schemes given by Eq. A.37, Eq. A.39 and Eq. A.41. It can be seen that for the time integration scheme given by Eq. A.37 the monotonic scheme is necessary for stability. The linear scheme is stable for the other time integration schemes, but generates new maxima and minima. The result of convecting the profile a further 400 grid points is shown in Fig. 2.3. At this location the linear scheme combined with the second-order time integration scheme given by Eq. A.39 is generating large new maxima and minima. The effect of clipping can be seen quite clearly in Fig. 2.3. The problem of clipping in which the maxima or minima are reduced is an inherent feature of monotonic schemes. The first-order time integration scheme given by Eq. A.37 exhibits a large phase shift error. For this reason this method is not commonly used when time accuracy is desired. A measure of the error as a function of the time that the profile has been convected is given by Fig. 2.4. It can be seen that the monotonic scheme has a lower value of error for all time integration schemes.
Figure 2.2: Profile convected 100 grid points at a CFL number of .125

While the present work is concerned with unsteady simulations, it should be noted that there can be issues with convergence if a steady state solution is sought using a monotonic scheme. This has been observed by the author when using the monotonic scheme described in the present work to obtain a steady state solution to flow in a square driven cavity, flow over a cylinder and flow around a sphere. Whether or not this occurs has been observed in the square driven cavity to depend on the time integration scheme used. What occurs in these cases is not an instability but an inability to drive the residual to zero. This has been observed by others (Jorgenson & Turkel, 1992). The system of equations that is solved in the steady state limit is, of course, a non-linear system as a result of the convective terms. This non-linearity is, of course, present if a non-monotonic (linear) convection scheme is used; however if a monotonic convection scheme
is used an additional non-linearity is introduced into the discretized system. This additional non-linearity is what can result in the inability to drive the residual to zero. It should be noted that for an explicit convection scheme this does not affect the ability to drive the residual to zero at each physical time step. This is a result of the fact that if using an explicit convection scheme, the non-linear convective terms appear as known source terms (because they are evaluated at previous time steps) in the system of equations at each physical time step. This system is a linear system with a guaranteed solution.

2.3.6 Results of Film Cooling Problem

The problem definition and boundary conditions are the same as used for the film cooling jet with a prescribed jet exit profile which is presented later (see Page 165). Three grids with
Figure 2.4: RMS error of Eq. A.35 at CFL number of .125

dimensions of $142 \times 39 \times 34$, $286 \times 80 \times 70$, and $574 \times 161 \times 142$ are used. The grid stretching is
the same as for the film cooling jet with a prescribed jet exit profile which is presented later. The
grids are obtained by roughly doubling the number of points in each direction, while keeping the
grid stretching parameters the same. The non-dimensional time steps used on the grids are .016,
.008, and .004 respectively. This results in maximum CFL numbers of ~.2 in the $x$ direction,
~.1 in the $y$ direction and ~.08 in the $z$ direction. The Reynolds number based on $U_0$ and jet
diameter is 6000. This Reynolds number is chosen so that stable solutions could be obtained for
the upwind schemes on at least the two finest grids. For instance the use of a fifth-order accurate
upwind scheme (Eq. A.29) on the $286 \times 80 \times 70$ grid is unstable at a Reynolds number of 6500.
The use of Eq. A.35 is unstable on all three grids at a Reynolds number of 6000. Third and fifth
order accurate upwind schemes (Eq. A.25 and Eq. A.29) are unstable on the $142 \times 39 \times 34$ grid at a Reynolds number of 6000. The monotonic versions of Eq. A.25 and Eq. A.29 are stable on this grid at this Reynolds number and are discussed. In this problem the boundary condition for the scalar is zero at the jet exit and one at the crossflow inlet. As a result the maximum and minimum values of the scalar should be zero and one respectively.

### 2.3.6.1 Linear Upwind Schemes

The maximum scalar in any part of the flow field as a function of time is given in Fig. 2.5a for the $286 \times 80 \times 70$ grid. This figure compares the monotonic version of Eq. A.35 with the linear schemes given by Eq. A.25 and Eq. A.29. Fig. 2.5b is the same but on the $574 \times 161 \times 142$ grid. The six point upwind linear scheme (Eq. A.29) has overshoots on the order of 30% on the $286 \times 80 \times 70$ grid and 15% on the $574 \times 161 \times 142$ grid. The four point upwind scheme overshoots less, approximately 15% on the $286 \times 80 \times 70$ grid and 11% on the $574 \times 161 \times 142$ grid. Because physically the scalar cannot be larger than one, this means that the maximum error in the scalar field using the upwind schemes is at least 30%. Due to the pressure gradient term, which acts as a source term in the equations for the velocity, the velocity (and therefore the kinetic energy) does not have known bounds such as the scalar field does. This makes it more difficult to quantify the effects of a non-monotonic scheme on the velocity field. However, an interesting difference in the results of the monotonic and the linear schemes can be seen in Fig. 2.6a, which shows the maximum kinetic energy in any part of the flow field as a function of time on the $286 \times 80 \times 70$ grid. Fig. 2.6b is similar, but on the $574 \times 161 \times 142$ grid. It can be seen that the kinetic energy fluctuates more extremely in the case of the linear schemes. In particular, the six point upwind scheme exhibits greater fluctuations. The fluctuations are larger for all schemes on the $574 \times 161 \times 142$ grid than on the $286 \times 80 \times 70$ grid. This might appear unusual; however, it must be considered that the $574 \times 161 \times 142$ grid contains approximately eight times as many grid points as the $286 \times 80 \times 70$ grid which means that the search for the maximum kinetic energy
occurs over a more refined space on the $574 \times 161 \times 142$ grid than on the $286 \times 80 \times 70$ grid.

2.3.6.2 Monotonic upwind schemes

The monotonic versions of Eq. A.25 and Eq. A.29 can be seen in Fig. 2.7 to drastically reduce overshoots in the scalar field. The overshoots are lower for both upwind schemes than for the monotonic version of central difference scheme (Eq. A.35). This is because, aside from the diffusion added as part of the monotonic limiters, the upwind schemes have natural diffusion (which comes strictly from the linear part of the schemes) that is added everywhere in the flow field. This diffusion stabilizes the schemes somewhat, although they may become unstable if the Reynolds number or grid spacing is increased. This lack of natural diffusion in the linear
part of the scheme is why the linear central difference scheme (Eq. A.35) is unstable with the combination of grids and Reynolds number used in the present work. In Fig. 2.8 the fluctuations in the kinetic energy are also greatly reduced, although as for the linear schemes they are also larger on the $574 \times 161 \times 142$ grid.

![Figure 2.7: Maximum scalar as a function of time, (a) $286 \times 80 \times 70$ grid, (b) $574 \times 161 \times 142$ grid](image)

![Figure 2.8: Maximum kinetic energy as a function of time, (a) $286 \times 80 \times 70$ grid, (b) $574 \times 161 \times 142$ grid](image)

### 2.3.6.3 Differences Between Monotonic and Linear Upwind Schemes

The difference between the linear schemes and the monotonic version of the same schemes can be seen in Fig. 2.9 for both the four and six point upwind schemes. The limiting can be seen to drastically reduce the overshoots for both the four and six point upwind schemes. The reduction is less for the four point than the six point upwind scheme, which is a result of the greater natural
diffusion of the four point upwind scheme.

Figure 2.9: Maximum scalar as a function of time, (a) $286 \times 80 \times 70$ grid, (b) $574 \times 161 \times 142$ grid

2.3.6.4 Monotonic Centered Scheme on Different grids

The behavior of the maximum scalar as a function of time on the different grids can be seen to be relatively independent of the grid for the monotonic version of Eq. A.35 (Fig. 2.10). Note that a solution can be obtained on the coarsest grid ($142 \times 39 \times 34$) using the monotonic version of Eq. A.35, while both of the linear upwind schemes are unstable on this grid, at this Reynolds and Prandtl number.

Figure 2.10: Monotonic version of Eq. A.35 on different grids
Chapter 3 Pressure Velocity Coupling

3.1 Introduction

The pressure velocity coupling in the solution of the unsteady incompressible Navier Stokes equations has long been a computationally expensive part of the solution process. The basic problem is that of determining an equation to solve for the pressure. Within the framework of finite differences there have historically been four approaches. Each approach (briefly described below) has specific advantages and disadvantages. Their use has been dictated by the application at hand, and the preference of the user. In this paper we focus attention on the artificial compressibility and Symmetric Coupled Gauss Seidel (SCGS) approaches. Two new schemes that contain elements of the artificial compressibility and SCGS methods are proposed for determining the pressure and velocity fields. Comparisons of the new methods are made with other methods with respect to robustness and efficiency for parallel computations. The goal is to develop a method that is suitable for large scale computations, such as Direct Numerical Simulations (DNS).

The first and most common approach for obtaining the pressure field have been methods that involve the solution of a Poisson equation for pressure. Such methods include the fractional step method (Chorin, 1968) and the SIMPLE type methods (Patankar, 1980). In these methods, an elliptic pressure-Poisson equation is derived from the momentum and continuity equations. The solution of the resulting Poisson equation is the greatest computational expense of this class of methods.

The second is the method of artificial compressibility (Chorin, 1967) which involves the addition of a pseudo time derivative to the equation set. It can be shown to be the result of low Mach number preconditioning of the compressible Navier-Stokes equations as the Mach number
goes to zero. It is known that the explicit solution of the compressible Navier-Stokes equations at low Mach numbers is very inefficient (Pletcher & Chen, 1993) (Merkle & Choi, 1988) due to the increasing stiffness of the equations as the Mach number is decreased. This is a result of the increasing ratio of sound speed to that of the velocity. To retain stability, increasingly small time steps are required in order to capture the acoustic waves, the speed of which increases relative to the convection speed. One solution to this would be to use an implicit method that eliminates the stability restriction on the time step. The solution of the equations arising from an implicit method would be obtained by an iterative method, direct methods being too expensive. If the time step needed to accurately capture the acoustic waves is of the order of that needed for stability for an explicit scheme, and one was interested in accurately capturing the acoustic waves, then there would be no reason to use an implicit method, which is more computationally expensive per time step. Therefore, in order to be computationally efficient, a time step would be chosen such that the convective terms (which have a substantially larger time scale than the acoustic waves at low Mach numbers) alone are accurately represented. This time step would be too large to accurately resolve the acoustic waves. As a result, the computational method of solving the equations cannot capture the physics concerning the acoustic waves. Each iteration of the iterative method could then be thought of as equivalent to an iteration in pseudo time. Low Mach number preconditioning introduces a pseudo time derivative and reduces the effective speed of sound so that the acoustic waves travel at a velocity close to the convective velocity. The preconditioning clearly must be chosen so that it does not affect the equations when the pseudo time derivative goes to zero. As in an implicit method, a time step will be chosen based on the convective terms. This time step will be much larger than that which would be chosen if the acoustic waves were of interest. Of crucial importance is that the acoustic waves are not in general resolved in an implicit method or in low Mach number preconditioning because of the large physical time step. The
time scale at which the acoustic waves operate is simply not resolved. Another way of looking at this is that in a truly incompressible flow there is no direct influence of the pressure history at each time step. The method of artificial compressibility introduces a finite speed of sound, of a magnitude equivalent to the convective velocity, into the incompressible Navier-Stokes equations which have an infinite speed of sound. The artificial compressibility method requires the selection of a parameter, which is related to the artificially introduced speed of sound, that can vary by three orders of magnitude depending on the flow. Thus, the optimal value of this parameter has to be chosen by trial and error, and, given the large range of values that this parameter can take, that requires that several trial solutions be attempted. Therefore, choosing this parameter for large scale computations can be a very time consuming task.

The third approach is what is known as direct methods. These methods involve a global coupled solution of the entire discretized system in matrix form at one step. If an implicit time integration scheme is used for the convective terms then iterations must be performed at each time step to solve the resulting nonlinear equations. Direct methods are infeasible for large problems due to the computational expense of direct solutions of matrices.

The fourth approach is SCGS (Vanka, 1986). Most solutions to the unsteady incompressible Navier Stokes equations are extremely computationally expensive in terms of spatial and time resolution requirements. For this reason solution algorithms that work on vector, and more importantly, distributed memory parallel machines are required. This requirement eliminates the original SCGS algorithm due to its inherently serial nature. However SCGS can be parallelized by the use of grid coloring schemes, although coloring strategies in complex multi-block grids can become quite difficult or impossible.

It is possible to use a conjugate gradient method, such as GMRES, on the entire discretized system or on the systems that arise from the artificial compressibility method. A conjugate
gradient method is not a good choice if a DNS solution is sought for a flow. The reason lies in the fact that the error in the iterative solution of the discretized system has a specific form which is a result of the small physical time steps used in DNS. Due to the fact that small time steps (i.e. CFL numbers ~1) are needed for accuracy, the solution at the time level being sought is close to the solution at the previous time level. Because the solution at the previous time level can be used as a guess for the solution at the time level being sought, the error in wave number space is almost entirely in the high wave number region. Conjugate gradient methods reduce the error in all wavelengths equally, which is good for flows in which all wavelengths of error are significant, but inefficient if that is not the case as in DNS. Local iterative methods (such as SCGS and the artificial compressibility method as implemented in the present work) in general have the property of reducing the errors in the high wave number region effectively while leaving the error in the low wave number region relatively unchanged. This makes these methods highly suitable for DNS flows. However, for flows in which all wavelengths of error are significant, these methods must be used in a multigrid scheme, in order to achieve fast reduction of all wavelengths of error.

This paper describes two new methods suitable for parallel computations that combine elements of SCGS and the artificial compressibility method. This work is motivated by the need for a more robust algorithm for the pressure-velocity coupling that is suitable for parallel computations. Comparisons are made between the new methods, a SCGS scheme parallelized by grid coloring and the method of artificial compressibility. These methods are particularly evaluated from the perspective of their suitability for the Direct Numerical Simulation (DNS) of various flows.

3.2 Nomenclature

\[ a = \text{Coefficient of } u \text{ in the } x \text{ momentum equation (possibly a function of } u, v \text{ and } w \text{ from convective terms)} \]
\[ b = \text{Coefficient of } v \text{ in the } y \text{ momentum equation (possibly a function of } u, v \text{ and } w \text{ from convective terms)} \]
\[ c = \text{Coefficient of } w \text{ in the } z \text{ momentum equation (possibly a function of } u, v \text{ and } w \text{ from convective terms)} \]
\[ A = \text{Coefficient of } p \text{ in the } x \text{ momentum equation (only a function of the grid)} \]
$B =$ Coefficient of $p$ in the $y$ momentum equation (only a function of the grid)  
$C =$ Coefficient of $p$ in the $z$ momentum equation (only a function of the grid)  
$d =$ Coefficient of $u$ in the continuity equation (only a function of the grid)  
$e =$ Coefficient of $v$ in the continuity equation (only a function of the grid)  
$f =$ Coefficient of $w$ in the continuity equation (only a function of the grid)  
$g =$ Coefficient of the time derivative term at the $n+1$ level in the momentum equations.  
$q =$ Coefficient of neighboring $u$ in the $x$ momentum equation resulting from implicit convection or diffusion.  
$r =$ Coefficient of neighboring $v$ in the $y$ momentum equation resulting from implicit convection or diffusion.  
$s =$ Coefficient of neighboring $w$ in the $z$ momentum equation resulting from implicit convection or diffusion.  
$U =$ Residual of $x$ momentum equation  
$V =$ Residual of $y$ momentum equation  
$W =$ Residual of $z$ momentum equation  
$D =$ Residual of continuity equation

3.3 Governing Equations and Discretization

3.3.1 Governing Equations

The governing equations of interest in the present work are the non-conservative unsteady three-dimensional Navier-Stokes equations (Eq. 2.1 and Eq. 2.2).

3.3.2 Spatial Discretization

Equations 2.1 and 2.2 are discretized using the finite difference method on a staggered grid, as shown in Fig. 3.1. The grid has each pressure location surrounded by six velocity components. Pressure is not defined on non-periodic boundaries. Four point upwind biased or seven point central difference schemes are used for the derivatives in the convective terms. On an even grid the accuracy of the schemes are $O(\Delta x^3)$ and $O(\Delta x^7)$ respectively. A monotonic limiter is applied to the central difference convection scheme. Due to the staggered grid, none of the velocity components are stored at the same location. They are however, needed at the same location in order to compute the parts of $u_j \frac{\partial u_i}{\partial x_j}$ where $i \neq j$. They are obtained by fitting a two-dimensional Lagrange surface over the point in question. The fitting uses products of four or six point Lagrange polynomials. This results in respectively sixteen or thirty-six points in the fitting of the surface. On an even grid the accuracy of the interpolation schemes are $O(\Delta x^4)$.
and $O(\Delta x^6)$ respectively. The diffusive terms are represented by five or seven point centered schemes, with an accuracy on an even grid of $O(\Delta x^4)$ and $O(\Delta x^6)$ respectively. The pressure gradient term and the terms in the continuity equation are represented by a two-point centered scheme, with an accuracy on an even grid of $O(\Delta x^2)$. The number of points in all schemes is retained as a non-periodic boundary is approached. This is done by keeping the number of points in the stencil constant while shifting towards the boundary the point at which the derivative is evaluated. For a centered scheme, this results in moving from a centered stencil to one that is biased away from the non-periodic boundary.

3.3.3 Temporal Discretization

In the present work, explicit and implicit schemes are used for the convection and diffusion terms and comparisons are made between the results of both such schemes. The current time level (i.e. the time level for which the solution is sought) is denoted by the $n + 1$ superscript. The pressure gradient is always treated implicitly and appears only at the current time level. The explicit scheme used is a third-order accurate in time multi level scheme given by Eq. 3.1.

$$u_i^{n+1} = \Delta t \left( - \frac{\partial p}{\partial x_i} \right)^{n+1} + \frac{23}{12} \Delta t \left( \frac{1}{Re} \frac{\partial^2 u_i}{\partial x_j \partial x_j} - u_j \frac{\partial u_i}{\partial x_j} \right)^n - \frac{1}{16} \frac{\Delta t}{12} \left( \frac{1}{Re} \frac{\partial^2 u_i}{\partial x_j \partial x_j} - u_j \frac{\partial u_i}{\partial x_j} \right)^{n-1} + 5 \frac{\Delta t}{12} \left( \frac{1}{Re} \frac{\partial^2 u_i}{\partial x_j \partial x_j} - u_j \frac{\partial u_i}{\partial x_j} \right)^{n-2} + u_i^n$$

(3.1)

The implicit scheme used is a third-order accurate in time scheme given by Eq. A.46.

3.4 Artificial Compressibility

This method was first proposed by (Chorin, 1967), who used it to solve the steady incompressible Navier-Stokes equations. It has since been used by other researchers ((McHugh & Ramshaw, 1995), (Soh & Goodrich, 1988), (Muldoon & Acharya, 1998) and (Beddhu et al., 1994)) to solve complicated unsteady flows. It consists of the addition of an artificial time derivative to the momentum and continuity equations. These equations are then advanced in the artificial time dimension (pseudo time) until the pseudo time derivative goes to zero. For an
unsteady problem this must be done every time step. Because the continuity equation does not contain pressure, it is introduced by defining an artificial equation of state, \( p = \rho \beta \), which results in the definition of an artificial speed of sound \( \sqrt{\beta} \). The resulting system of equations is given by Eq. 3.2 and Eq. 3.3.

\[
\begin{align*}
\frac{\partial u_i}{\partial \tau} &= -\frac{\partial p}{\partial x_i} + \frac{1}{Re} \frac{\partial^2 u_i}{\partial x_j \partial x_j} - \frac{\partial u_i}{\partial t} - u_j \frac{\partial u_i}{\partial x_j} \\
\frac{\partial p}{\partial \tau} &= -\beta \frac{\partial u_i}{\partial x_i} \quad \beta > 0
\end{align*}
\]

At first sight, it might appear that \( \beta \) is relatively easy to choose. For instance, \( \sqrt{\beta} \) could be chosen to be equal to some representative convective velocity. Computational experience has shown (Rogers & Kwak, 1991) that this is not the case, and that \( \beta \) can vary from 1 to 1000 depending on the flow and the pseudo and physical time steps. In general, as \( \beta \) is increased, a smaller pseudo time step must be taken and the ratio of the momentum residuals to the residual of the continuity equation increases. It has been shown that for unsteady flows \( \beta \) is strongly a function of the physical time step (McHugh & Ramshaw, 1995). It would seem possible to have \( \beta \) chosen locally (as a result of local flow conditions), but to the author’s knowledge this has not been explored. Since time accuracy in pseudo time (\( \tau \)) is of no concern, the time derivatives in Eq. 3.2 and Eq. 3.3 are discretized using first-order methods. Both implicit (Rogers & Kwak, 1991) and explicit (McHugh & Ramshaw, 1995) methods have been used. Local time stepping, in which the equation is advanced at different rates in pseudo time depending on spatial position, can be used. This improves the convergence rate by not restricting the global pseudo time step to the minimum required locally.

3.4.1 Velocity

The momentum equations (Eq. 2.1) are represented in discretized form by Eq. 3.4. In the following equations, the subscript \( i \) denotes the velocity component of interest while the arrows
or vector notation are used to indicate the inclusion of all three velocity components.

\[ 0 = F_i \left( \overrightarrow{u}^{n+1}, \overrightarrow{u}^{n}, \overrightarrow{u}^{n-1}, \ldots \right) - \frac{\partial p}{\partial x_i}^{n+1} - g u_i^{n+1} \]  

(3.4)

In Eq. 3.4, \( F_i \) contains the convection and diffusion terms, source terms and all parts of the time derivative at previous levels. It is not necessarily a function of the \( n + 1 \) time level. For instance, one might choose an explicit representation for one or both of the convection or diffusion terms. Note that \( F_i \) does not contain the part of the time derivative term at the \( n + 1 \) level. That term is given by \( g u_i^{n+1} \). A pseudo time derivative term is now added to the momentum equations. This term has nothing to do with real or physical time. The goal is to advance this new equation in pseudo time until it reaches steady state at which point a solution will be obtained to the representation of the momentum equations.

\[ \frac{\partial u_i}{\partial \tau} = F_i \left( \overrightarrow{u}^{n+1}, \overrightarrow{u}^{n}, \overrightarrow{u}^{n-1}, \ldots \right) - \frac{\partial p}{\partial x_i}^{n+1} - g u_i^{n+1} \]

This pseudo time derivative term is represented by a first-order accurate expression.

\[ \frac{u_i^{n+1,m+1} - u_i^{n+1,m}}{\Delta \tau_i} = F_i \left( \overrightarrow{u}^{n+1,m}, \overrightarrow{u}^{n}, \overrightarrow{u}^{n-1}, \ldots \right) - \frac{\partial p}{\partial x_i}^{n+1,m} - g u_i^{n+1,m} \]

\[ u_i^{n+1,m+1} = \left[ F_i \left( \overrightarrow{u}^{n+1,m}, \overrightarrow{u}^{n}, \overrightarrow{u}^{n-1}, \ldots \right) - \frac{\partial p}{\partial x_i}^{n+1,m} - g u_i^{n+1,m} \right] \Delta \tau_i + u_i^{n+1,m} \]

3.4.1.1 Local Time Stepping

Define \( G_i \left( \overrightarrow{u}^{n+1,m}, \overrightarrow{u}^{n}, \overrightarrow{u}^{n-1}, \ldots \right) - h_i \left( \overrightarrow{u}^{n+1,m} \right) u_i^{n+1,m} = F_i \left( \overrightarrow{u}^{n+1,m}, \overrightarrow{u}^{n}, \overrightarrow{u}^{n-1}, \ldots \right) - g u_i^{n+1,m} \) and the local time step (\( \Delta \tau_i \)) as in Eq. 3.5 (where \( \alpha_i \) is an under-relaxation factor).

\[ \Delta \tau_i = \frac{\alpha_i}{h_i \left( \overrightarrow{u}^{n+1,m} \right)} \]

(3.5)

Substituting the above definitions in the discretized momentum equations results in:

\[ u_i^{n+1,m+1} = \left[ G_i \left( \overrightarrow{u}^{n+1,m}, \overrightarrow{u}^{n}, \overrightarrow{u}^{n-1}, \ldots \right) - \frac{\partial p}{\partial x_i}^{n+1,m} \right] \frac{\alpha_i}{h_i \left( \overrightarrow{u}^{n+1,m} \right)} + u_i^{n+1,m} \]

Factoring out \( u_i^{n+1,m} \) leads to the following equation for \( u_i^{n+1,m+1} \):

\[ u_i^{n+1,m+1} = \left[ G_i \left( \overrightarrow{u}^{n+1,m}, \overrightarrow{u}^{n}, \overrightarrow{u}^{n-1}, \ldots \right) - \frac{\partial p}{\partial x_i}^{n+1,m} \right] \frac{\alpha_i}{h_i \left( \overrightarrow{u}^{n+1,m} \right)} + \left[ 1 - \alpha_i \right] u_i^{n+1,m} \]

(3.6)
It can be seen that this choice of time step is equivalent to Jacobi iteration with an under-relaxation factor of $\alpha_i$. It is local time stepping because the pseudo time step is a function of the velocity and the spatial location. Other choices (such as a constant time step for the entire spatial domain) can be made for the time step. The use of a constant time step results in different under-relaxation factors across the spatial domain which is inefficient as certain regions will have low under-relaxation factors. Note that if convection and diffusion are integrated explicitly in physical time, then $h_i \left( \vec{u}^{n+1,m} \right) = g$ and the local time step is not "local" as it is a constant over the entire spatial domain. If the time differencing is implicit then sometimes there is an advantage to freezing the evaluation of the implicit part of the computationally expensive convection and diffusion terms. Sub-cycling (not done in the present work) is then used to solve for pressure and velocity without recomputing the implicit part of the convection and diffusion terms. In the present work, the under-relaxation parameters for the momentum equations as used in all of the methods are referred to as $\alpha_u, \alpha_v, \alpha_w$. The under-relaxation parameter used for the pressure equation for all methods except for artificial compressibility is referred to as $\alpha_p$. It will be shown that the choice of the relaxation parameters is relatively straightforward, and that the same values can be used to provide robust convergence for a range of application problems.

3.4.2 Pressure

If an explicit first-order method is used to discretize Eq. 3.3 in pseudo time, the resulting equation for pressure is given by:

$$ p^{n+1,m+1} = \left( -\beta \frac{\partial \vec{u}_i^{n+1}}{\partial x_i} \right) \Delta \tau_p + p^{n+1,m} \quad (3.7) $$

The parameter $\Delta \tau_p$ in Eq. 3.7 is chosen as the average of the six pseudo time steps in Eq. 3.5 corresponding to the six velocity points surrounding each pressure location.

3.4.3 Algorithm

The use of Eq. 3.7 along with Eq. 3.6 defines the artificial compressibility method as used in the present work. Pseudocode which describes how the artificial compressibility algorithm can
be implemented is given below.

compute momentum residuals (no need for continuity equation residuals at this point)
solve for $u_i^{n+1,m+1}$ using Eq. 3.6
communicate $u_i^{n+1,m+1}$ between different processes
compute continuity equation residual using $u_i^{n+1,m+1}$
solve for $p^{n+1,m+1}$ using Eq. 3.7
communicate $p^{n+1,m+1}$ between different processes

The only synchronization points are communication routines. Other than these points all computations can proceed independently of each other.

3.5 SCGS

In the SCGS method the change in velocity and pressure is solved for at each iteration. To derive these equations, the discretized version of Eq. 2.1 and Eq. 2.2 can be written in the following general nonlinear matrix form:

$$H(u)u = s$$

Here $u$ is the solution, $H$ represents the discretized differential operator for the Navier-Stokes equations which in general is a function of $u$, and $s$ is a source term. The residual $R$ that results from an initial guess $u^*$ can be written as:

$$R = H(u^*)u^* - s = H(u^*)[u - \Delta u] - s = H(u^*)u - s - H(u^*)\Delta u \quad \text{where} \quad \Delta u + u^* = u$$

By definition $H(u)u - s = 0$. Since the intention is to use the procedure iteratively, the assumption that $H(u^*) = H(u)$ can be made. This results in the residual form given below (Eq. 3.8).

$$H(u^*)\Delta u = -R \quad (3.8)$$

It is permissible to change $H(u^*)$ when used in the above equation. For instance, if $H(u^*)$ is replaced by $\tilde{H}(u^*)$, a zero residual would result in $\Delta u = 0$ if $\tilde{H}(u^*)$ is a linearly independent
matrix. Of course there is no guarantee that the sequence of iterates resulting from using $\tilde{H}(u^*)$ instead of $H(u^*)$ will be a convergent sequence. This ability to modify the iteration matrix will be used when certain off-diagonal terms are dropped from the matrices that describe the pressure velocity coupling of some of the algorithms in the present work. Note that if an explicit time integration scheme is used for the convective terms in Eq. 2.1 then $H$ does not depend on $u$. In this case, no approximation is involved in writing in residual form if all the (linear) terms in $H$ are maintained.

First proposed by (Vanka, 1986), SCGS involves the coupling of the six momentum equations that surround a pressure location plus the continuity equation at that location. Referring to $p_{i,j}$ in Fig. 3.1 (which shows a two-dimensional schematic for simplicity), if an explicit formulation is used for the convective and diffusive terms, the equations for the six velocities surrounding $p_{i,j}$, along with the continuity equation at that same location, can be written in residual form as Eq. 3.9. Note that because the convective and diffusive terms are integrated in time using an explicit scheme, no terms have been dropped or approximated in Eq. 3.9. In the remainder of the present work the subscript indices $i, j, k$ refer to the grid points on a three-dimensional finite difference grid.

$$
\begin{bmatrix}
\Delta u_{i-1,j,k} & \Delta u_{i,j,k} & \Delta v_{i,j-1,k} & \Delta v_{i,j,k} & \Delta w_{i,j,k-1} & \Delta w_{i,j,k} \\
A_{i-1,j,k} & A_{i,j,k} & B_{i,j-1,k} & B_{i,j,k} & C_{i,j,k-1} & C_{i,j,k} \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\begin{bmatrix}
\Delta u_{i-1,j,k} \\
\Delta u_{i,j,k} \\
\Delta v_{i,j-1,k} \\
\Delta v_{i,j,k} \\
\Delta w_{i,j,k-1} \\
\Delta w_{i,j,k} \\
\end{bmatrix}
\begin{bmatrix}
U_{i-1,j,k} \\
U_{i,j,k} \\
V_{i,j-1,k} \\
V_{i,j,k} \\
W_{i,j,k-1} \\
W_{i,j,k} \\
D_{i,j,k} \\
\end{bmatrix}
$$

(3.9)

The above matrix equation can be solved, at each pressure location, for the changes in the six velocities and the change in pressure. However, as one moves from one pressure location to the next pressure location solving the matrix equation, one will have two solutions for each interior velocity. If one solves the matrix equation at main grid point $(i, j, k)$ one will get a value for
\[ \Delta u_{i-1,j,k}, \Delta u_{i,j,k}, \Delta v_{i,j-1,k}, \Delta v_{i,j,k}, \Delta w_{i,j,k-1}, \Delta w_{i,j,k} \] and \( \Delta p_{i,j,k} \). If one then solves the equation at main grid point \((i+1, j, k)\) one will get a value for \( \Delta u_{i,j,k}, \Delta u_{i+1,j,k}, \Delta v_{i+1,j-1,k}, \Delta v_{i+1,j,k}, \Delta w_{i+1,j,k-1}, \Delta w_{i+1,j,k} \) and \( \Delta p_{i+1,j,k} \). The result is that one will then have two different values for \( \Delta u_{i,j,k} \). All attempts to combine these two values have been highly unstable. The original SCGS algorithm did not have this problem because, when solving the equation at main grid point \((i+1, j, k)\) in Fig. 3.1, it used the previously computed value of \( u_{i,j,k} \) to recompute the residuals.

As a result, as one swept across the grid, one solved for each velocity twice but always used or kept the last one. This means that the expensive-to-compute momentum equation residuals are computed twice for each sweep of the grid. Depending on the direction \( i, j \) or \( k \) that one sweeps across the grid while solving for the velocities and pressure, one gets a different answer. Unfortunately SCGS is sensitive to the direction in which sweeps are made. Making a sweep in the direction opposite to the main velocity component can be unstable. This presents a problem in recirculating flows. The method does have the advantage of easy-to-choose under-relaxation parameters (e.g., \( \sim 0.9 \) if the convection and diffusion terms are explicit) for the changes in velocity and pressure. A more serious defect in using it for the direct numerical simulation of flows is that it is an inherently serial algorithm. Note that one could use SCGS separately in each domain as assigned to each processor. One could then use some method of combining the two different solutions obtained for the velocity field at domain boundaries. This has not been explored in the present work. One reason is that an instability may occur at domain boundaries, similar to the one experienced when trying to combine the two different solutions obtained for the velocity field at all interior points. Indeed, as the number of processors increases, the number of points at which some special treatment would be needed to combine the two solutions also increases. This relates to the other problem with this approach in that the solution would depend on the number of processors used to obtain it. This is a highly undesirable feature of any numerical algorithm.
as it makes it essentially impossible to verify that the algorithm as implemented in software and hardware is correct. Using the methods in the present work, it has been verified that it is possible to obtain a solution on one processor that is binary identical to one obtained from using a multiple number of processors in parallel.

If we consider using Eq. 3.9 to solve only for pressure we can see that we do not have these problems. The pressure at each main grid point can be uniquely determined in a completely parallel fashion independent of neighboring pressures. Therefore one can use SCGS to solve only for pressure and some other method (pseudo time stepping or relaxation) to solve for the velocity.

This is the main idea behind the following new algorithm.

3.6 New Algorithm, SCGS-PP

The idea for Symmetric Coupled Gauss Seidel Parallel Pressure (SCGS-PP) comes from combining elements of SCGS and artificial compressibility. The artificial compressibility algorithm is easily parallelized but, as noted earlier, contains two hard-to-choose parameters ($\Delta \tau$ and particularly $\beta$). The SCGS algorithm is inherently serial but has easy-to-choose relaxation parameters. The SCGS-PP algorithm is derived by using an equation for pressure obtained from the matrix equation Eq. 3.9 along with a local time stepping method to solve for the velocity.

Both the solution of the pressure and the velocity can proceed in a parallel fashion respectively.

3.6.1 Pressure

3.6.1.1 Explicit Convection Diffusion Time Integration

Solving the matrix equation Eq. 3.9 for $\Delta p_{i,j,k}$ results in:

$$
\Delta p_{i,j,k} = \frac{\left[ D_{i,j,k} - \beta_{i-1,j,k} U_{i-1,j,k} - \beta_{i,j,k} U_{i,j,k} - \gamma_{i,j-1,k} V_{i,j,k-1} - \gamma_{i,j,k-1} W_{i,j,k-1} - \lambda_{i,j,k} W_{i,j,k} \right]}{\chi_{i,j,k}}
$$

where

$$
\beta_{i-1,j,k} = \frac{d_{i-1,j,k}}{a_{i-1,j,k}}, \quad \beta_{i,j,k} = \frac{d_{i,j,k}}{a_{i,j,k}}, \quad \gamma_{i,j-1,k} = \frac{e_{i,j-1,k}}{b_{i,j-1,k}}, \quad \gamma_{i,j,k-1} = \frac{e_{i,j,k-1}}{b_{i,j,k-1}}, \quad \lambda_{i,j,k} = \frac{f_{i,j,k}}{c_{i,j,k}},
$$

$$
\chi_{i,j,k} = \frac{f_{i,j,k}}{c_{i,j,k}} - \frac{e_{i,j,k}}{b_{i,j,k}} - \frac{d_{i,j,k}}{a_{i,j,k}}.
$$
\[ \chi_{i,j,k} = -A_{i-1,j,k}d_{i-1,j,k} - A_{i,j,k}d_{i,j,k} - B_{i,j-1,k}e_{i,j-1,k} - B_{i,j,k}e_{i,j,k} - C_{i,j,k-1}f_{i,j,k-1} - C_{i,j,k}f_{i,j,k} \]

It is necessary to apply under-relaxation to \( \Delta p_{i,j,k} \) obtained from Eq. 3.10 in order to achieve convergence. This is discussed in the results section.

### 3.6.1.2 Implicit Convection Diffusion Time Integration

Due to the derivatives involved in the convection and diffusion terms, additional elements on the off diagonals arise in the matrix (Eq. 3.11) which describe the coupling between the six velocity points and pressure. Note that if an implicit convection scheme is not used, then the additional elements depend only on the grid and time step. Otherwise, the additional elements will also depend on the solution, as a result of the nonlinearity of the convective term. The values of these additional elements will depend on the specific differencing schemes used for the convection and diffusion terms.

\[
\begin{bmatrix}
    a_{i-1,j,k} & q_{i-1,j,k} & 0 & 0 & 0 & 0 & A_{i-1,j,k} \\
    q_{i,j,k} & a_{i,j,k} & 0 & 0 & 0 & 0 & A_{i,j,k} \\
    0 & 0 & b_{i,j-1,k} & r_{i,j-1,k} & 0 & 0 & B_{i,j-1,k} \\
    0 & 0 & r_{i,j,k} & b_{i,j,k} & 0 & 0 & B_{i,j,k} \\
    0 & 0 & 0 & 0 & c_{i,j,k-1} & s_{i,j,k-1} & C_{i,j,k-1} \\
    0 & 0 & 0 & 0 & s_{i,j,k} & c_{i,j,k} & C_{i,j,k} \\
    d_{i-1,j,k} & d_{i,j,k} & e_{i,j-1,k} & e_{i,j,k} & f_{i,j,k-1} & f_{i,j,k} & 0 \\
\end{bmatrix} \begin{bmatrix}
    \Delta u_{i-1,j,k} \\
    \Delta u_{i,j,k} \\
    \Delta v_{i,j-1,k} \\
    \Delta v_{i,j,k} \\
    \Delta w_{i,j,k-1} \\
    \Delta w_{i,j,k} \\
    \Delta p_{i,j,k} \\
\end{bmatrix} = \begin{bmatrix}
    U_{i-1,j,k} \\
    U_{i,j,k} \\
    V_{i,j-1,k} \\
    V_{i,j,k} \\
    W_{i,j,k-1} \\
    W_{i,j,k} \\
    D_{i,j,k} \\
\end{bmatrix}
\]

(3.11)

These additional terms make it cumbersome to obtain an analytical expression for \( \Delta p_{i,j,k} \) as was done for the case of an explicit convection and diffusion scheme. However, it can be seen that the expression for \( \Delta p_{i,j,k} \) can be obtained by using the appropriate row of the inverted matrix in Eq. 3.11 to form a linear combination of the residuals. Once this matrix inversion has been done, the amount of work needed to determine \( \Delta p_{i,j,k} \) is the same as for the case of explicit convection and diffusion. Note that as the general procedure involves solving for the changes in the velocity and pressure, it may be possible to change the matrix in Eq. 3.11 and still obtain a
solution. In particular the inclusion of the additional elements on the off diagonals arising from the convection and diffusion schemes is seen to make little difference in the rate of convergence and hence these elements are not used in the present work. This ability to modify the matrix can be used for computational efficiency if an implicit convection scheme is used. In this case (and even if the off diagonal terms are dropped) the matrix will depend on the solution. However, it has been observed that it is not necessary to perform the matrix inversion at every iteration. It is more efficient computationally to perform the matrix inversion only at certain time steps and obtain a representative inverted matrix. It is necessary to apply under-relaxation to $\Delta p_{i,j,k}$ in order to achieve convergence. This is discussed in the results section.

### 3.6.2 Velocity

The equation for the velocity is obtained by using Eq. 3.6.

### 3.6.3 Algorithm

The use of Eq. 3.10 along with Eq. 3.6 defines the SCGS-PP method. Pseudocode which describes how the SCGS-PP algorithm can be implemented is given below.

1. **compute momentum residuals** (no need for continuity equation residuals at this point)
2. **solve for** $u_i^{n+1,m+1}$ using Eq. 3.6
3. **communicate** $u_i^{n+1,m+1}$ between different processes
4. **compute momentum and continuity equation residuals** using $u_i^{n+1,m+1}$
5. **solve for** $p_i^{n+1,m+1}$ using Eq. 3.10 if explicit or the appropriate analogue from Eq. 3.11 if implicit
6. **communicate** $p_i^{n+1,m+1}$ between different processes

The only synchronization points are communication routines. Other than these points, all computations can proceed independently of each other.

### 3.7 New Algorithm, SCGS-PPV

This idea of defining symmetric subsets of the discretized equations that can be used to provide
a unique solution for a variable can also be applied to the velocity and used to define a new algorithm termed Symmetric Coupled Gauss Seidel Parallel Pressure Velocity (SCGS-PPV).

Consider the grid points surrounded by the red line in Fig. 3.1. For clarity, the third dimension and associated velocity component are not shown but can be deduced. For simplicity, the grid is assumed to be even with the same grid spacing ($\Delta x = 1/\alpha$) in all directions. Writing the discretized Navier-Stokes equations for the points in Fig. 3.1 results in Eq. 3.12.

\[
\begin{aligned}
\Delta u_{i-1,j,k} &= g \\ 
\Delta u_{i,j,k} &= 0 \\ 
\Delta u_{i+1,j,k} &= 0 \\ 
\Delta v_{i,j-1,k} &= 0 \\ 
\Delta v_{i,j,k} &= 0 \\ 
\Delta v_{i,j+1,k-1} &= 0 \\ 
\Delta v_{i+1,j,k} &= 0 \\ 
\Delta w_{i,j,k-1} &= 0 \\ 
\Delta w_{i,j,k} &= g \\ 
\Delta w_{i+1,j,k-1} &= 0 \\ 
\Delta w_{i+1,j+1,k} &= 0 \\ 
\Delta p_{i,j,k} &= -\alpha \\ 
\Delta p_{i+1,j,k} &= -\alpha \\
\end{aligned}
\]

\[
\begin{aligned}
\Delta u_{i,j,k} &= -\frac{1}{l g} \left( U_{i-1,j,k} + 5U_{i,j,k} + U_{i+1,j,k} - V_{i,j-1,k} + V_{i,j,k} + V_{i+1,j-1,k} + V_{i+1,j,k} - V_{i+1,j-1,k} - W_{i,j,k-1} + W_{i+1,j,k-1} - W_{i+1,j,k-1} - D_{i,j,k} \frac{g}{\alpha} - D_{i+1,j,k} \frac{g}{\alpha} \right) \\
\end{aligned}
\]

This yields the following solution for $\Delta u_{i,j,k}$.

\[
\Delta u_{i,j,k} = -\frac{1}{l g} \left( U_{i-1,j,k} + 5U_{i,j,k} + U_{i+1,j,k} - V_{i,j-1,k} + V_{i,j,k} + V_{i+1,j-1,k} + V_{i+1,j,k} - V_{i+1,j-1,k} - W_{i,j,k-1} + W_{i+1,j,k-1} - W_{i+1,j,k-1} - D_{i,j,k} \frac{g}{\alpha} - D_{i+1,j,k} \frac{g}{\alpha} \right) \\
\]
Similar equations (Eq. 3.14 and Eq. 3.15) can be obtained for $\Delta v_{i,j,k}$ and $\Delta w_{i,j,k}$ if the appropriate analogue to Eq. 3.12 is defined for the other two velocity components. Therefore (Eq. 3.14 and Eq. 3.15) are not obtained from Eq. 3.12.

$$
\Delta v_{i,j,k} = -\frac{1}{7g} \left( V_{i,j-1,k} + 5V_{i,j,k} + V_{i,j+1,k} - U_{i,j,k} + U_{i-1,j,k} + U_{i,j+1,k} - U_{i-1,j+1,k} \\
- W_{i,j,k} + W_{i,j-1,k} + W_{i,j+1,k} - W_{i,j+1,k-1} + D_{i,j,k} \frac{\alpha}{\tau} - D_{i,j+1,k} \frac{\alpha}{\tau} \right) 
$$

(3.14)

$$
\Delta w_{i,j,k} = -\frac{1}{7g} \left( W_{i,j,k-1} + 5W_{i,j,k} + W_{i,j,k+1} - U_{i,j,k} + U_{i-1,j,k} + U_{i,j,k+1} - U_{i-1,j,k+1} \\
- V_{i,j,k} + V_{i,j-1,k} + V_{i,j,k+1} - V_{i,j,k+1} + D_{i,j,k} \frac{\alpha}{\tau} - D_{i,j,k+1} \frac{\alpha}{\tau} \right) 
$$

(3.15)

These equations are used to solve for the velocity. Equation 3.12 has been written assuming that the grid spacing is the same in all directions. It also assumes two-point stencils for the pressure gradient and continuity equation and explicit treatment of the convection and diffusion terms. If this is not the case, the structure of Eq. 3.12 will be more complicated. These restrictions are necessary in order to easily obtain an analytic expression for $\Delta u_{i,j,k}$, $\Delta v_{i,j,k}$, $\Delta w_{i,j,k}$. The same general results would follow for an uneven grid and/or higher than two-point stencils for the pressure gradient and/or continuity equation. If an implicit convection scheme is used, the coefficients multiplying the residuals in Eq. 3.13 would be functions of the grid and velocity. Also, if an implicit convection or diffusion scheme is used, additional terms will appear in Eq. 3.12 as a result of the coupling between velocities introduced by the finite difference representations of the derivatives in the convection or diffusion schemes. In these cases, the expression for $\Delta u_{i,j,k}$, $\Delta v_{i,j,k}$, $\Delta w_{i,j,k}$ can be determined by inverting a modified Eq. 3.12 and using the appropriate row of the inverted matrix as weights for the residual vector. As the inclusion of the additional terms resulting from an implicit convection or diffusion scheme in the SCGS-PP algorithm (Eq. 3.11) did not result in a significant difference in the rate of convergence, these terms have not been included in Eq. 3.12. This is considered to be a valid assumption as
the behavior of the SCGS-PP and SCGS-PPV algorithms are very similar for both explicit and implicit convection-diffusion schemes, as will be seen when the algorithms are applied to various problems.

3.7.1 Algorithm

As Equations 3.13, 3.14 and 3.15 define a unique method to solve for the velocity in parallel, the question is what method to use to solve for the pressure. All attempts to use the artificial compressibility formulation to solve for pressure while using Eq. 3.13, Eq. 3.14 and Eq. 3.15 to solve for the velocity have been unstable. The use of Eq. 3.10 is stable and along with Eq. 3.13, Eq. 3.14 and Eq. 3.15 defines the SCGS-PPV method. One application of the SCGS-PPV algorithm is given by the following. Pseudocode which describes how the SCGS-PPV algorithm can be implemented is given below.

compute momentum and continuity equation residuals
solve for \( u_{i}^{n+1,m+1} \) using Eq. 3.13, Eq. 3.14 and Eq. 3.15
communicate \( u_{i}^{n+1,m+1} \) between different processes
compute momentum and continuity equation residuals using \( u_{i}^{n+1,m+1} \)
solve for \( p^{n+1,m+1} \) using Eq. 3.10
communicate \( p^{n+1,m+1} \) between different processes

The only synchronization points are communication routines. Other then these points, all computations can proceed independently of each other.

3.8 Colored SCGS

It is possible in certain cases to split the grid points at which pressure is solved, such that SCGS can be used to provide a unique solution for the velocity and pressure in parallel. Fig. 3.2 shows how this can be done in two dimensions. In Fig. 3.2, Eq. 3.9 or (if implicit time integration is used for the convective and or diffusive terms Eq. 3.11) can be solved at each blue grid point independently of any other points. This will give a solution for the velocity at every interior
velocity grid point and for the pressure at approximately half of the pressure grid points. If Eq. 3.9 or Eq. 3.11 is then solved at all red pressure points, pressure at the remaining pressure grid points will be solved for, along with the velocity at all interior velocity grid points. This grid coloring can be extended to three dimensions while still using two colors. Note that due to the fact that one application of colored SCGS solves the velocity field twice, the amount of computational work is approximately twice that of the SCGS-PP, SCGS-PPV or artificial compressibility algorithms.

One note regarding this method is that if a flow is solved that has no gradients in one direction, colored SCGS will generate gradients in this direction at each iteration. These gradients will of course decay as the iterative process proceeds. These gradients result from the fact that the solution of Eq. 3.9 or Eq. 3.11 at one color’s points will result in new pressures at each of that color’s pressure grid points. These pressures will generate non-zero pressure gradients when used with the other color’s unchanged pressures to compute the pressure gradients needed in computing the residuals of the momentum equations at all interior velocity grid points.

3.8.1 Implicit Convection Diffusion Time Integration

As in SCGS-PP, additional elements on the off diagonals arise as a result of an implicit
convection and or diffusion scheme (Eq. 3.11). While the expression for $\Delta p_{i,j,k}$ is the same as in the SCGS-PP algorithm, the expressions used for $\Delta u_{i-1,j,k}$, $\Delta u_{i,j,k}$, $\Delta v_{i,j-1,k}$, $\Delta v_{i,j,k}$, $\Delta w_{i,j,k-1}$, $\Delta w_{i,j,k}$ are different. In the SCGS-PP algorithm they are found using Eq. 3.6. In the colored SCGS algorithm they are found from solving Eq. 3.11. After $\Delta p_{i,j,k}$ is known this results in $2 \times 2$ linear systems for the pairs $(\Delta u_{i-1,j,k}, \Delta u_{i,j,k})$, $(\Delta v_{i,j-1,k}, \Delta v_{i,j,k})$ and $(\Delta w_{i,j,k-1}, \Delta w_{i,j,k})$. It has been found that the inclusion of the additional elements on the off diagonals arising from implicit convection or diffusion schemes makes little difference in the residual history. As these terms involve significant additional computations and storage, they have not been included in the present work.

3.9 Fourier Mode Analysis

An important method of evaluating the properties of an iterative scheme is the Fourier Mode Analysis. The basic idea is to represent the error as a Fourier series and examine the effect of the iterative scheme in Fourier (frequency) space.

Any stationary iterative scheme to solve $Au = b$ can be written as

$$C(u^{m+1} - u^m) = b - Au^m \text{ or } u^{m+1} = C^{-1}(b + (C - A)u^m) = C^{-1}b + (I - C^{-1}A)u^m$$

where $m$ is the iteration count.

Defining the current approximation to the solution $u^m$ as $u^m = \varepsilon^m + u$, where $\varepsilon^m$ is the error, $u$ is the exact solution and substituting results in

$$\varepsilon^{m+1} + u = C^{-1}b + (I - C^{-1}A)(\varepsilon^m + u) = C^{-1}b + I\varepsilon^m - C^{-1}A\varepsilon^m + Iu - C^{-1}Au = C^{-1}(b - Au) + (I - C^{-1}A)\varepsilon^m + Iu$$

As $Au = b$ this reduces to

$$\varepsilon^{m+1} = (I - C^{-1}A)\varepsilon^m$$

Therefore, the behavior of the error is a function only of the matrix $(I - C^{-1}A)$ and does not depend on the right hand side vector $b$. Any grid function (e.g. the solution $u_{i,j,k}$ or the error $\varepsilon_{i,j,k}$) can be written as (Wesseling, 1992):
\[ u_{i,j,k} = \sum_{k_z=1}^{N_z-1} \sum_{k_y=1}^{N_y-1} \sum_{k_x=1}^{N_x-1} a_{k_x,k_y,k_z} \sin\left(\frac{i\pi k_x}{N_x}\right) \sin\left(\frac{j\pi k_y}{N_y}\right) \sin\left(\frac{k\pi k_z}{N_z}\right) \] (3.16)

where

\[ a_{k_x,k_y,k_z} = \sum_{k=1}^{N_z-1} \sum_{j=1}^{N_y-1} \sum_{i=1}^{N_x-1} u_{i,j,k} \sin\left(\frac{i\pi k_x}{N_x}\right) \sin\left(\frac{j\pi k_y}{N_y}\right) \sin\left(\frac{k\pi k_z}{N_z}\right) \] (3.17)

An analysis of how iterative schemes affect the Fourier spectrum can now be made by comparing the resulting Fourier coefficients of the error with those of the original error. Of particular interest is the ratio of the magnitude of the resulting Fourier coefficients to the original Fourier coefficients, at each frequency or wave number. This is known as the smoothing factor and is given by Eq. 3.18.

\[ \rho_{m}^{k_x,k_y,k_z} = \left| \frac{a_{k_x,k_y,k_z}^{m}}{a_{k_x,k_y,k_z}^{original}} \right| \] where \( m \in [1, \infty) \) is the iteration index (3.18)

### 3.9.1 Model Problem

A model problem is constructed with Dirichlet boundary conditions on the boundary \( \Gamma \). The boundary conditions are \( \vec{u} = 0 \) on \( \Gamma \). No boundary conditions are needed for pressure. The initial conditions at the previous time steps are \( \vec{u} = 0 \). Pressure is treated fully implicitly and therefore no initial conditions are needed for it. The solution to this problem is \( \vec{u} = 0 \) in the interior and pressure equal to a constant. As a result, the intermediate solution at any point in the iterative process is the error. Since there is no error on \( \Gamma \), the error is defined only in the interior. As regards the discrete grid this means that the error is defined from the first to the last grid point in each coordinate direction excluding those points that lie on \( \Gamma \). An error field is generated by use of Eq. 3.16 with \( a_{k_x,k_y,k_z} = 1 \) for the velocity and pressure. This error is imposed as an initial guess at the current time step and the iterative scheme is applied a number of times. The resulting velocity and pressure field (which is the error field) is then transformed to Fourier space by use of Eq. 3.17. As the pressure can only be determined up to a constant, it must be treated slightly differently from the velocity. After using Eq. 3.16 with \( a_{k_x,k_y,k_z} = 1 \) to generate a pressure field,
a constant is added to the field such that the pressure at one point is zero. The resulting field is then transformed to Fourier space by use of Eq. 3.17 and stored. After one application of the iterative scheme is made, the resulting pressure field is adjusted by a constant so that the pressure at the same point is zero. This pressure field is then transformed to Fourier space at which point Eq. 3.18 can be used.

3.9.2 Results of the Model Problem

Figure 3.3 shows isosurfaces of the smoothing factor for $u$ as a function of wave number space for the four different iterative algorithms. As expected for methods that involve local smoothing, all methods show the highest value of the smoothing factor (Eq. 3.18) at low wave numbers (near the origin in wave number space). The artificial compressibility method has higher smoothing factors compared to the other algorithms, while the SCGS-PP and SCGS-PPV algorithms have indistinguishable smoothing factors.

Figure 3.4 shows isosurfaces of the smoothing factor for $p$ as a function of wave number for four the different iterative schemes. The isosurfaces form planes oriented towards the origin. The isosurface furthest from the origin is associated with very low values of the smoothing factor. It can be seen that the smoothing factor for $p$ is more than two orders of magnitude larger than the smoothing factor for $u$. Again, the artificial compressibility method has higher smoothing factors compared to the other algorithms, while the SCGS-PP and SCGS-PPV algorithms have indistinguishable smoothing factors.

An interesting observation as regards the colored SCGS scheme is that, for an error field with $\alpha_{k_x, k_y, k_z} = 1$ in Eq. 3.16 for the velocity and pressure, the exact solution can be obtained after one iteration, if the grid colors are solved in one of the two possible orders and no underrelaxation is used. This is true only if the pressure gradient and continuity equation finite difference stencils have two-points.
The reason for this has not been thoroughly investigated. A possible explanation is as follows. The error in the model problem’s pressure field is zero, for the six red grid pressure points surrounding each of the blue grid pressure points in which the error in the pressure field is not zero. If a two-point finite difference stencil is used for the pressure gradient terms in the momentum equations and for the continuity equation, then the blue grid pressure points will only see the six surrounding pressures, which are at the correct value, and the result will be the correct velocity (zero) at all velocity grid points and the correct pressure field at all the blue grid pressure points, if the blue grid is solved first. For the model problem results in the present work, the grid colors are solved in the order that does not result in the exact solution after one iteration.
Figure 3.4: Isosurfaces of $\rho_p$ on a $16 \times 16 \times 16$ grid
Figure 3.5 shows the maximum and average smoothing factors as a function of iteration number. The average smoothing factor is defined by taking the average of the smoothing factors within a certain wave number range (Eq. 3.19).

$$\bar{\rho}_{average} = \frac{1}{M} \sum_{k_x=1}^{N_x-1} \sum_{k_y=1}^{N_y-1} \sum_{k_z=1}^{N_z-1} \left| \rho_{k_x, k_y, k_z} \right| \text{ for } \left[ (k_x^{max} - k_x)^2 + (k_y^{max} - k_y)^2 + (k_z^{max} - k_z)^2 \right]^{\frac{1}{2}} \leq 10$$

where $M$ is the number of times $\left[ (k_x^{max} - k_x)^2 + (k_y^{max} - k_y)^2 + (k_z^{max} - k_z)^2 \right]^{\frac{1}{2}} \leq 10$

While it is true that the maximum smoothing factor will determine the rate of convergence in general, this will not be true for an iterative method that contains the highest value of Eq. 3.18 at low wave numbers, if the error has no component at low wave numbers. This will be the case for
unsteady flows that have reached a statistically fully developed state and in which small time steps are being taken to advance the solution. In this case, the initial guess from the previous time step, which determines the error, is very good and will contain mainly error at high wave numbers. Note that the smoothing factors for the pressure in Fig. 3.5 are identical for the SCGS-PP and SCGS-PPV algorithms, and that both are better than the artificial compressibility algorithm.

Fig. 3.6 shows the residual as a function of iteration number. It can be seen that SCGS-PPV is significantly better than SCGS-PP or artificial compressibility in reducing the error of the momentum equations. The best method appears to be colored SCGS. However, this method involves twice as much computational work per iteration as other methods. More importantly, as colored SCGS relies on a grid splitting or coloring, the author is not aware of any way to color an arbitrary collection of Cartesian multi-block grids, such as that used in the following example of flow over a backstep. Also, while it has not been done in the present work, there should be no reason why the two new methods could not be used in a curvilinear coordinate system. In this case, it is not possible to color a collection of multi-block curvilinear grids connected in such a way that the underlying block structure is not Cartesian (i.e. if in two (three) dimensions more or less than four (eight) blocks meet at a point). As a result, colored SCGS cannot be used in this case. This is illustrated in Fig. 3.7 which shows a two-dimensional multi-block structured grid, consisting of three blocks. The non-Cartesian nature of the underlying block structure is shown by the black lines. In Fig. 3.7 two different solutions will be obtained at the velocity points indicated by the arrows. This is because along the boundary on which the arrows lie, red points are paired with red points instead of with blue points (and vice versa for the blue points).

3.10 Computational Work

The computational work involved in the application of an algorithm is an important factor in weighing its usefulness. It is difficult to obtain an accurate measure of the computational work involved in an algorithm which will be used on different types of computers, since
Figure 3.6: Residual history on a $16 \times 16 \times 16$ grid

Figure 3.7: Grid that colored SCGS cannot be used on
computational work is a function of the computer architecture, the programming methodology and the optimizations performed by the compiler. An estimate can be made by comparing the number of floating point operations required by each algorithm. Table 3.1 lists the number of floating point operations (Flops) used by each algorithm in solving for the velocity and pressure field. It should be noted that Table 3.1 only includes the operations needed to solve for the velocity and pressure field at each time step. It does not include the operations needed for the convection and diffusion terms as the number of operations for these terms can vary widely depending on what spatial differencing scheme is used for these terms. Also, the operations required by the explicit time integration scheme used are not included. These other operations are not included as they are the same for each of the algorithms. Table 3.2 contains the ratios of the wall clock time of the algorithms as applied on a $32 \times 32 \times 32$ grid. The fact that the number of operations and wall clock time required by SCGS-PPV is much greater than any others, coupled with the fact that it has essentially the same behavior as SCGS-PP (as will be shown later) results in SCGS-PP being the preferred algorithm of the two. As each application of the colored SCGS method solves for the velocity twice, the number of subiterations used for the colored SCGS algorithm is always half that of the other methods in order to give a fair comparison.
3.11 Results

In this section, the artificial compressibility, SCGS-PP, SCGS-PPV and colored SCGS algorithms are evaluated by applying them to the DNS of a channel flow and a driven cavity flow. In addition, the SCGS-PP algorithm is used for the DNS of flow over a backstep.

3.11.1 DNS of Channel Flow

3.11.1.1 Problem Description

A channel flow at $Re_{r}$ of 180 is simulated. The flow is driven by a constant body force term which is added to the $x$ momentum equation. All quantities are non-dimensionalized by the channel half height $h$ and the friction velocity $u_{r} = \sqrt{\nu \frac{\partial \bar{u}}{\partial y} |_{y=0}}$. A schematic of the flow domain along with boundary conditions and dimensions is given in Fig. 3.8. The computational domain is 6.4 units in the streamwise direction ($x$), 2 units in the wall normal direction and 3.2 units in the cross-stream direction ($z$). The grid dimensions are $128 \times 128 \times 128$. Evenly spaced grids are used in the $x$ and $z$ directions, with grid spacings in wall units of 9 and 4.5 respectively. A stretched grid that concentrates points near the two walls is used in the $y$ direction. The first three main grid points in wall units are at .716, 2.158 and 3.671 from the wall in the $y$ direction. The maximum grid spacing in wall units in $y$ is 5.04 (at the channel center line). It has been shown (Kawamura et al., 1998) that this resolution is sufficient for this Reynolds number. It is necessary to use a method to trigger turbulence for this particular flow. The method used in the present work is to place two rows of wall jets on the top and bottom walls. These wall jets either injected or removed fluid from the domain depending on their spatial location. The net sum of mass injected into the domain was approximately zero. These wall jets were turned on until the flow (based on the bulk velocity) had traveled $12.4h$, after which they were turned off. In addition, as an initial condition, the $v$ velocity at each interior point is set to a random number between $-2$ and 2.

3.11.1.2 Numerical Scheme

A sixth-order central difference convective scheme with a monotonic Total Variation
Diminishing (TVD) type limiter is used when the explicit time integration scheme of Eq. 3.1 is used. A third-order upwind biased convective scheme is used in conjunction with the implicit time discretization scheme (Eq. A.46). Fourth-order interpolation is used for the velocity in the convective terms e.g. for \( v \) in \( \frac{\partial u}{\partial y} \) etc. A fourth-order central difference scheme is used for the diffusive terms. The pressure gradient and the continuity equation are represented by second-order centered schemes.

3.11.1.3 Explicit Scheme Results

Two different time steps (\( \Delta t = .00015 \) and \( \Delta t = .00005 \)) are used to demonstrate the effect of the time step on the various numerical parameters that are required for the artificial compressibility, SCGS-PP, SCGS-PPV and colored SCGS algorithms. For this flow the time
step of .00015 corresponds to a maximum CFL number of approximately .0645 based on the \( u \) velocity and grid spacing in the \( x \) direction. The other CFL numbers based on the other two directions are smaller. Table 3.3 shows the numerical parameters used for the four schemes. In Cases 1 and 3 the value for \( \beta \), required by the artificial compressibility method (Eq. 3.3), was chosen using a trial and error method in which \( \beta \) was continually increased until an instability set in, causing the solution to diverge. At that point \( \beta \) was decreased somewhat and this value, which is considered to be optimal, was used for the simulation. The consequences of not choosing an optimal value for \( \beta \) are shown later. In certain instances, one of the consequences is that the solution diverges. In general, the simulation had to be integrated for a couple of hundred physical time steps in order to determine whether the solution was diverging. The value of \( \beta \) found is considered optimal, considering the constraints of the author’s time involved in performing the trial and error simulations, approximately ten of which were done in order to find each optimal value of \( \beta \). The choice of the parameters \( \alpha_p, \alpha_u, \alpha_v, \alpha_w \) for the other schemes was made simply by using values that have been used by the author in solving other flows, such as jets in crossflow and flow over spheres and cylinders. Case 2 is included to demonstrate the effect of using the same value of \( \beta \) that is optimal (\( \beta = 400 \)) for a certain time step (\( \Delta t = .00015 \)) for a simulation for which a smaller time step (\( \Delta t = .00005 \)) is chosen. A different value of \( \beta \) (4000), is optimal for this lower time step (\( \Delta t = .00005 \)). Using \( \beta = 4000 \) (which is optimal for a time step of (\( \Delta t = .00005 \)) for a simulation using a larger time step of (\( \Delta t = .00015 \)), results in a diverging solution. For the artificial compressibility, SCGS-PP and SCGS-PPV methods, ten subiterations were used for a time step of (\( \Delta t = .00005 \)) and twenty for a time step of (\( \Delta t = .00015 \)). Because each application of the colored SCGS method solves for the velocity twice, the number of subiterations used for the colored SCGS algorithm is always half that of the other methods in order to give a fair comparison.
Table 3.3: Channel time step and numerical parameters, explicit time integration

<table>
<thead>
<tr>
<th>Case</th>
<th>time step and numerical parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case 1</td>
<td>artificial compressibility, $\Delta t = .00005$, $\beta = 4000$, $\alpha_u = \alpha_v = \alpha_w = .915$</td>
</tr>
<tr>
<td>Case 2</td>
<td>artificial compressibility, $\Delta t = .00005$, $\beta = 400$, $\alpha_u = \alpha_v = \alpha_w = .915$</td>
</tr>
<tr>
<td>Case 3</td>
<td>artificial compressibility, $\Delta t = .00015$, $\beta = 400$, $\alpha_u = \alpha_v = \alpha_w = .915$</td>
</tr>
<tr>
<td>Case 4</td>
<td>SCGS-PP , $\Delta t = .00005$, $\alpha_p = \alpha_u = \alpha_v = \alpha_w = .925$</td>
</tr>
<tr>
<td>Case 5</td>
<td>SCGS-PP , $\Delta t = .000015$, $\alpha_p = \alpha_u = \alpha_v = \alpha_w = .925$</td>
</tr>
<tr>
<td>Case 6</td>
<td>colored SCGS , $\Delta t = .00005$, $\alpha_p = \alpha_u = \alpha_v = \alpha_w = .975$</td>
</tr>
<tr>
<td>Case 7</td>
<td>colored SCGS , $\Delta t = .000015$, $\alpha_p = \alpha_u = \alpha_v = \alpha_w = .975$</td>
</tr>
<tr>
<td>Case 8</td>
<td>SCGS-PPV , $\Delta t = .00005$, $\alpha_p = \alpha_u = \alpha_v = \alpha_w = .925$</td>
</tr>
<tr>
<td>Case 9</td>
<td>SCGS-PPV , $\Delta t = .000015$, $\alpha_p = \alpha_u = \alpha_v = \alpha_w = .925$</td>
</tr>
</tbody>
</table>

**Residual Level**  Fig. 3.9 shows the history as a function of time of the average residual of the continuity equation at a time step of ($\Delta t = .00005$). This is defined by the arithmetic mean of the absolute value of the residual of the continuity equation at every pressure grid point. It can be seen that for the artificial compressibility method, using the value for $\beta$ that is optimal for a time step of ($\Delta t = .00015$) results in a residual level that is an order of magnitude higher than that obtained using an optimal value of $\beta$ for this time step ($\Delta t = .00005$). In addition, the value of the residual for the artificial compressibility method using an optimum value for $\beta$ is approximately 6 times larger than that resulting from the SCGS-PP and SCGS-PPV methods. Relative to the colored SCGS method, the residual is an order of magnitude larger. The residual history of the SCGS-PP and SCGS-PPV methods are very similar. This confirms what was found when the smoothing properties of the schemes were examined. The spike in the graph is the result of turning off the wall jets used to trigger the transition to turbulence. The same trends exist for the time step of ($\Delta t = .00015$), as can be seen in Fig. 3.10.

**Statistics**  In order to characterize turbulent flows it is necessary to collect statistics that are the result of time averaging various quantities. As the flow field is homogenous in the $x$ and $z$ directions, averaging is carried out in both directions. All cases were run until the flow (based on the bulk velocity) had traveled $135h$. Statistics were collected at each time step after the flow (based on the bulk velocity) had traveled $27h$. Due to symmetries, there are certain statistical
quantities which are zero for this flow. For instance, the mean value of the velocity \( v \) in the wall normal direction should be zero. Fig. 3.11 shows the mean velocity for the four methods at a time step of \((\Delta t = .00005)\). It can be seen that the artificial compressibility method in general results in larger errors. The use of a non-optimized value for \( \beta \) results in significantly larger errors. The instantaneous value of \( v \) integrated over the entire \( xz \) plane at any location in the \( y \) direction is zero as a result of conservation of mass. Fig. 3.12a and Fig. 3.12b show the time history of this quantity (from an \( xz \) plane at \( y = h \)). It can be seen that the errors in this quantity take a significantly longer time to decay for the artificial compressibility method, particularly if a non-optimized value for \( \beta \) is used. The value of \( v \) integrated over the entire \( xz \) plane is positive in the beginning of the simulations because the net mass flow into the domain, due to the wall jets, is not exactly zero. The sudden changes in Fig. 3.12 are the result of turning off the wall jets, after which the value quickly begins to oscillate around the correct value of zero.
Figure 3.10: Average residual of the continuity equation, channel flow, $\Delta t = .00015$
Figure 3.11: Mean $v$, channel flow, $\Delta t = 0.0005$

Figure 3.12: Channel flow, explicit scheme, spatially averaged $v$ in $xz$ plane, (a) Artificial compressibility, (b) SCGS-PP, SCGS-PPV and colored SCGS
3.11.1.4 Implicit Scheme Results

Two different time steps, \((\Delta t = .0015)\) and \((\Delta t = .0005)\), are used to demonstrate the effect of the time step on the various numerical parameters that are required for the artificial compressibility, SCGS-PP, SCGS-PPV and colored SCGS algorithms. For this flow the time step of \(.0015\) corresponds to a maximum CFL number of approximately \(.64\) based on the \(u\) velocity and grid spacing in the \(x\) direction. The other CFL numbers based on the other two directions are smaller. Table 3.4 shows the numerical parameters used for the four schemes. It is necessary to use smaller under-relaxation factors for the implicit time integration scheme than for the explicit time integration scheme. This is because a nonlinear system of equations must be solved at each physical time step if an implicit time integration scheme is used for the convective terms as in Eq. A.46. In Cases 10 and 12 the value for \(\beta\), required by the artificial compressibility method (Eq. 3.3), was chosen by the same trial and error method as used for the explicit time integration scheme. As for the explicit time integration scheme, a couple of hundred physical time steps were needed to determine whether the solution was diverging. The choice of the parameters \(\alpha_p, \alpha_u, \alpha_v, \alpha_w\) for the other schemes was made simply by using values that the author has used in the past for other flows, such as jets in crossflow and flow over spheres and cylinders. Case 2 is included to demonstrate the effect of using the same value of \(\beta\) that is optimal for a certain time step \((\Delta t = .0015)\) for a simulation for which a smaller time step \((\Delta t = .0005)\) is chosen. Using \(\beta = 575\) (which is optimal for a time step of \((\Delta t = .0005)\)) for a simulation using a larger time step of \((\Delta t = .0015)\), results in a diverging solution. For the artificial compressibility, SCGS-PP and SCGS-PPV methods, sixty subiterations were used for both the time steps. As each application of the colored SCGS method solves twice for the velocity, the number of subiterations used for the colored SCGS algorithm is always half that of the other methods in order to give a fair comparison.
Table 3.4: Channel time step and numerical parameters, implicit time integration

<table>
<thead>
<tr>
<th>Case</th>
<th>time step and numerical parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case 10</td>
<td>artificial compressibility, $\Delta t = .0005$, $\beta = 575$, $\alpha_u = \alpha_v = \alpha_w = .4$</td>
</tr>
<tr>
<td>Case 11</td>
<td>artificial compressibility, $\Delta t = .0005$, $\beta = 125$, $\alpha_u = \alpha_v = \alpha_w = .4$</td>
</tr>
<tr>
<td>Case 12</td>
<td>artificial compressibility, $\Delta t = .0005$, $\beta = 125$, $\alpha_u = \alpha_v = \alpha_w = .4$</td>
</tr>
<tr>
<td>Case 13</td>
<td>SCGS-PP, $\Delta t = .0005$, $\alpha_p = .225$, $\alpha_u = \alpha_v = \alpha_w = .8$</td>
</tr>
<tr>
<td>Case 14</td>
<td>SCGS-PP, $\Delta t = .0015$, $\alpha_p = .225$, $\alpha_u = \alpha_v = \alpha_w = .8$</td>
</tr>
<tr>
<td>Case 15</td>
<td>colored SCGS, $\Delta t = .0005$, $\alpha_p = \alpha_u = \alpha_v = \alpha_w = .725$</td>
</tr>
<tr>
<td>Case 16</td>
<td>colored SCGS, $\Delta t = .0015$, $\alpha_p = \alpha_u = \alpha_v = \alpha_w = .725$</td>
</tr>
<tr>
<td>Case 17</td>
<td>SCGS-PPV, $\Delta t = .0005$, $\alpha_p = .225$, $\alpha_u = \alpha_v = \alpha_w = .8$</td>
</tr>
<tr>
<td>Case 18</td>
<td>SCGS-PPV, $\Delta t = .0015$, $\alpha_p = .225$, $\alpha_u = \alpha_v = \alpha_w = .8$</td>
</tr>
</tbody>
</table>

**Residual Level**  Fig. 3.13 shows the history as a function of time of the average residual of the continuity equation at a time step of ($\Delta t = .0005$). For the artificial compressibility method, using the value for $\beta$ that is optimal for a time step of ($\Delta t = .0015$) results in a residual level that is somewhat higher than that obtained using an optimal value of $\beta$ for this time step ($\Delta t = .0005$). The difference, however, is not nearly as great as in the explicit time integration cases. As in the explicit time integration cases, the residual history of the SCGS-PP and SCGS-PPV methods are very similar. The spike in the graph is the result of turning off the wall jets used to trigger the transition to turbulence. For the time step of ($\Delta t = .0015$) a slightly lower residual is obtained using the artificial compressibility algorithm (Fig. 3.14). However, note that this occurs only after using an optimal value for $\beta$ which was obtained by a painstaking trial and error process. The reason for the small differences between the algorithms in the residual history are may be due to the fact that at these higher CFL numbers more of the error exists in the low wave number range. In this range, the smoothing properties of the algorithms approach each other for the linear system resulting from explicit time integration as was shown earlier. It is to be expected that this behavior also exists for the nonlinear system resulting from implicit time integration.

**Statistics**  All implicit cases were run for the same length of time and had statistics collected over the same period of time, as the explicit cases. Fig. 3.15 shows the mean $v$ for the four methods at a time step of ($\Delta t = .0005$). As for the explicit cases, using the value for $\beta$ that is optimal
for a time step of $(\Delta t = .0015)$ results in significantly greater error than that obtained using the optimal value of $\beta$ for this time step $(\Delta t = .0005)$. The instantaneous value of $v$ integrated over the entire $xz$ plane at any location in the $y$ direction is zero as a result of conservation of mass. Fig. 3.16a and Fig. 3.16b show the time history of the instantaneous value of $v$ integrated over the entire $xz$ plane (at $y = h$). It can be seen that the errors in this quantity take significantly longer to decay for the artificial compressibility method, if a non-optimized value for $\beta$ is used. The value of $v$ integrated over the entire $xz$ plane is positive in the beginning of the simulations because the net mass flow into the domain, due to the wall jets, is not exactly zero. The sudden changes in Fig. 3.16 are the result of turning off the wall jets, after which the value quickly begins to oscillate around the correct value of zero.
Figure 3.14: Average residual of the continuity equation, channel flow, $\Delta t = .0015$
Figure 3.15: Mean $v$, channel flow, $\Delta t = .0005$

Figure 3.16: Channel flow, implicit scheme, spatially averaged $v$ in $xz$ plane, (a) Artificial compressibility, (b) SCGS-PP, SCGS-PPV and colored SCGS
3.11.2 DNS of Driven Cavity

A driven cavity flow at a Reynolds number of 10000 based on the moving wall velocity and the cavity height is simulated. The flow is driven by the movement of the top wall in the $x$ direction. All quantities are non-dimensionalized by the cavity height $h$ and the velocity of the top wall. A schematic of the flow domain along with boundary conditions and dimensions is given in Fig. 3.17. The grid dimensions are $64 \times 64 \times 64$. Stretched grids that concentrate points near the walls are used in all three directions. For the velocity field, the initial conditions used are that all three components of velocity are set to a random number between $-\frac{1}{2}$ and $\frac{1}{2}$. The initial condition for pressure is to set it to zero everywhere.

![Figure 3.17: Driven cavity geometry and boundary conditions](image)

3.11.2.1 Numerical Scheme

The same numerical schemes used for the channel flow are used for the driven cavity flow.

3.11.2.2 Explicit Scheme Results

Two different time steps, $(\Delta t = .0012)$ and $(\Delta t = .0003)$, are used to demonstrate the effect of the time step on the numerical parameters of the four schemes. For this flow, the time step
Table 3.5: Driven cavity time step and numerical parameters, explicit time integration

<table>
<thead>
<tr>
<th>Case</th>
<th>time step and numerical parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case 1</td>
<td>artificial compressibility, $\Delta t = .0003, \beta = 60, \alpha_u = \alpha_v = \alpha_w = .915$</td>
</tr>
<tr>
<td>Case 2</td>
<td>artificial compressibility, $\Delta t = .0003, \beta = 4.25, \alpha_u = \alpha_v = \alpha_w = .915$</td>
</tr>
<tr>
<td>Case 3</td>
<td>artificial compressibility, $\Delta t = .0012, \beta = 4.25, \alpha_u = \alpha_v = \alpha_w = .915$</td>
</tr>
<tr>
<td>Case 4</td>
<td>SCGS-PP, $\Delta t = .0003, \alpha_p = \alpha_u = \alpha_v = \alpha_w = .925$</td>
</tr>
<tr>
<td>Case 5</td>
<td>SCGS-PP, $\Delta t = .0012, \alpha_p = \alpha_u = \alpha_v = \alpha_w = .925$</td>
</tr>
<tr>
<td>Case 6</td>
<td>colored SCGS, $\Delta t = .0003, \alpha_p = \alpha_u = \alpha_v = \alpha_w = .975$</td>
</tr>
<tr>
<td>Case 7</td>
<td>colored SCGS, $\Delta t = .0012, \alpha_p = \alpha_u = \alpha_v = \alpha_w = .975$</td>
</tr>
<tr>
<td>Case 8</td>
<td>SCGS-PPV, $\Delta t = .0003, \alpha_p = \alpha_u = \alpha_v = \alpha_w = .925$</td>
</tr>
<tr>
<td>Case 9</td>
<td>SCGS-PPV, $\Delta t = .0012, \alpha_p = \alpha_u = \alpha_v = \alpha_w = .925$</td>
</tr>
</tbody>
</table>

of .0012 corresponds to a maximum CFL number of approximately .16 based on the $u$ velocity and grid spacing in the $x$ direction. Table 3.5 shows the numerical parameters used for the four schemes. In Cases 1 and 3 the value for $\beta$ required by the artificial compressibility method (Eq. 3.3) was chosen using the same trial and error method used for the channel flow. The same parameters $\alpha_p, \alpha_u, \alpha_v, \alpha_w$ that were used for the channel flow are used for this flow. Case 2 is included to demonstrate the effect of using the same value of $\beta$ that is optimal for a certain time step ($\Delta t = .0012$) for a simulation for which a different time step ($\Delta t = .0003$) is chosen. Using the value of ($\beta = 60$), which is optimal for a time step of ($\Delta t = .0003$) for a simulation using a time step of ($\Delta t = .0012$) results in a diverging solution for which no results could be obtained. For the artificial compressibility, SCGS-PP and SCGS-PPV methods, eight subiterations were used for a time step of ($\Delta t = .0003$) and ten for a time step of ($\Delta t = .0012$). Because each application of the colored SCGS method solves for the velocity twice, the number of subiterations used for the colored SCGS algorithm is always half that of the other methods in order to give a fair comparison.

**Residual level** Fig. 3.18 shows the history as a function of time of the average residual of the continuity equation at a time step of ($\Delta t = .0003$). This is defined by the arithmetic mean of the absolute value of the residual of the continuity equation at every pressure grid point. It can be seen that for the artificial compressibility method using the value for $\beta$ that is optimal for a
time step of \((\Delta t = .0012)\) results in a residual level that is an order of magnitude higher than that obtained using an optimal value of \(\beta\) for this particular time step.

![Graph showing average residual of the continuity equation, driven cavity, \(\Delta t = .0003\)](image)

**Figure 3.18:** Average residual of the continuity equation, driven cavity, \(\Delta t = .0003\)

**Other Measures of Error**  A measure of error that can be used for this flow is the mass flow through a plane that cuts through the entire cavity. Due to the boundary conditions for this flow, the mass flow through this plane must be zero. For convenience, the plane chosen is the \(yz\) plane in the center of the cavity (at \(x = .5h\)). Fig. 3.19 shows the behavior of this integrated quantity as a function of time. A dramatic increase in the error for the artificial compressibility method can be seen if a non-optimized value for \(\beta\) is used. The artificial compressibility method used with the optimized value of \(\beta\) has a larger error than the other three methods, which have practically identical behavior as regards this measure of error.
Figure 3.19: Mass flow through \(yz\) plane at \(x = 0.5h\), driven cavity

Table 3.6: Driven cavity time step and numerical parameters, implicit time integration

<table>
<thead>
<tr>
<th>Case</th>
<th>time step and numerical parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case 10</td>
<td>artificial compressibility, (\Delta t = 0.003, \beta = 2.6), (\alpha_u = \alpha_v = \alpha_w = 0.85)</td>
</tr>
<tr>
<td>Case 11</td>
<td>artificial compressibility, (\Delta t = 0.003, \beta = 0.75), (\alpha_u = \alpha_v = \alpha_w = 0.85)</td>
</tr>
<tr>
<td>Case 12</td>
<td>artificial compressibility, (\Delta t = 0.006, \beta = 0.75), (\alpha_u = \alpha_v = \alpha_w = 0.85)</td>
</tr>
<tr>
<td>Case 13</td>
<td>SCGS-PP, (\Delta t = 0.003, \alpha_p = \alpha_u = \alpha_v = \alpha_w = 0.7)</td>
</tr>
<tr>
<td>Case 14</td>
<td>SCGS-PP, (\Delta t = 0.006, \alpha_p = \alpha_u = \alpha_v = \alpha_w = 0.7)</td>
</tr>
<tr>
<td>Case 15</td>
<td>colored SCGS, (\Delta t = 0.003, \alpha_p = \alpha_u = \alpha_v = \alpha_w = 0.95)</td>
</tr>
<tr>
<td>Case 16</td>
<td>colored SCGS, (\Delta t = 0.006, \alpha_p = \alpha_u = \alpha_v = \alpha_w = 0.95)</td>
</tr>
<tr>
<td>Case 17</td>
<td>SCGS-PPV, (\Delta t = 0.003, \alpha_p = \alpha_u = \alpha_v = \alpha_w = 0.7)</td>
</tr>
<tr>
<td>Case 18</td>
<td>SCGS-PPV, (\Delta t = 0.006, \alpha_p = \alpha_u = \alpha_v = \alpha_w = 0.7)</td>
</tr>
</tbody>
</table>
3.11.2.3 Implicit Scheme Results

For this flow, the time step of .006 corresponds to a maximum CFL number of approximately .85 based on the $v$ velocity and grid spacing in the $y$ direction. Table 3.6 shows the numerical parameters used for the four schemes. As in the channel flow, it is necessary to use smaller under-relaxation factors for the implicit time integration scheme than for the explicit time integration scheme. As demonstrated previously, using a value of $\beta$ that is optimal for a smaller time step ($\Delta t = .003$) in a simulation with a larger time step ($\Delta t = .006$) results in a solution that quickly diverges. For the artificial compressibility, SCGS-PP and SCGS-PPV methods, twenty-four subiterations were used for a time step of ($\Delta t = .003$) and thirty for a time step of ($\Delta t = .006$). Because each application of the colored SCGS algorithm solves for the velocity twice, the number of subiterations used for the colored SCGS algorithm is always half that of the other methods in order to give a fair comparison.

Fig. 3.20 shows the history as a function of time of the average residual of the continuity equation at a time step of ($\Delta t = .003$). It can be seen that for the artificial compressibility method using the value for $\beta$ that is optimal for a time step of ($\Delta t = .0012$) results in a residual level that is an order of magnitude higher than that obtained using an optimal value of $\beta$ for this particular time step ($\Delta t = .003$). The artificial compressibility method using the optimal value of $\beta$ performs worse than the other three methods.

3.11.3 DNS of Flow Over a Backstep

This flow is a good test of a numerical method as it has a strong separation and recirculation region. All quantities are non-dimensionalized by the step height $h$ and the freestream velocity $U_0$. A schematic of the flow domain along with boundary conditions and dimensions is given in Fig. 3.21. Referring to Fig. 3.21 $L_f = 5h$, $L_p = 4h$, $L_up = 10h$, and $L_down = 20$. The Reynolds number based on freestream velocity and step height is 5000. The domain is decomposed into 128 Cartesian zones. Each zone is assigned to a processor on a parallel computer. The wall clock
Figure 3.20: Average residual of the continuity equation, driven cavity, $\Delta t = .003$

time per real time step while collecting all second-order statistics is .539 seconds on a cluster of dual Xeons with a Myrinet interconnect.

3.11.3.1 Boundary Conditions

Providing realistic inflow boundary conditions for a turbulent flow is a challenging part of a simulation. The correct method is to trip the boundary layer while extending the upstream domain far enough that the correct spatially developed boundary layer results. (Le et al., 1997) (referred to as LMK in figures) imposed a fluctuating component with a prescribed energy spectrum onto a mean profile. In this study the mean boundary layer is tripped by means of a row of hypercubes just downstream of the inlet. The hypercubes are represented by the immersed boundary method (Fadlun et al., 2000) (Peskin, 1977). Due to the relatively coarse grid near the inlet, only the
velocity points were defined as immersed boundary points. This coarse grid precludes any meaningful resolution of the hypercubes, which simply function as generic obstacles in the flow field. At the inlet a spline fit of the experimental data of (Jovic & Driver, 1994) (Jovic & Driver, 1995) (Driver & Jovic, 1998) (referred to as JD in figures) was imposed on $u$ and $v$ while $w$ was set to zero. All velocity components were set to zero on all walls. At the freestream boundary $y = L_f + h$ the following conditions are imposed:

$$\frac{\partial u}{\partial y} = \frac{\partial w}{\partial y} = v = 0$$

At the outlet a convective boundary condition (Eq. 3.20) is used for all three velocity components. $U_c$ is a constant chosen such that when multiplied by the area of the outflow plane the result equals the total volume flow into the domain. This boundary condition allows structures
to pass out of the domain without generating significant reflections or gradients that could affect events upstream.

\[ \frac{\partial u_i}{\partial t} + U_c \frac{\partial u_i}{\partial x} = 0 \]  

(3.20)

### 3.11.3.2 Numerical Issues

The SCGS-PP algorithm is used as the solver. Five subiterations of the algorithm are done at each physical time step. The physical time step is \( 0.0005 \), resulting in a maximum CFL number of \( \approx 0.15 \) at \( x = 1 \times 10^{-3}, y = 2.3 \). The first three \( u \) velocity grid spacings in the \( x \) direction from the step face are at \( 3.732 \times 10^{-3}, 3.924 \times 10^{-3} \) and \( 4.125 \times 10^{-3} \). The first three \( v \) velocity grid spacings in the \( y \) direction from the upstream wall are at \( 5.256 \times 10^{-3}, 3.777 \times 10^{-3} \) and \( 6.342 \times 10^{-3} \). The first three \( v \) velocity grid spacings in the \( x \) direction from the downstream wall are at \( 5.920 \times 10^{-3}, 6.142 \times 10^{-3} \) and \( 6.371 \times 10^{-3} \). The step size in the \( z \) direction is \( 5.26 \times 10^{-3} \). The initial conditions are \( u \) set equal to the freestream value upstream of the step and to \( U_c \) in Eq. 3.20 downstream of the step. At all interior points \( v \) and \( w \) and \( p \) are set to zero.

A third-order upwind biased convective scheme is used along with fourth-order interpolation of velocity e.g. for \( v \) in \( v \frac{\partial u}{\partial y} \) etc. A fourth-order scheme is used for the diffusive terms. The pressure gradient and the continuity equation are represented by second-order centered schemes. The time integration is fully explicit second-order Adams-Bashforth.

### 3.11.3.3 Statistics

Because the flow field is homogenous in the \( z \) direction, averaging is carried out in this direction. The statistical averages of the various quantities are computed as a running average. The solution is advanced for \( 180000 \) time steps before statistics are collected. This corresponds to traveling \( 90h \) at the freestream velocity. The statistics are averaged over \( 460700 \) time steps. This corresponds to traveling \( 230.35h \) at the freestream velocity.

Excellent agreement with DNS data (Le et al., 1997) and experimental data (Jovic & Driver, 1994), (Jovic & Driver, 1995) and (Driver & Jovic, 1998) is seen for the \( u \) profile in Fig. 3.22.
In particular, the inlet boundary condition results in a correct $u$ profile upstream of the backstep. The streamwise Reynolds stress component is shown in Fig. 3.24. The $\overline{u'v'}$ component of the Reynolds stress tensor is shown in Fig. 3.25. Results agree well with (Le et al., 1997), (Jovic & Driver, 1994), (Jovic & Driver, 1995) and (Driver & Jovic, 1998). A significant discrepancy exists in both the present DNS and the DNS of (Le et al., 1997) as regards the experimental values for $v$ mean Fig. 3.23. Both the present DNS and the DNS of (Le et al., 1997) show good agreement with each other but only qualitative agreement with the experimental data. Note that the values are relatively small and it is possible that the uncertainties at these low magnitudes contribute to the observed differences.

The mean reattachment length is found to be $6.278h$. It is determined by locating the first pair (in the $x$ direction) of grid points closest to (but not on) the wall where the sign of the mean streamwise velocity changes. The mean reattachment length as measured by (Jovic & Driver, 1994) (Jovic & Driver, 1995) and (Driver & Jovic, 1998) is $6h$. The value computed by (Le et al., 1997) is $6.28h$.

3.12 Conclusion

Two new parallel algorithms, SCGS-PP and SCGS-PPV, have been developed which have the important practical advantage of relaxation parameters that do not change significantly depending on the specific flow being solved. By contrast, the artificial compressibility parameter $\beta$ is difficult to choose as it can vary three orders of magnitude depending on the flow and the pseudo and physical time steps. While a value for $\beta$ can be found for which the iterative method will converge efficiently (although less so than the proposed method), finding this value requires a significant amount of time. Simply choosing an arbitrary value for $\beta$ is likely to result in an iterative solution that quickly diverges. Of the two new algorithms (SCGS-PP and SCGS-PPV) SCGS-PP is to be preferred as it performs as well as SCGS-PPV and requires significantly less computational effort. The smoothing properties of SCGS-PP make it suitable for solving DNS
fl
ows. Unlike the original SCGS algorithm, SCGS-PP and SCGS-PPV are suitable for use on parallel computers which are commonly used for computing DNS flows. SCGS-PP has been used to perform a DNS of three flows, two of which contain a strong recirculation region. Compared to the colored SCGS algorithm, SCGS-PP and SCGS-PPV do not require the ability to split or color the grid. While it has not been done in the present work, we believe there is no reason that the two new methods could not be used in a multi-block curvilinear grid in which it is not possible to color the grid and therefore use the colored SCGS algorithm. The slight increase in computational expense per iteration in the SCGS-PP algorithm as compared to the artificial compressibility algorithm is more than compensated for by the increased rate of convergence and the ease of choosing the relaxation parameters as compared to the difficulty in choosing $\beta$. 

Figure 3.22: Averaged $u$ velocity at different $x/h$ locations
Figure 3.23: Averaged $v$ velocity at different $x/h$ locations
Figure 3.24: $u'u'$ at different $x/h$ locations
Figure 3.25: $\bar{u}'\bar{v}'$ at different $x/h$ locations
Chapter 4 Immersed Boundary Method

One of the most difficult tasks of Computational Fluid Dynamics is the generation of a grid around the object being modeled. This has always involved a high degree of user interaction. If the object is moving or deforming, then the necessary regridding is a challenging and computationally expensive problem. The immersed boundary method (Peskin, 1977) (Goldstein et al., 1993) (Fadlun et al., 2000) (Beyer, 1992) has the potential to simplify these problems associated with the grid. In particular, the computation of the grid around the object being modeled can be almost completely automated, once the object has been defined. In the immersed boundary method the Navier-Stokes equations are generally solved on a Cartesian grid. The Cartesian grid enables the use of efficient numerical methods that can be parallelized relatively easily. The influence of objects on the flow is simulated by the addition of source terms to the Navier-Stokes equations.

The pressure velocity coupling in the solution of the unsteady incompressible Navier Stokes equations has long been a computationally expensive part of the solution process. The basic problem is that of determining an equation to use to solve for the pressure. To the author’s knowledge the immersed boundary method applied to incompressible flows has been used only in conjunction with fractional step methods (Chorin, 1968) and the SIMPLE type methods (Patankar, 1980). In these methods, an elliptic pressure-Poisson equation is derived from the momentum and continuity equations. The solution of this Poisson equation results in a pressure field which can then be used to correct the velocity field to ensure that it is divergence free. The solution of the pressure-Poisson equation is the greatest computational expense of this class of methods. In the present work the artificial compressibility (Chorin, 1967) method and the colored SCGS method have been used to obtain solutions for the pressure and velocity.
The simulation of an object in a flow by adding forcing terms to the momentum equations does not necessarily take into account the continuity equation. As a result, the velocity field, obtained from the immersed boundary method, is generally not divergence free. As a result of the unphysical non-divergence free flow field there are problems in the solution of the Navier-Stokes equations using the artificial compressibility, SCGS, colored SCGS, SCSG-PPV or SCGS-PP methods. This paper describes a new method to ensure that the velocity field obtained from the immersed boundary method.

4.1 Nomenclature

\( a \) = change in the interpolated immersed boundary \( u \) velocity needed to make velocity field divergence free
\( b \) = change in the interpolated immersed boundary \( v \) velocity needed to make velocity field divergence free
\( c \) = change in the interpolated immersed boundary \( w \) velocity needed to make velocity field divergence free
\( \lambda \) = Lagrange multiplier
\( U \) = Residual of the equation governing \( a \) (obtained from the objective function modified with Lagrange multipliers)
\( V \) = Residual of the equation governing \( b \) (obtained from the objective function modified with Lagrange multipliers)
\( W \) = Residual of the equation governing \( c \) (obtained from the objective function modified with Lagrange multipliers)
\( \Lambda \) = Residual of the equation governing \( \lambda \) (obtained from the objective function modified with Lagrange multipliers)

4.2 Governing Equations and Discretization

4.2.1 Spatial Discretization

The governing equations of interest in the present work are the non-conservative unsteady three-dimensional Navier-Stokes equations (Eq. 2.1 and Eq. 2.2). Equation 2.1 and Eq. 2.2 are discretized on a staggered grid shown in Fig. 4.1. The grid has each pressure grid point surrounded by six velocity grid points. Pressure is not defined on non-periodic boundaries. A sixth-order accurate central difference scheme with monotonic limiters is used for the convective terms. Due to the staggered grid, none of the velocity components are stored at the same location. They are, however, needed at the same location in order to compute the parts of \( u_j \frac{\partial u_i}{\partial x_j} \) where \( i \neq j \).
They are obtained by fitting a two-dimensional Lagrange surface over the point in question. The fitting uses products of four point Lagrange polynomials and is fourth-order accurate. This results in a total of sixteen points involved in the fitting of the surface. The number of points and hence the accuracy involved is retained as the boundary of the flow domain is approached. The diffusive terms are represented by sixth-order accurate central difference schemes. The pressure gradient term and the terms in the continuity equation are represented by two-point centered stencils. The order of all stencils is retained as a boundary is approached. This is done by keeping the number of points in the stencil constant while shifting towards the boundary the point at which the derivative is evaluated. For a centered scheme this results in moving from a centered stencil to one that is biased away from the boundary. The stencils are not changed in the vicinity of an immersed boundary.

4.2.2 Temporal Discretization

The convective and diffusive terms are integrated in time using the fully explicit second-order Adams-Bashforth scheme. The pressure gradient term is evaluated implicitly at the \( n + 1 \) time step. The result is Eq. 4.1. Equation 4.1 along with Eq. 2.2 (evaluated at the \( n + 1 \) time step) are solved at each time step. The solution to this system of equations is obtained by the use of a colored version of SCGS (Vanka, 1986).

\[
\frac{u_i^{n+1} - u_i^n}{\Delta t} = -\frac{\partial p}{\partial x_i}^{n+1} + \frac{3}{2} \left[ u_j \frac{\partial u_i}{\partial x_j} + \frac{1}{Re} \frac{\partial^2 u_i}{\partial x_j \partial x_j} \right] - \frac{1}{2} \left[ u_j \frac{\partial u_i}{\partial x_j} + \frac{1}{Re} \frac{\partial^2 u_i}{\partial x_j \partial x_j} \right]
\] (4.1)

4.3 Immersed Boundaries

To simulate an object within the flow field, a forcing term \( F_i \) is added to the momentum equations.

\[
\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{1}{Re} \frac{\partial^2 u_i}{\partial x_j \partial x_j} + F_i
\]

This forcing term will be chosen so that the velocity field will satisfy the appropriate boundary conditions defined by the object. A number of different forms have been used for \( F_i \). The present
work uses the method of (Mohd-Yusof, 1997). Applying a spatial and time discretization scheme, which is represented by $H_i$, the velocity at the next time step can be written as:

$$u_i^{n+1} = H_i(u_i^{n+1}, u_i^n, u_i^{n-1}, p_i^{n+1}) + F_i$$

Denoting the desired velocity at the boundary by $\Phi_i$, the value of $F_i$ that results in the desired velocity at the boundary is:

$$F_i = \Phi_i - H_i(u_i^{n+1}, u_i^n, u_i^{n-1}, p_i^{n+1})$$

This is equivalent to setting $u_i^{n+1} = \Phi_i$ at every time step. This is trivial if the boundary of the object is coincident with the velocity grid points.

4.3.1 Selection of Immersed Boundary Points

As the grid points will generally not coincide with the immersed boundary, it is necessary to
decide at which grid points $F_i$ will be added. Grid points at which $F_i$ is added are known as immersed boundary points. It is conceivable to add the forcing to all grid points. However, as the influence of an immersed boundary on a point far away would be expected to be small, it appears to make physical and computational sense to limit the addition of $F_i$ to a subset of points near the immersed boundary. The selection scheme to determine this subset of points is as follows. All interior velocity points are examined to see if a boundary lies between them and one or more of its six neighbors $(i - 1, j, k)$, $(i + 1, j, k)$, $(i, j - 1, k)$, $(i, j + 1, k)$, $(i, j, k - 1)$, $(i, j, k + 1)$. These points are marked as possible immersed boundary points. Each mass conservation cell is then examined to see if the six velocity points that surround it are possible immersed boundary points. If they are, then all six of the points are defined as immersed boundary points. The result of this selection process is that all immersed boundary points belong to either one or two mass conservation cells that have all six velocity points determined by the immersed boundary interpolation procedure. Some investigators (Kim et al., 2001), when representing flow over an object, have chosen to add the forcing only at grid points on or inside the object. The rational for doing this is that only the flow outside the object is of interest. However, this approach prevents modeling of immersed boundaries with a thickness less then the spacing between velocity points. This approach is not used in the present work as no distinction is made between the inside or outside of an immersed boundary.

4.3.2 Basic Interpolation Scheme

Once a choice has been made of the grid points at which the forcing will be added, a value for $\Phi_i$ must be determined. The determination of $\Phi_i$ at a velocity grid point is equivalent to setting the velocity at that grid point to $\Phi_i$. As the grid points will generally not coincide with the immersed boundary, this will require an interpolation scheme. The simplest choice is one-dimensional interpolation along coordinate lines. For instance, the immersed boundary point $u_{i,j-1}$ in Fig. 4.3 can be determined by second-order accurate one-dimensional interpolation in the $+j$ direction.
using the points $\Phi^{+j}, u_{i,j}, u_{i,j+1}$. However $u_{i,j-1}$ could also be determined by second-order accurate one-dimensional interpolation in the $+i$ direction using the points $\Phi^{+i}, u_{i,j}, u_{i,j+1}$.

In general, in three dimensions there can be up to six possible directions to interpolate. The present work, in certain cases, resolves this ambiguity by interpolating in the direction in which the boundary is closest to the grid point (stencil chosen is shown in Fig. 4.3) provided the interpolation stencil meets a certain criteria. This criterion is whether the grid points used in the stencil are themselves immersed boundary grid points. If they are, then the interpolation scheme will not be a function strictly of velocities at non-immersed boundary grid points. As a result, using a fixed velocity field at non-immersed boundary grid points, repeated application of the interpolation scheme at all immersed boundary grid points will result in changing values.
at some of the immersed boundary grid points. Which immersed boundary grid points change and how they change will depend on the coupling between immersed boundary grid points introduced by the presence of immersed boundary grid points in the stencils used to interpolate other immersed boundary grid points. In the present work, this issue is avoided by choosing the stencil in which the boundary is closest to the grid point being interpolated and which does not contain any immersed boundary grid points. It is also possible to use reflection interpolation. The method of (Mohd-Yusof, 1997) (used in the present work) is equivalent to replacing the momentum equation (from which the velocity is determined) at each immersed boundary grid point with an equation for the velocity that is a linear combination of velocities at non-immersed boundary grid points.
4.4 Conservation of Mass

A problem with the above interpolation procedure is that the continuity equation is not taken into account. Thus it is possible that the resulting velocity field will not be divergence free. (Kim et al., 2001) present a method by which a mass source term is added to the continuity equation (Eq. 2.2). This mass source term is equal to the negative of the mass source (resulting only from the contribution of immersed boundary points in that cell) at each mass conservation cell. The values at these immersed boundary grid points are obtained by interpolating the pseudo velocities, which are obtained by advancing the momentum equations in time without the pressure gradient. By definition, the sum of this added mass source term over the entire domain is zero and therefore global mass conservation is not affected. However, because the forcing term (which using the method of (Mohd-Yusof, 1997) is equivalent to interpolating the velocity onto the immersed boundary point) should depend on the divergence free velocity at the time step being solved for, and not on the pseudo velocities, the resulting velocity field will not satisfy the desired velocity at the boundary exactly. It may be possible to overcome this problem by solving the pressure-Poisson, momentum and immersed boundary interpolation equations iteratively at each time step. However, this iterative process would be computationally expensive. Another issue is that due to the added mass source term, the velocity obtained from solving the pressure-Poisson equation will not be divergence free. While (Kim et al., 2001) present results for various cases showing improvements due to the addition of the mass source term, it is unclear why this happens and why that particular form for the mass source term is chosen.

For an example of how the interpolation procedure can violate conservation of mass, consider the four velocities, $u_{i-1,j}$, $u_{i,j}$, $v_{i,j-1}$, $v_{i,j}$, surrounding the mass conservation cell at the $i,j$ location in Fig. 4.2. These velocities are not determined from the momentum equations. They are determined by the interpolation scheme, which makes them a linear combination of
velocities at non-immersed boundary points. In general, as the interpolation scheme is defined without regard to the continuity equation, the velocities obtained from the interpolation scheme will not satisfy conservation of mass for the cell shown. This will cause the pressure in these mass conservation cells to slowly increase without bound, if a solution method such as artificial compressibility, SCGS, colored SCGS, SCSG-PPV or SCGS-PP is used, in which the change in pressure is proportional to the divergence. Interestingly, while the pressure will slowly increase without bound, it has been observed by the author that the solution does not diverge. Note that, as a result of the local mass source or sink at a mass conservation cell, there will also be a source or sink of any chemical species present at that mass conservation cell.

4.5 Constrained Interpolation

A method of avoiding the mass conservation problem is to modify the interpolation scheme such that it results in a divergence free velocity field. The idea is to add an amount to each velocity obtained through the unmodified interpolation scheme, while minimizing the amount added, subject to the constraint that the resulting velocity field is divergence free. $Lu_n(u), Lv_n(v)$ and $Lw_n(w)$ are a linear combination of non-immersed boundary points that determine the unmodified interpolation scheme for the $u,v$ and $w$ components of velocity respectively at the $nth$ immersed boundary point of that respective velocity component. Define new velocities $	ilde{U}_n = Lu_n(u) + a_n, \tilde{V}_n = Lv_n(v) + b_n$ and $	ilde{W}_n = Lw_n(w) + c_n$ at each of the $n$ immersed boundary points of that respective velocity component. It is desired that these new velocities satisfy the continuity equation and be as close as possible to the unmodified interpolation schemes $Lu_n(u), Lv_n(v)$ and $Lw_n(w)$. This requires that $a_n, b_n, c_n$ be as small as possible subject to the constraints that the new velocities $	ilde{U}_n, \tilde{V}_n$ and $	ilde{W}_n$ satisfy the continuity equation. This defines a constrained minimization problem which can be cast in the following form (Equations 4.2 and 4.3):
Minimize

\[ H = \sum_{n=1}^{N_u} a_n^2 + \sum_{n=1}^{N_v} b_n^2 + \sum_{n=1}^{N_w} c_n^2 \]  \hspace{1cm} (4.2)

where \( N_u \) is the number of \( u \) immersed boundary points

where \( N_v \) is the number of \( v \) immersed boundary points

where \( N_w \) is the number of \( w \) immersed boundary points

Subject to the constraints that

\[ 0 = \frac{(\tilde{U}_{fu(m)} - \tilde{U}_{bu(m)})}{\Delta x_i(m)} + \frac{(\tilde{V}_{fv(m)} - \tilde{V}_{bv(m)})}{\Delta y_j(m)} + \frac{(\tilde{W}_{fw(m)} - \tilde{W}_{bw(m)})}{\Delta z_k(m)} \]  \hspace{1cm} (4.3)

for \( m = 1, N_c \) where \( N_c \) is the number of constraints

The notation used to describe the position of the variables with respect to each other is shown in Fig. 4.4 and Fig. 4.5. The subscript \( bu \) indicates a location one index behind the subscript \( fu \) in the \( i \) direction with regard to equations centered at mass conservation cells. Similarly the subscripts \( bv \) (\( bw \)) indicate a location one index behind the subscript \( fu \) (\( fw \)) in the \( j \) (\( k \)) direction with regard to equations centered at mass conservation cells. The subscript \( lu \) indicates a location one index behind the subscript \( ru \) in the \( i \) direction with regard to equations centered at \( u \) velocity locations. Similarly, the subscripts \( lv \) (\( lw \)) indicate a location one index behind the subscript \( rv \) (\( rw \)) in the \( j \) (\( k \)) direction with regard to equations centered at \( v \) (\( w \)) velocity locations.

The number of constraints \((N_c)\), is identical to the number of cells for which it is desired to satisfy conservation of mass. Equation 4.3 can be written as Eq. 4.4. Note that each \( a_n, b_n, c_n \) can be in at most two constraint equations. This is a direct result of the staggered grid in which each velocity point appears in at most two mass conservation cells.
Figure 4.4: Notation used for equations centered at mass conservation cells

Figure 4.5: Notation used for equations centered at velocity points
\[ 0 = \frac{(Lu_{fu}(m) + a_{fu}(m) - Lu_{bu}(m) - a_{bu}(m))}{\Delta x_i(m)} + \frac{(Lv_{fv}(m) + b_{fv}(m) - Lv_{bv}(m) - b_{bv}(m))}{\Delta y_j(m)} + \frac{(Lw_{fw}(m) + c_{fw}(m) - Lw_{bw}(m) - c_{bw}(m))}{\Delta z_k(m)} \]

for \( m = 1, N_c \) where \( N_c \) is the number of constraints

The constrained minimization problem given by Equations 4.2 and 4.4 can be converted into an unconstrained minimization problem by forming a modified objective function (Eq. 4.5) by the use of Lagrange multipliers (the \( \lambda_m \)).

\[ G = \sum_{n=1}^{N_u} a_n^2 + \sum_{n=1}^{N_v} b_n^2 + \sum_{n=1}^{N_w} c_n^2 \]

\[ + \sum_{m=1}^{N_c} \lambda_m \left( \frac{a_{fu}(m) - a_{bu}(m)}{\Delta x_i(m)} + \frac{b_{fv}(m) - b_{bv}(m)}{\Delta y_j(m)} + \frac{c_{fw}(m) - c_{bw}(m)}{\Delta z_k(m)} \right) + S_m \]
\[ \frac{\partial G}{\partial (a_1)} = 2a_1 - \frac{\lambda_{ru(1)}}{\Delta x_{ru(1)}} + \frac{\lambda_{lu(1)}}{\Delta x_{lu(1)}} = 0 \] (4.6)

... 

\[ \frac{\partial G}{\partial (a_{Nu})} = 2a_{Nu} - \frac{\lambda_{ru(Nu)}}{\Delta x_{ru(Nu)}} + \frac{\lambda_{lu(Nu)}}{\Delta x_{lu(Nu)}} = 0 \]

\[ \frac{\partial G}{\partial (b_1)} = 2b_1 - \frac{\lambda_{rv(1)}}{\Delta y_{rv(1)}} + \frac{\lambda_{lv(1)}}{\Delta y_{lv(1)}} = 0 \]

... 

\[ \frac{\partial G}{\partial (b_{Nu})} = 2b_{Nu} - \frac{\lambda_{rv(Nu)}}{\Delta y_{rv(Nu)}} + \frac{\lambda_{lv(Nu)}}{\Delta y_{lv(Nu)}} = 0 \]

\[ \frac{\partial G}{\partial (c_1)} = 2c_1 - \frac{\lambda_{rw(1)}}{\Delta z_{rw(1)}} + \frac{\lambda_{lw(1)}}{\Delta z_{lw(1)}} = 0 \]

... 

\[ \frac{\partial G}{\partial (c_{Nu})} = 2c_{Nu} - \frac{\lambda_{rw(Nu)}}{\Delta z_{rw(Nu)}} + \frac{\lambda_{lw(Nu)}}{\Delta z_{lw(Nu)}} = 0 \]

\[ \frac{\partial G}{\partial \lambda_{1}} = \left( \frac{a_{fu(1)} - a_{bu(1)}}{\Delta x_{i(1)}} \right) + \left( \frac{b_{fv(1)} - b_{bv(1)}}{\Delta y_{j(1)}} \right) + \left( \frac{c_{fw(1)} - c_{bw(1)}}{\Delta z_{k(1)}} \right) + S_1 = 0 \]

... 

\[ \frac{\partial G}{\partial \lambda_{Ne}} = \left( \frac{a_{fu(Ne)} - a_{bu(Ne)}}{\Delta x_{i(Ne)}} \right) + \left( \frac{b_{fv(Ne)} - b_{bv(Ne)}}{\Delta y_{j(Ne)}} \right) + \left( \frac{c_{fw(Ne)} - c_{bw(Ne)}}{\Delta z_{k(Ne)}} \right) + S_{Ne} = 0 \]

If Eq. 4.6 is solved, the result will be a velocity field that is divergence free at all mass conservation cells included as constraints. At this point, the choice of which mass conservation cells are made divergence free, i.e. appear as constraints in Eq. 4.6, has not been discussed.

Note that, to guarantee a solution to Eq. 4.6, the number of constraints must be less than or equal to the number of parameters that are being modified i.e. \( N_u + N_v + N_w \geq N_c \). A natural choice of constraints is all mass conservation cells in which all six velocity points are immersed boundary points. However, the author has observed that this choice of constraints does not result in a divergence free velocity field in the entire flow domain for some cases. A reason for this can be seen by considering Fig. 4.6. The only velocity in the \((i+1,j)\) mass conservation cell that is determined from the momentum equation is \( u_{i+1,j} \). As the other velocities in this mass
conservation cell are determined from the immersed boundary method without regard to the mass conservation of this cell (i.e. this cell is not a constraint in Eq. 4.6), they determine the value of $u_{i+1,j}$ needed to satisfy mass conservation for this cell. However, $u_{i+1,j}$ is being determined from the momentum equation written at that point. In three dimensions it has been found necessary to include as constraints all mass conservation cells in which three or more of the six velocity points in the cell are immersed boundary points.

4.6 Solving the Optimization Problem

One method of solving the resulting system of equations is a direct method. This would involve the inversion of a matrix with dimension $(N_u + N_v + N_w + N_c) \times (N_u + N_v + N_w + N_c)$. The computation of each $a_n, b_n, c_n$ would involve the dot product of a $(N_u + N_v + N_w + N_c)$ length row of the inverse of the matrix with the right hand side vector. As the first $(N_u + N_v + N_w)$ components of the right hand side vector are zero, the amount of work involved for the
computation of each \( a_n, b_n, c_n \) can be shortened to a dot product of size \( N_c \). Note that the matrix is a function only of the geometry and not the current velocity field. Therefore, if the geometry does not change, the matrix would only need to be inverted once. Taking into account the computational savings described, the computational work involved in computing \( a_n, b_n, c_n \) scales as \( (N_u + N_v + N_w) N_c^2 \). This scaling is unacceptable. An additional problem arises when considering parallel computing as the parallelization of direct methods is difficult.

### 4.7 SCGS-PO

Consider an initial guess \( u^* \) to the solution of a general linear system \( Au = b \). The residual \( R \) resulting from the initial guess \( u^* \) can be written as:

\[
R = Au^* - b = A(u - a) - b = Au - b - Aa \rightarrow Aa = -R \quad \text{as} \quad Au - b = 0
\]

where \( u \) is the exact solution and \( a + u^* = u \)

This form of the matrix equation which solves for the change in the variables, instead of the actual variables, is termed the residual form. An efficient iterative method will replace the difficult-to-invert matrix \( A \) with one that is easily inverted. A trivial-to-invert matrix is one formed from the diagonal elements of \( A \). This results in Jacobi iteration. However, if the matrix arising from Eq. 4.6 is examined it can be seen that the diagonal element is zero for the rows corresponding to the constraint equations. As a result, Jacobi iteration cannot be used. An iterative method that is suitable for a matrix equation with this structure can be obtained by solving a subset of equations with a certain structure. Consider Fig. 4.7 (in which for clarity only two dimensions are shown). For each constraint equation in Fig. 4.7, a matrix (Eq. 4.7) can be written. The result, in three dimensions, is a sparse \( 7 \times 7 \) matrix for which an analytical solution can be easily obtained. The solution for \( \Delta \lambda \) obtained from Eq. 4.7 is given by Eq. 4.8. The solution for the changes in velocity are given by Eq. 4.9.
\[
\begin{bmatrix}
2 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{\Delta x} \\
0 & 2 & 0 & 0 & 0 & 0 & -\frac{1}{\Delta y} \\
0 & 0 & 2 & 0 & 0 & 0 & -\frac{1}{\Delta z} \\
0 & 0 & 0 & 2 & 0 & 0 & -\frac{1}{\Delta z} \\
0 & 0 & 0 & 0 & 2 & 0 & 0 \\
-\frac{1}{\Delta x} & \frac{1}{\Delta x} & \frac{1}{\Delta y} & \frac{1}{\Delta y} & \frac{1}{\Delta z} & \frac{1}{\Delta z} & 0
\end{bmatrix}
\begin{bmatrix}
\Delta b \\
\Delta f \\
\Delta b \\
\Delta f \\
\Delta b \\
\Delta \lambda
\end{bmatrix} =
\begin{bmatrix}
-U_b \\
-U_f \\
-V_b \\
-V_f \\
-W_b \\
-W_f
\end{bmatrix}
\] (4.7)

\[
\Delta \lambda = \left(\frac{2}{\Delta x^2} + \frac{2}{\Delta y^2} + \frac{2}{\Delta z^2}\right)^{-1}\left(2\Lambda + \frac{U_b}{\Delta x} - \frac{U_f}{\Delta x} + \frac{V_b}{\Delta y} - \frac{V_f}{\Delta y} + \frac{W_b}{\Delta z} - \frac{W_f}{\Delta z}\right)
\] (4.8)

\[
a_b = \frac{1}{2}\left(-U_b + \frac{\Delta \lambda}{\Delta x}\right)
\] (4.9)

\[
a_f = \frac{1}{2}\left(-U_f - \frac{\Delta \lambda}{\Delta x}\right)
\]

\[
b_b = \frac{1}{2}\left(-V_b + \frac{\Delta \lambda}{\Delta y}\right)
\]

\[
b_f = \frac{1}{2}\left(-V_f - \frac{\Delta \lambda}{\Delta y}\right)
\]

\[
c_b = \frac{1}{2}\left(-W_b + \frac{\Delta \lambda}{\Delta z}\right)
\]

\[
c_f = \frac{1}{2}\left(-W_f - \frac{\Delta \lambda}{\Delta z}\right)
\]

The structure of Eq. 4.7 is identical to that used in the SCGS method (Vanka, 1986) which defines a way to couple the solutions of the pressure and velocity in the Navier-Stokes equations. As Eq. 4.7 is written for each constraint equation which describes a specific mass conservation cell, and a velocity point can lie in two mass conservation cells, the possibility exists of obtaining two different solutions at each interior velocity point. Considering Fig. 4.7 two (in general different) solutions to \(a_{i,j}\) will be obtained from Eq. 4.7 when it is solved at the \(i, j\) and \(i + 1, j\) locations. A similar situation occurs in the SCGS method (Vanka, 1986). When using the SCGS method while attempting to obtain solutions to the Navier-Stokes equations (Vanka, 1986), all attempts by the author to combine these two values have been highly unstable. A similar effort to combine the two solutions obtained from Eq. 4.7 has not been attempted by the author. The issue of two solutions at each interior velocity point can be avoided by using the latest computed value for each interior velocity point when computing the residuals. However, this means that Eq. 4.7
at the \( i + 1, j \) point cannot be solved until Eq. 4.7 at the \( i, j \) point has been solved. Consequently, the algorithm is serial and not appropriate for use on parallel computers. However, analogous to the SCGS-PP algorithm, only the solution for \( \Delta \lambda \) from Eq. 4.7 can be used while the solutions for the changes in velocity can be obtained from Eq. 4.6. The result is a parallel algorithm, termed Symmetric Coupled Gauss Seidel Parallel Optimization (SCGS-PO).

4.7.1 Grid Coloring

It is possible in certain cases to split the domain so that a solver analogous to SCGS can be used to provide a unique solution for the velocity and pressure in parallel. As the structure of Eq. 4.7 is identical to that used in the SCGS method (Vanka, 1986), this grid splitting or coloring avoids the possibility of obtaining two solutions, without causing the algorithm to become serial. Fig. 4.8 shows how this splitting can be done in two dimensions. In Fig. 4.8, Eq. 4.7 can be solved at each blue point independently of any red points. This yields a solution for all immersed boundary points that belong to two constraint equations and all the Lagrange multipliers in the blue grid. If Eq. 4.7 is then solved for all red points, the remaining Lagrange multipliers will be solved for, along with all immersed boundary points that belong to two constraint equations. Immersed boundary points belonging to only one constraint equation will be solved for when Eq. 4.7 is solved for on the colored grid containing that constraint equation. This grid coloring can be extended to three dimensions while still using two colors.
4.7.2 Flow Over a Sphere

Flow over a sphere is solved using the regular interpolation and the optimized divergence free interpolation. The Reynolds number based on the sphere diameter and the inflow velocity is 100. Periodic boundaries are used in the \( y \) and \( z \) directions, while at the outlet a convective boundary condition (Eq. 3.20) is used for all three velocity components. The computational domain in the \( x \) direction extends \( 5d \) upstream and \( 20d \) downstream of the sphere. The boundaries in the \( y \) and \( z \) directions extend \( 7d \) on each side of the sphere. At this Reynolds number the flow is steady. A stretched grid is used that decreases the grid spacing in all three dimensions in the region of the sphere. The computed drag coefficient is 1.16 using the regular interpolation and 1.34 using the optimized divergence free interpolation. The value found in (Kim et al., 2001) was 1.087.
Fig. 4.9 shows a comparison of the velocity vectors along with contours of the $x$ component of velocity. The planes shown in Fig. 4.9 and Fig. 4.10 are in the $xy$ plane at $z = 0$. The flow is axi-symmetric in the $yz$ plane. It can be seen that the divergence free interpolation results in a vastly different flow field inside the sphere. Figure 4.10 shows contours of pressure. It can be seen that for the case of the regular interpolation the pressure increases without bound. While it might be expected that this would cause the solution to diverge it does not. It should be noted that in the case of the divergence free interpolation the pressure in all mass conservation cells that are constraints does not change from its initial value if the divergence free condition is enforced exactly. In this case, the pressure at these mass conservation cells that are constraints cannot be considered physical.
4.8 Optimizing Entire Flow Field

4.8.1 Fractional Step

In the fractional step approach (Chorin, 1968) the momentum equations are split into two parts (Equations 4.10 and 4.11) the sum of which equals the momentum equations. In Eq. 4.10, \( C^n_i \) and \( D^n_i \) represent explicit time integration schemes applied to the spatially discretized convection and diffusion terms respectively.

\[
\frac{\tilde{u}_i - u^n_i}{\Delta t} = C^n_i + D^n_i \\
\frac{u^{n+1}_i - \tilde{u}_i}{\Delta t} = -\frac{\partial p^{n+1}}{\partial x_i} \\
\frac{\partial}{\partial x_i} \left( \frac{u^{n+1}_i - \tilde{u}_i}{\Delta t} \right) = \frac{\partial}{\partial x_i} \left( -\frac{\partial p^{n+1}}{\partial x_i} \right) \quad \text{as} \quad \frac{\partial u^{n+1}_i}{\partial x_i} = 0 \rightarrow \frac{\partial \tilde{u}_i}{\partial x_i} = \frac{\partial^2 p^{n+1}}{\partial x_i \partial x_i} \quad (4.12)
\]
Equation 4.10 is advanced in time to obtain a pseudo velocity field \( \tilde{u}_i \). This pseudo velocity field generally will not satisfy the divergence free condition. An equation for a divergence free velocity field can be obtained by taking the divergence of Eq. 4.11 and enforcing the divergence of the velocity at the \( n + 1 \) time step to be zero, resulting in Eq. 4.12. Once the pressure field is obtained from the solution of Eq. 4.12, the velocity field at the \( n + 1 \) time step can be obtained from Eq. 4.11.

### 4.8.2 Lagrange Multiplier Approach

Consider Eq. 4.13 in which \( \tilde{u}_i \) has been obtained from Eq. 4.10. \( \Delta u_i \) is the change in \( \tilde{u}_i \).

\[
\Delta u_i^{n+1} = \tilde{u}_i + \Delta u_i \tag{4.13}
\]

A problem can now be defined in which it is desired that \( u_i^{n+1} \) be as close as possible to \( \tilde{u}_i \) subject to the constraint that \( u_i^{n+1} \) be divergence free. This means that \( \Delta u_i \) be minimized in some sense subject to the constraint that \( \tilde{u}_i + \Delta u_i \) is divergence free. Forming a modified objective function using Lagrange multipliers results in Eq. 4.14.

\[
G = \sum_{i} \Delta u_{i,j,k}^2 + \sum_{i} \Delta v_{i,j,k}^2 + \sum_{i} \Delta w_{i,j,k}^2 + \sum_{i} \lambda_{i,j,k} \left( \frac{\Delta u_{i,j,k} - \Delta u_{i-1,j,k}}{\Delta x u_i} + \frac{\Delta v_{i,j,k} - \Delta v_{i,j-1,k}}{\Delta y v_j} + \frac{\Delta w_{i,j,k} - \Delta w_{i,j,k-1}}{\Delta z w_k} + S_m \right)
\]

where \( S_m = \frac{(\tilde{u}_{i,j,k} - \tilde{u}_{i-1,j,k})}{\Delta x u_i} + \frac{(\tilde{v}_{i,j,k} - \tilde{v}_{i,j-1,k})}{\Delta y v_j} + \frac{(\tilde{w}_{i,j,k} - \tilde{w}_{i,j,k-1})}{\Delta z w_k} \)

and \( \Delta x u_i \) is the distance in the \( x \) direction between velocity points at the \( i \)th main grid point.

and \( N_u, N_v, N_w, N_p \) is the number of interior \( u, v, w, p \) velocity points respectively

Setting the partial derivatives of Eq. 4.14 to zero results in a system of equations (Eq. 4.15).

Note that all velocity points lie in exactly two mass conservation cells. The structure of Eq. 4.15 is identical to that used in SCGS (Vanka, 1986), which defines a way to couple the solutions of the pressure and velocity in the Navier-Stokes equations. As a result, it can be solved and parallelized by the use of algorithms analogous to colored SCGS and SCGS-PP.
\[
\frac{\partial G}{\partial (\Delta u_{i,j,k})} = 2\Delta u_{i,j,k} - \frac{\lambda_{i+1,j,k}^{n+1}}{\Delta x u_{i+1}} + \frac{\lambda_{i,j,k}^{n+1}}{\Delta x u_{i}} = 0
\] (4.15)

\[
\frac{\partial G}{\partial (\Delta v_{i,j,k})} = 2\Delta v_{i,j,k} - \frac{\lambda_{i+1,j,k}^{n+1}}{\Delta y v_{j+1}} + \frac{\lambda_{i,j,k}^{n+1}}{\Delta y v_{j}} = 0
\]

\[
\frac{\partial G}{\partial (\Delta w_{i,j,k})} = 2\Delta w_{i,j,k} - \frac{\lambda_{i,j,k+1}^{n+1}}{\Delta z w_{k+1}} + \frac{\lambda_{i,j,k}^{n+1}}{\Delta z w_{k}} = 0
\]

\[
\frac{\partial G}{\partial (\lambda_{i,j,k}^{n+1})} = \frac{(\Delta u_{i,j,k} - \Delta u_{i-1,j,k})}{\Delta x u_{i}} + \frac{(\Delta v_{i,j,k} - \Delta v_{i,j-1,k})}{\Delta y v_{j}} + \frac{(\Delta w_{i,j,k} - \Delta w_{i,j,k-1})}{\Delta z w_{k}} + S_m
\]

Obtaining the solution of Eq. 4.15 and rearranging results in:

\[
\frac{\Delta u_{i,j,k}}{\Delta t} = \frac{\lambda_{i+1,j,k}^{n+1}}{2\Delta t \Delta x u_{i+1}} + \frac{\lambda_{i,j,k}^{n+1}}{2\Delta t \Delta x u_{i}} = 0
\]

Adding to the \(x\) component of Eq. 4.10 results in:

\[
\frac{\Delta u_{i,j,k}}{\Delta t} - \frac{\lambda_{i+1,j,k}^{n+1}}{2\Delta t \Delta x u_{i+1}} + \frac{\lambda_{i,j,k}^{n+1}}{2\Delta t \Delta x u_{i}} + \frac{\bar{u}_{i,j,k} - \Delta u_{i,j,k}^{n}}{\Delta t} = C_1^n + D_1^{n+1}
\]

simplifying

\[
\frac{\bar{u}_{i,j,k} + \Delta u_{i,j,k} - u_{i,j,k}^{n}}{\Delta t} = \frac{u_{i,j,k}^{n+1} - u_{i,j,k}^{n}}{\Delta t} = C_1^n + D_1^{n+1} + \frac{\lambda_{i+1,j,k}^{n+1}}{2\Delta t \Delta x u_{i+1}} - \frac{\lambda_{i,j,k}^{n+1}}{2\Delta t \Delta x u_{i}}
\] (4.16)

Using a two-point stencil for the pressure gradient, the \(x\) momentum equation is Eq. 4.17.

\[
\frac{u_{i,j,k}^{n+1} - u_{i,j,k}^{n}}{\Delta t} = C_1^n + D_1^{n+1} - \left( \frac{p_{i+1,j,k}^{n+1} - p_{i,j,k}^{n+1}}{\Delta x_i} \right) \Delta x_i
\] (4.17)

\[
\frac{\lambda_{i+1,j,k}^{n+1}}{2\Delta t \Delta x u_{i+1}} - \frac{\lambda_{i,j,k}^{n+1}}{2\Delta t \Delta x u_{i}} = - \left( \frac{p_{i+1,j,k}^{n+1} - p_{i,j,k}^{n+1}}{\Delta x_i} \right) \Delta x_i
\] (4.18)

If Eq. 4.16 is subtracted from Eq. 4.17 (resulting in Eq. 4.18), it can be seen that \(-\frac{\lambda}{2\Delta t} = p\), if \(\Delta x u_i = \Delta x u_{i+1} = \Delta x_i\). This will occur only if the grid spacing is constant in each coordinate direction. If so, then pressure is seen to be linearly related to the Lagrange multiplier.

4.8.3 Lagrange Multiplier Approach with Incorporated Immersed Boundaries

Using the method of (Mohd-Yusof, 1997), at each immersed boundary grid point, Eq. 4.10 is replaced with an interpolation equation for the pseudo velocity that is a linear combination (denoted by the operator \(L\)) of pseudo velocities at non-immersed boundary points. The solution of Eq. 4.15 proceeds as before.
\[
\frac{\widetilde{u}_i - u_i^n}{\Delta t} = C_i^n + D_i^n \text{ for non-immersed boundary points}
\]

\[
\widetilde{u}_i = L(\widetilde{u}_i \text{ non-immersed boundary points}) = L([C_i^n + D_i^n] \Delta t + u_i^n) \text{ for immersed boundary points}
\]

When the solution of Eq. 4.15 for the \(x\) component of change in velocity is added to the interpolation scheme for the \(x\) component of the pseudo velocity the result is Eq. 4.19. This equation may make physical sense because the resulting expression for the velocity at immersed boundary points is of the same form as Eq. 4.16 and (if the grid is even) Eq. 4.17, with \([C_i^n + D_i^n] \Delta t + u_i^n\) at the immersed boundary point replaced with a linear combination of \([C_i^n + D_i^n] \Delta t + u_i^n\) at non-immersed boundary points. If the grid is not even, then the relationship \(-\lambda \frac{\Delta t}{2\Delta x} = p\) does not hold. As a result, the absolute value of \(\lambda\) is important. This has been verified numerically by the author.

\[
\widetilde{u}_{i,j,k} + \Delta u_{i,j,k} = u_{i,j,k}^{n+1} = L([C_i^n + D_i^n] \Delta t + u_i^n) + \frac{\lambda_{i+1,j,k}^{n+1}}{2\Delta x u_{i+1}} - \frac{\lambda_{i,j,k}^{n+1}}{2\Delta x u_i}
\] (4.19)

### 4.9 Defining the Forcing Using Pseudo Velocities

\[
\frac{\widetilde{u}_i - u_i^n}{\Delta t} = C_i^n + D_i^n + F_i
\] (4.20)

The forcing can be determined by considering Eq. 4.20. This results in a forcing that depends on the pseudo velocity and therefore depends only on the previous time step velocity field and not on the pressure gradient at the \(n + 1\) time level. As the immersed boundary generally does not coincide with the grid points an interpolation operator \(L\) is used to define the pseudo velocity at immersed boundary points resulting in Eq. 4.21.

\[
F_i(\widetilde{u}_i) = \frac{L(\widetilde{u}_i) - u_i^n}{\Delta t} - C_i^n - D_i^n
\] (4.21)

Note that as the interpolation operator \(L\) may contain immersed boundary points (i.e. points at which the forcing \(F_i\) is added) this implies a coupled system of equations in order to find \(F_i\) from Eq. 4.20. In the present work this is avoided by determining the pseudo velocities at the grid
points involved in the interpolation operator $L$ without considering $F_i$ and then using these pseudo velocities to find $F_i$. If the interpolation operator $L$ does not contain any immersed boundary points (for the geometries considered in the present work it contains very few) then this issue does not arise. The momentum equations solved are given by Eq. 4.22.

$$\frac{u_{i}^{n+1} - u_{i}^{n}}{\Delta t} = C_{i}^{n} + D_{i}^{n} - \frac{\partial p^{n+1}}{\partial x_{i}} + F_{i}(\bar{u}_{i})$$  \hspace{1cm} (4.22)

$$\frac{u_{i}^{n+1} - u_{i}^{n}}{\Delta t} = C_{i}^{n} + D_{i}^{n} - \frac{\partial p^{n+1}}{\partial x_{i}} + \left(\frac{L(\bar{u}_{i}) - u_{i}^{n}}{\Delta t} - C_{i}^{n} - D_{i}^{n}\right) = \frac{L(\bar{u}_{i}) - u_{i}^{n}}{\Delta t} - \frac{\partial p^{n+1}}{\partial x_{i}}$$

$$u_{i}^{n+1} = L(\bar{u}_{i}) - \Delta t \frac{\partial p^{n+1}}{\partial x_{i}}$$  \hspace{1cm} (4.23)

The solution to Eq. 4.23 along with the continuity equation will guarantee a divergence free field. The addition of the forcing term has only resulted in a change in the source term (i.e. the term not at the $n + 1$ time level) at each immersed boundary point. The problem with this approach is that the forcing is determined such that the pseudo velocity meets the desired velocity of the boundary. This approach is not correct because the forcing should be chosen such that the velocity meets the desired velocity of the boundary. The difference between the pseudo velocity and the velocity is $\Delta t \frac{\partial p^{n+1}}{\partial x_{i}}$. This presence of the pressure gradient is what allows the velocity to satisfy conservation of mass. However it prevents the pressure gradient at the $n + 1$ time level from having any effect on the forcing term. As the error is proportional to $\Delta t$, the issue arises as to how the time step affects the solution, including the undesirable dependence of a steady state solution on the time step.

### 4.9.1 Flow Over a Cylinder

The Lagrange multiplier approach with immersed boundaries has been applied to the case of flow over a cylinder. The Reynolds number is 40. Periodic boundaries are used in the $y$ and $z$ directions. The computational domain in the $x$ direction extends $5d$ upstream and $20d$ downstream of the cylinder. The boundaries in the $y$ direction extend $7d$ above and below the
Figure 4.11: Cylinder in crossflow, $Re = 40$ (even grid)

cylinder. At this Reynolds number the flow is two-dimensional and therefore the $z$ direction is not resolved. The grid spacing is even although it is different in each of the three dimensions. The computed drag coefficient is 1.75. The value found in (Kim et al., 2001) was 1.51. Figure 4.11 shows contours of the Lagrange multiplier field. It can be seen that the Lagrange multiplier field has the opposite sign of the expected pressure field.

4.10 Conclusion

Two new methods that ensure a divergence free velocity field in conjunction with the immersed boundary method have been developed. The governing equations are the incompressible Navier-Stokes equations. These equations are discretized on a staggered grid. The new methods are based on constrained optimization. One of these methods is particularly suited for use with
the artificial compressibility, SCGS and SCGS-PP methods. The other method is restricted to evenly spaced grids. Both methods can be parallelized. These methods have been demonstrated on two flows, demonstrating the ability to drive the residual of the continuity equation to zero.
Tetra is a computer program written by the author that solves the non-conservative form of the unsteady incompressible three-dimensional Navier-Stokes equations (Eq. 2.1 and Eq. 2.2). It can also solve equations describing the transport of any number of passive scalars (Eq. 2.3). Reacting flows have not been solved with Tetra, but could be if the appropriate source terms describing the chemical reactions were added to the scalar equations. Artificial compressibility, SCGS-PP, SCGS-PPV and colored SCGS have been implemented as solvers for the Navier-Stokes equations (Eq. 2.1 and Eq. 2.2). Finite difference schemes of up to sixth-order have been implemented for the various terms in Equations 2.1, 2.2 and 2.3. These equations can be integrated (i.e. discretized) in time using a large number of implicit or explicit schemes. It is possible to use an implicit scheme for the diffusive terms and an explicit scheme for the convective terms, and vice versa. The program is written in Fortran 95 and uses many of the new capabilities of Fortran 95. The computational grid used is Cartesian. The grid can be stretched in all three dimensions. There are four types of boundary conditions, periodic, no slip walls, imposed flux and outflow. Complicated and/or moving geometries can be solved using the immersed boundary method. Passive massless particles that are transported by the flow have been implemented in a parallel algorithm. These particles can be added to the flow and tracked in time for purposes such as determining residence times and visualization. All the results presented in the present work are obtained from Tetra.

5.1 Parallelization

5.1.1 HPF

The first code written in the course of the present work was designed to run on Cray vector machines such as the C90 and T90. This code was written so that the computationally expensive
loops could be vectorized by the compiler. Parallelization was accomplished by means of Cray’s auto-tasking compiler. This auto-tasking is a means of distributing Fortran do-loops, in which the computational work can be executed in parallel, across multiple processors. This approach worked well, primarily because of the shared memory architecture and its associated low latency for memory access. As of this writing there are few vector machines in use. By far the most common supercomputing architectures are distributed memory machines. These machines are not vector machines and due to the distributed memory architecture, do not have global (shared) memory access. When it was realized that it was necessary to develop a program for distributed memory architectures an attempt was made to use High Performance Fortran (HPF). HPF is an addition to the Fortran standard that is intended to parallelize existing serial programs with a minimum of user input. Of course, the algorithm must be capable of being parallelized. A nearest-neighbor algorithm as used by Tetra is a good candidate for parallelization. Essentially the only changes to Tetra should be to insert directives to give the HPF compiler some idea of how it should distribute the arrays over the processors. The communication required by the distribution of the arrays should then be done automatically by the HPF compiler. The first HPF compiler tested was the PGHPF compiler (by the Portland Group) on the Cray T3E. An error was uncovered in the compiler which resulted in wrong floating point numbers if optimization (which was necessary for acceptable performance) was turned on. This problem was corrected by the Portland Group, but other problems such as extremely long compile times remained. The compile time was longest in subroutines that had complicated (high order accurate) finite difference stencils. The next HPF compiler tested was from the Digital Equipment Corporation (DEC). This compiler was tested on DEC Alpha machines, running the Tru64 unix operating system. The compiler ran out of memory (on a machine containing four $10^9$ bytes of RAM) after 15–20 minutes attempting to compile one subroutine of Tetra. The subroutines that the compiler
was unable to compile contained the high order accurate finite difference stencils. The last HPF compiler tested was ADAPTOR. It was unable to compile the dynamic memory allocation parts of Tetra. Simple test problems, such as solving the heat equation in two dimensions using Jacobi iteration and simple second-order accurate finite difference stencils, performed well using the DEC and ADAPTOR compilers. The underlying problem with all the HPF compilers tested appears to be their inability to analyze the communication patterns resulting from high order accurate finite difference stencils. The conclusion is that current HPF compilers are unable to parallelize algorithms that contain complicated high order accurate finite difference stencils, such as used in the present work.

5.1.2 MPI

The MPI (Message Passing Interface) standard is a description of a set of library routines that enable processes to communicate, pass data to each other and perform certain collective operations. The MPI standard has been implemented by a number of companies and organizations. MPI (unlike HPF) is not a compiler that analyzes a program and makes decisions how data is distributed and communicated. While MPI provides a means to enable processes to communicate and exchange data, it does not specify what data is transferred or where it is sent. In MPI the user has explicit control over every message (or piece of data) sent or received by each process. This makes for very complicated but potentially efficient programs. In particular, distributed arrays must be dimensioned to their correct local size explicitly. This changes the programming structure from that of managing one process to that of managing many processes and their communications with each other. It should be noted that this change in the programming paradigm greatly increases the difficulty in uncovering faults in the program. This difficulty is the primary reason why software able to take advantage of parallel machines is expensive to create. The parallelization of Tetra is accomplished by a domain decomposition coupled with overlap points for variables that must be communicated (Fig. 5.1). The domain decomposition can be
obtained by dividing up one Cartesian grid or it can be defined as a collection of Cartesian blocks. In the latter case the block structure is obtained from a grid generation package called GridPro.

5.2 Performance

Tetra has been run successfully on the following machines in Table 5.1. The vast majority of the results obtained from using Tetra in the present work came from simulations run on Helix, SuperMike and the IBM SP2, IBM SP3 and IBM SP4. The other machines were used primarily for development and testing. Tetra has been run on up to 512 processors (on a Cray T3E), but is commonly run on \( \leq 256 \) processors. Figure 5.2 shows the performance of Tetra, using a fixed...
grid size, as the number of processors increases on the machines Helix and Marcellus. The speedup of Tetra is seen to be much worse on Marcellus than on Helix. This is a result of the faster data communication network of Helix coupled with the greater amount of communication vs computation that must be done as the number of processors increases. Nonetheless, the wall clock times of Tetra on the two machines are very close (Fig. 5.2b). Table 5.2 shows the time taken to solve 728 time steps for the case of flow over a backstep on a grid of approximately one million points. The numbers represent speedup relative to the machine "Casper". The performance of any parallel code using a fixed grid size will degrade as the number of processors increases. This degradation results from the increasing ratio of time devoted to communication over that devoted to computation. This effect can be minimized by keeping the amount of computational work per process the same, when increasing the number of processes. This is accomplished in Fig. 5.3 and Fig. 5.4 by doubling the number of finite difference grid points as the number of processors is doubled. Two sets of grids are used (see Table 5.3) to keep the number of finite difference points per processor the same (either $32^3$ or $64^3$). In Figures 5.3 and 5.4, 1000 time steps are solved of a channel flow, using three subiterations of the colored SCGS algorithm. Fig. 5.3 shows the time to solve this problem on the machines Marcellus and Helix (see Table 5.1). Perfect parallel performance would result in a constant value of one in Fig. 5.3. It can be seen that the parallel performance is much better on Helix than Marcellus when 64 processors are used, particularly on the smaller ($32 \times 32 \times 32$) base grid. A sharp decrease in the parallel performance on Marcellus is seen on the smaller ($32 \times 32 \times 32$) base grid after eight processors. This is likely the result of the architecture of Marcellus which is a collection of nodes, each of which consists of eight processors sharing a global memory space. The connection between nodes is such that each processor has a 90Mb/s data connection to a processor in a different node. If a job uses less than nine processors then all processors are in the same node. This allows processors to pass
Table 5.1: Machines to which Tetra has been ported

<table>
<thead>
<tr>
<th>Machine Type</th>
<th>Processor Description</th>
<th>Network Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compaq DEC Alpha 21264</td>
<td>600Mhz 4 floating point operations/cycle</td>
<td>100Mb/s</td>
</tr>
<tr>
<td>SGI 3000 (ruby at ERDC)</td>
<td>400Mhz</td>
<td>Gigabit Ethernet</td>
</tr>
<tr>
<td>Cray SV1 (Zeus)</td>
<td>500Mhz</td>
<td>N/A (Shared memory)</td>
</tr>
<tr>
<td>Cray T3E</td>
<td>450Mhz 4 floating point operations/cycle</td>
<td>480Mb/s</td>
</tr>
<tr>
<td>IBM SP2</td>
<td>195Mhz 4 floating point operations/cycle</td>
<td>60Mb/s</td>
</tr>
<tr>
<td>IBM SP3</td>
<td>375Mhz 4 floating point operations/cycle</td>
<td>37.5Mb/s</td>
</tr>
<tr>
<td>IBM SP4 (Marcellus at NAVO)</td>
<td>1.3Ghz 4 floating point operations/cycle</td>
<td>90Mb/s</td>
</tr>
<tr>
<td>IBM SP4 (Iceflyer at ARSC)</td>
<td>1.7Ghz 4 floating point operations/cycle</td>
<td>Shared memory</td>
</tr>
<tr>
<td>AMD Thunderbird x86</td>
<td>1.1Ghz 2 floating point operations/cycle</td>
<td>100Mb/s</td>
</tr>
<tr>
<td>Pentium III x86</td>
<td>933Mhz 1 floating point operations/cycle</td>
<td>Myrinet (489 Mb/s)</td>
</tr>
<tr>
<td>Xeon (SuperMike)</td>
<td>1.8Ghz 1 floating point operations/cycle</td>
<td>Myrinet (489 Mb/s)</td>
</tr>
<tr>
<td>Xeon (Helix)</td>
<td>2Ghz 1 floating point operations/cycle</td>
<td>Myrinet (489 Mb/s)</td>
</tr>
</tbody>
</table>

data using a shared memory version of MPI, which is much faster than communications over
the network. Note that as the nodes use shared memory, if less than eight processors are used,
the memory access time will decrease as the number of processors used per nodes decreases.
This is a result of a decrease in memory contention between processors which are sharing the
same access path to the global (shared) memory. Both of these factors illustrate the difficulty in
measuring parallel performance. Helix consists of a collection of nodes, each of which consists
of two processors sharing a global memory space. In this case a dramatic increase in performance
can be seen when one processor is used compared to two. This is more pronounced on the larger
(64 × 64 × 64) base grid. This may be a result of the greater potential for memory contention
on the larger grid. The wall clock time required to solve this problem can be seen in Fig. 5.4.
Helix is slower than Marcellus except for the smaller (32 × 32 × 32) base grid using 32 and 64
processors. This is not unexpected as the processor in Marcellus is capable of four floating point
operations per clock cycle whereas the processor in Helix is capable of one. The decreased time
for Helix on the smaller (32 × 32 × 32) base grid using 32 and 64 processors results from Helix’s
faster communication. This faster communication becomes more important on the smaller grid,
which has an increasing ratio of time devoted to communication over that devoted to computation,
compared to the 64 × 64 × 64 base grid.
Figure 5.2: Performance of Tetra on a $128 \times 128 \times 128$ grid, (a) Speedup, (b) wall clock time

Table 5.2: Performance of Tetra on different machines

<table>
<thead>
<tr>
<th>Machine Description</th>
<th>Wall Clock Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM SP2 Power3, 195Mhz at LSU (Casper)</td>
<td>1</td>
</tr>
<tr>
<td>Dual Xeons 1.8Ghz at LSU (Mike)</td>
<td>2.3447</td>
</tr>
<tr>
<td>IBM SP4, 1.3Ghz at NAVO/MSRC used 4 out 8 processors/node</td>
<td>4.6873</td>
</tr>
</tbody>
</table>

5.3 I/O

A bottleneck in the scalability of software on parallel machines is that of I/O. Visualization of the data from some of the simulations in the present work requires the use of many (~700) time sequences of data. The scalability issue can be avoided if each process writes to a local disk (assuming one exists). However, if 256 processors are used for running Tetra, as is common for the largest simulations, then the resulting number of files is 179200. This is too large a number of files to be efficiently used by post-processing software or for an operating system to efficiently manage. In addition, there is the practical problem of having to move the files from the local disks to a single disk for post-processing. These problems can be eliminated by creating one data

Table 5.3: Grids used in performance test of Tetra to keep the number of finite difference points the same for each process

<table>
<thead>
<tr>
<th># processors</th>
<th>first base grid</th>
<th>second base grid</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>32 $\times$ 32 $\times$ 32</td>
<td>64 $\times$ 64 $\times$ 64</td>
</tr>
<tr>
<td>2</td>
<td>64 $\times$ 32 $\times$ 32</td>
<td>128 $\times$ 64 $\times$ 64</td>
</tr>
<tr>
<td>4</td>
<td>64 $\times$ 64 $\times$ 32</td>
<td>128 $\times$ 128 $\times$ 64</td>
</tr>
<tr>
<td>8</td>
<td>64 $\times$ 64 $\times$ 64</td>
<td>128 $\times$ 128 $\times$ 128</td>
</tr>
<tr>
<td>16</td>
<td>128 $\times$ 64 $\times$ 64</td>
<td>256 $\times$ 128 $\times$ 128</td>
</tr>
<tr>
<td>32</td>
<td>128 $\times$ 128 $\times$ 64</td>
<td>256 $\times$ 256 $\times$ 128</td>
</tr>
<tr>
<td>64</td>
<td>128 $\times$ 128 $\times$ 128</td>
<td>256 $\times$ 256 $\times$ 256</td>
</tr>
</tbody>
</table>
Figure 5.3: Normalized time of Tetra on Marcellus and Helix, (a) $32 \times 32 \times 32$ base grid, (b) $64 \times 64 \times 64$ base grid

Figure 5.4: Wall clock time of Tetra on Marcellus and Helix, (a) $32 \times 32 \times 32$ base grid, (b) $64 \times 64 \times 64$ base grid

file at each time step. This will require that all processes write to the same file. In this case the writing process will have to proceed sequentially. This approach has been implemented in Tetra for outputting data files for post-processing.

5.4 Lagrangian Particles

Understanding the physics of a dynamically evolving flow requires methods that can visualize the structures that exist in the flow. One method used in the present work is to collect time sequences of isosurfaces of various quantities (such as pressure, vorticity, gradients of the scalars etc.). This method is very useful, but does have one disadvantage in that it is difficult to tell
the direction and origin of flow. Experimentalists frequently use smoke traces to visualize flow structures and to observe the flow patterns and directions. The smoke consists of particles that are small with respect to the smallest flow structures and are essentially neutrally buoyant with insignificant inertia. As a result they move at essentially the fluid velocity and have little effect on the flow. This can be simulated numerically by introducing massless passive particles that move at the fluid velocity into the flow. These particles have no effect on the flow and their motion is dependent only on the flow. Unlike experiments, particles can be introduced at arbitrary locations without the detrimental effect of the presence of physical seeding apparatus influencing the flow. These particles can be tagged by the locations at which they are introduced into the flow. This is very useful in attempting to understand the origin and development of flow structures. Additional insight can be obtained by marking or coloring the particles by their residence time, distance traveled or acceleration. It is also possible to interpolate quantities associated with the flow field such as pressure, vorticity, scalar field, etc. onto the particles.

5.4.1 Parallelization

The parallelization of the particles presents a problem. In order to load balance the computations associated with each particle, the particles should be distributed evenly around the processes that are used by the flow solver to solve Equations 2.1, 2.2 and 2.3 on an Eulerian grid. However, as the particles will move wherever the flow determines, all the particles could collect in a physical region that is being solved by a specific process. The time integration of each particle requires the values of the neighboring velocity points. Depending on flow patterns, this could mean that all needed velocities have to be communicated from the process in which the particles have collected (due to flow patterns) to the process to which a given particle has been assigned to computationally. This approach balances computations, but potentially maximizes the communication. Since communication is the bottleneck on distributed memory machines and the computations associated with the particles are relatively small compared to the computations
required to solve Equations 2.1, 2.2 and 2.3 on the Eulerian grid, this approach has not been taken. A different approach has been taken which assigns particles to a process if they lie within the spatial domain of the Eulerian grid solved by that process. This approach minimizes communication, but potentially maximizes the imbalance of computations. If all the particles collect in a physical region that is being solved on one process then all computations concerning the particles will be done by that one process. As a result there will be no parallelization of the computational work associated with the particles. The basic problem is that the mapping between the Lagrangian particles and the underlying Eulerian grid depends on the flow. If the flow results in particles clustered around one spatial location then the parallelization in either approach will suffer. In the approach used in the present work, this degradation in parallel efficiency will be as a result of uneven load balancing as opposed to increased communication.

5.4.2 Implementation

The governing equation for the motion of a particle is given by:

\[
\frac{d\vec{x}_p}{dt} = \vec{u}(\vec{x}, t)
\]

where \(\vec{u}(\vec{x}, t)\) is the velocity of the fluid at the particles location and \(\vec{x}_p\) is the location of the particles.

This equation is represented numerically by a third-order accurate implicit multilevel scheme (i.e. Eq. A.40):

\[
\vec{x}_p^{n+1} = \Delta t \left[ 5 \vec{u}(\vec{x}, t)^{n+1} + 8 \vec{u}(\vec{x}, t)^n - \vec{u}(\vec{x}, t)^{n-1} \right] + \vec{x}_p^n
\]

As the particle will in general not be at a location coinciding with the underlying Eulerian grid on which the flow is solved, interpolation will be needed to determine the velocity of the fluid at the particle’s location. This requires finding the particle’s nearest neighbors on the Eulerian grid. For a general curvilinear grid or unstructured grid this would be a daunting task. However, as the underlying Eulerian grid is Cartesian, the process of locating the nearest neighbors can be made
quite efficient. This is because the location in each dimension is a function of only one index. An integer analogy to the bisection algorithm can be used to efficiently find the neighboring points. This is given below in Fortran 95.

\[ \text{ileft}=\text{lboun}(x,1) \ ; \ \text{iright}=\text{uboun}(x,1) \]

\[ \text{do ntemp}=1,\text{huge}(1)-1 \]

\[ i=(\text{ileft}+\text{iright})/2 \]

\[ \text{if}(\text{ileft}+1 == \text{iright})\text{then} \ ; \ \text{exit} \]

\[ \text{elseif}(x(i) < =\text{particle}(n)\%x(1))\text{then} \ ; \ \text{ileft}=i \]

\[ \text{elseif}(x(i) > \text{particle}(n)\%x(1))\text{then} \ ; \ \text{iright}=i \ ; \ \text{endif} \]

\[ \text{enddo} \]

5.5 Code Validation

Tetra contains \(~15,000\) lines of Fortran 95 code along with numerous calls to MPI routines. This size, coupled with the difficult programming structure involving the managing of many processes creates a large potential for error. Certain errors, such as out of bounds memory access, the use of uninitialized variables and mismatched subroutine argument lists, can be caught by compilers. Other methods used to assess the properties of Tetra are described in the following.

5.5.1 Constructing Artificial Flow Fields

The first method is based upon the fact that finite difference schemes will give exact results if the solution is composed of polynomials of a certain degree or lower. For instance, if a five point scheme that involves fitting a fourth-order polynomial to the five points is used for the convective terms, it will have no error if the solution is composed of polynomials of fourth or lower degree. There are no known three-dimensional flows composed of polynomials. However, it is possible to create artificial flow fields that have this property. This is done by adding a source term that will cause Equations 2.1, 2.2 and 2.3 to be satisfied. This can be thought of as adding a spatially and temporally varying body force term to a physical flow such that it causes the flow field to match
the specified flow field. For example, if the lowest order polynomial interpolated exactly by the finite difference scheme is a fourth-order polynomial, the resulting expressions for the variables would be appropriate.

\[
p = x^5 y^5 z^5
\]

\[
u = x^5 y^4 z^4 T(t)
\]

\[
v = x^4 y^5 z^4 T(t)
\]

\[
w = x^4 y^4 z^5 T(t)
\]

\[
s = x^5 y^5 z^5 T(t)
\]

\[T(t)\] is a polynomial expression that provides time dependence. Note that it is not included in the expression for pressure. These expressions are now substituted into Equations 2.1, 2.2 and 2.3.

\[
\frac{\partial (x^5 y^4 z^4 T)}{\partial t} + x^5 y^4 z^4 T \frac{\partial (x^5 y^4 z^4 T)}{\partial x} + x^4 y^5 z^4 T \frac{\partial (x^5 y^4 z^4 T)}{\partial y} + x^4 y^4 z^5 T \frac{\partial (x^5 y^4 z^4 T)}{\partial z} \\
= - \frac{\partial (x^5 y^5 z^5)}{\partial x} + \frac{1}{Re} \left[ \frac{\partial^2 (x^5 y^4 z^4 T)}{\partial x^2} + \frac{\partial^2 (x^5 y^4 z^4 T)}{\partial y^2} + \frac{\partial^2 (x^5 y^4 z^4 T)}{\partial z^2} \right] + Sx \\
- \frac{\partial (x^5 y^4 z^4 T)}{\partial y} + \frac{1}{Re} \left[ \frac{\partial^2 (x^5 y^4 z^4 T)}{\partial x^2} + \frac{\partial^2 (x^5 y^4 z^4 T)}{\partial y^2} + \frac{\partial^2 (x^5 y^4 z^4 T)}{\partial z^2} \right] + Sy \\
- \frac{\partial (x^5 y^4 z^4 T)}{\partial z} + \frac{1}{Re} \left[ \frac{\partial^2 (x^5 y^4 z^4 T)}{\partial x^2} + \frac{\partial^2 (x^5 y^4 z^4 T)}{\partial y^2} + \frac{\partial^2 (x^5 y^4 z^4 T)}{\partial z^2} \right] + Sz \\
\frac{\partial (x^5 y^4 z^5)}{\partial x} + \frac{\partial (x^4 y^5 z^4 T)}{\partial y} + \frac{\partial (x^4 y^4 z^5 T)}{\partial z} + Cx = 0 \\
\frac{\partial (x^5 y^5 z^5 T)}{\partial t} + x^5 y^4 z^4 T \frac{\partial (x^5 y^5 z^5 T)}{\partial x} + x^4 y^5 z^4 T \frac{\partial (x^5 y^5 z^5 T)}{\partial y} + x^4 y^4 z^5 T \frac{\partial (x^5 y^5 z^5 T)}{\partial z} \\
= \frac{1}{Re Pr} \left[ \frac{\partial^2 (x^5 y^5 z^5 T)}{\partial x^2} + \frac{\partial^2 (x^5 y^5 z^5 T)}{\partial y^2} + \frac{\partial^2 (x^5 y^5 z^5 T)}{\partial z^2} \right] + S
Simplifying:

\[ S_x = 5x^4y^5z^5 + x^5y^4z^4 \frac{\partial T(t)}{\partial t} + 13x^9y^8z^8T^2(t) \]

\[ -\frac{1}{Re} \left( 20x^3y^4z^4T(t) + 12x^5y^2z^4T(t) + 12x^5y^4z^2T(t) \right) \]

\[ S_y = 5x^5y^4z^5 + x^4y^5z^4 \frac{\partial T(t)}{\partial t} + 13x^8y^8z^8T^2(t) \]

\[ -\frac{1}{Re} \left( 12x^2y^5z^4T(t) + 20x^4y^3z^4T(t) + 12x^4y^5z^2T(t) \right) \]

\[ S_z = 5x^5y^5z^4 + x^4y^4z^5 \frac{\partial T(t)}{\partial t} + 13x^8y^8z^9T^2(t) \]

\[ -\frac{1}{Re} \left( 12x^2y^4z^5T(t) + 12x^4y^2z^5T(t) + 20x^4y^4z^3T(t) \right) \]

\[ C_x = -15x^4y^4z^4T(t) \]

\[ S = x^5y^5z^5 \frac{\partial T(t)}{\partial t} + 15x^9y^9z^9T^2(t) \]

\[ -\frac{1}{RePr} \left( 20x^3y^5z^5T(t) + 20x^3y^3z^5T(t) + 20x^5y^5z^3T(t) \right) \]

If a problem with Dirichlet boundary conditions is set up with boundary conditions obtained from the above polynomial expressions for the variables, and the above source terms are added to Equations 2.1, 2.2 and 2.3, then a zero residual (within machine accuracy) should result. The variables should also remain within machine accuracy of their original values as the solution progresses in time. This, of course, requires that the time step be appropriate for stability. It is important that the maximum range in the spatial dimensions be within \([-1, 1]\). This is because the large values resulting from raising \(x, y\) and \(z\) to high powers outside this range creates a large roundoff error when added to the small values resulting from raising \(x, y\) and \(z\) to high powers inside this range. This large error will mask possible underlying errors in the code being tested. This procedure has been used to test Tetra with results that were within machine accuracy. The above method is excellent for validation for cases where all boundary conditions are Dirichlet. However, flow problems are frequently solved where one or more boundary conditions is periodic. The above procedure will not work in such cases. It should be noted that monotonic TVD-type
limiters, as used for some applications in the present work, make the finite difference scheme a function of the local flow field. Depending on the local flow field, the finite difference scheme may reduce to a first-order scheme if that is necessary to maintain monotonicity. As a result, the higher-order schemes cannot be tested with this method in conjunction with monotonic limiters.

5.6 Varying Grid Spacing

In general, any numerical approximation on a discrete grid will contain truncation error terms of the form:

$$\varepsilon = c^n_i \Delta x^n_i + c^{n+1}_i \Delta x^{n+1}_i + \ldots + c^\infty_i \Delta x^\infty_i,$$

where $n$ depends on the numerical scheme (i.e. $n = 3$) for a spatially third-order accurate scheme. The terms $c^n_i, c^{n+1}_i + \ldots + c^\infty_i$ are functions of partial derivatives. Note that for high order discretized representations of Equations 2.1, 2.2 and 2.3 as used in the present work, $c^n_i, c^{n+1}_i + \ldots + c^\infty_i$ are extremely complicated expressions. In the following it is assumed that the error is dominated by the leading order error terms of power $n$ (i.e. $c^n_i \Delta x^n_i$). This being the case, solutions obtained on different grids can be compared by varying the grid spacing in a certain manner. There are two methods which can be used.

5.6.1 Order in the Solution

Three grids are used in the method. In the following $U$ is the unknown exact solution and $a < 1$ is a parameter that controls the relative grid spacing between the three grids.

$$\frac{\text{finer grid} - (\text{coarse grid})}{\text{finest grid} - (\text{finer grid})} = \frac{U + cx_n (a\Delta x)^n - U - cx_n (\Delta x)^n}{U + cx_n (aa\Delta x)^n - U - cx_n (a\Delta x)^n} = \frac{a^n - 1}{(aa)^n - a^n} = A$$

$$-Aa^n a^n + (1 + A)a^n - 1 = 0$$

The solution to this is $a^n = (1, \frac{1}{A})$. We reject the first possibility because this would require a zeroth order numerical scheme. Therefore we conclude that:

$$n = \log_a \left( \frac{1}{A} \right) \quad (5.1)$$

Due to the involvement of spatial discretization errors with the time integration scheme it is
not possible to use the above method to compare values at a certain point in time. This is a result of the non-linearity of convective terms in Eq. 2.1 and Eq. 2.3. It can be used to compare time-averaged results.

5.6.2 Order in the Residual

As before, an artificial flow field is generated. This flow field, instead of being composed of polynomials will be composed, at least in the periodic directions, of periodic functions such as sines or cosines. Since in general this artificial flow field will not satisfy the Navier-Stokes equations, a source term is added so that the flow field does satisfy the Navier-Stokes equations in the presence of this source term. This can be thought of as adding a spatially varying body force term to a physical problem such that it causes the flow field to match the specified flow field. Two grids are used. In the following $R$ is the exact residual (which is zero) and $a < 1$ is a parameter that controls the ratio of grid spacing between the two grids.

\[
\frac{(\text{coarse grid})}{(\text{finer grid})} = \frac{R + cx_n (\Delta x)^n}{R + cx_n (a\Delta x)^n} = \frac{1}{a^n} = B
\]

This results in:

\[
n = \log_a \left( \frac{1}{B} \right)
\]  

Equations 5.1 and 5.2 are the primary way to test the accuracy of the implementation of the differencing, which is based on polynomials, with respect to the periodic boundaries. As higher order terms are neglected in obtaining equations 5.1 and 5.2 the exponent $n$ computed will not be exactly the value of the leading order error term. In addition the form of the function used to generate the artificial flow field will have an effect on the exponent determined from equations 5.1 and 5.2. Tetra has been tested with Eq. 5.2 using six different grids. Both grids in which the grid spacing is constant (even grids) and in which the grid spacing changes from grid point to grid point (uneven grids) are used in the tests. The dimensions of the grids used are $8 \times 8 \times 8, 16 \times 16 \times 16, 32 \times 32 \times 32, 64 \times 64 \times 64, 128 \times 128 \times 128$ and $256 \times 256 \times 256$. 
These grids are obtained from each other by doubling the number of grid points in each direction, while keeping the grid stretching parameters the same. The $x$-axis in these figures represents the pair of grids needed in Eq. 5.2 to obtain the order of accuracy. The grids are paired in the following order $8 \times 8 \times 8 \leftrightarrow 16 \times 16 \times 16 \leftrightarrow 32 \times 32 \times 32 \leftrightarrow 64 \times 64 \times 64 \leftrightarrow 128 \times 128 \times 128 \leftrightarrow 256 \times 256 \times 256$. As a result, $a$ in Eq. 5.2 is 2. Periodic boundary conditions are set in the $x$ and $z$ directions. All components of velocity, the pressure and the scalar field are set in the entire flow domain including the boundaries using Eq. 5.3. A sixth-order central difference convection scheme (Eq. A.35), a sixth-order central difference diffusion scheme (Eq. A.22 and a sixth-order central difference scheme for the pressure gradient (Eq. A.19) is used to discretize Eq. 2.1. A sixth-order central difference scheme (Eq. A.16) is used to discretize Eq. 2.2. Sixth-order central difference convection (Eq. A.35) and diffusion (Eq. A.22) schemes are used to discretize Eq. 2.3. A sixth-order interpolation scheme is used to interpolate the velocity onto the main grid points that the scalar is located at. The number of points used in the stencils is not reduced as a non-periodic boundary is approached. The results are shown in Figures 5.5 and 5.6. The order of accuracy of Eq. 2.2 in Fig. 5.5 approaches five. The formal order of accuracy of Eq. A.16 is six but this is true only for the centered scheme on an even grid. If the number of points in the scheme is maintained but the point at which the derivative is evaluated shifts (as occurs near a non-periodic boundary), then the formal order of the scheme drops to five. This can be seen in Fig. 5.5. The reduction in the order of accuracy (to formally first-order) resulting from the application of a monotonic convection scheme can be seen in Figures 5.7 and 5.8. The most stringent measure of error (the maximum of the absolute value of error over the entire flow field) is chosen in all figures (except for Fig. 5.9). Other measures of error over the entire flow field, such as the root mean square and the arithmetic mean of the absolute value, can also be used. As the number of grid pairs used increases, these three measures of error converge (Fig. 5.9).
\[ \sum_{k_z=0}^{2} \sum_{k_y=0}^{2} \sum_{k_x=0}^{2} \cos\left(\frac{2\pi k_x (x - x_b)}{L_x} \right) \cos\left(\frac{2\pi k_y (y - y_b)}{L_y} \right) \cos\left(\frac{2\pi k_z (z - z_b)}{L_z} \right) \quad (5.3) \]

where

- \( L_x \) is the length of the domain in the \( x \) direction
- \( L_y \) is the length of the domain in the \( y \) direction
- \( L_z \) is the length of the domain in the \( z \) direction
- \( x_b \) is the first point in the domain in the \( x \) direction
- \( y_b \) is the first point in the domain in the \( y \) direction
- \( z_b \) is the first point in the domain in the \( z \) direction

Figure 5.5: Order of accuracy of discretization of Equations 2.1, 2.2 and 2.3 on an even grid
Figure 5.6: Order of accuracy of discretization of Equations 2.1, 2.2 and 2.3 on an uneven grid

Figure 5.7: Order of accuracy of discretization of Equations 2.1 and 2.3 on an even grid using monotonic convection scheme
Figure 5.8: Order of accuracy of discretization of Equations 2.1 and 2.3 on an uneven grid using monotonic convection scheme.

Figure 5.9: Different measures of error used to determine the order of accuracy of discretization of the $x$ component of Eq. 2.1 on an uneven grid.
Chapter 6  Budgets of k-\(\varepsilon\) Equations

6.1  Introduction

The accurate prediction of turbulent flows of interest to industry is one of the major problems in fluid dynamics. While DNS provides a more accurate model for a flow, its usefulness is severely limited by the large computational expense required. As a result, the simulations used by industry for design purposes, in which many solutions of different configurations are required quickly, are almost entirely obtained by using the Reynolds Averaged Navier-Stokes equations. To solve the Reynolds Averaged Navier-Stokes equations (Eq. 6.1), closure models must be used to represent the correlations resulting from applying the time averaging process to the Navier-Stokes equations. These models involve various constants for which values must be chosen. It is well known that Reynolds Averaged Navier-Stokes models experience great difficulty in predicting the flow and the heat transfer for complicated flows. This is largely because the Reynolds Averaged Navier-Stokes models in common use were developed and/or had constants tuned for flows which are simpler and quite different from complicated flows such as jets in crossflow. Flows commonly used to develop and tune Reynolds Averaged Navier-Stokes models include channel flow, isotropic turbulence, and boundary layer flows that, in the mean sense, vary in only two dimensions. In addition most turbulence models make certain assumptions (such as isotropic turbulence) that is known not to be the case for many flows of industrial interest. Most of the many correlations that must be modeled in turbulence models are very difficult to measure experimentally. As a result, DNS provides the only practical means of providing these correlations for turbulence modelers. DNS data has been used to improve turbulence models for the case of a channel flow in the work of (Rodi & Mansour, 1993) and (N. N. Mansour & Moin, 1988). DNS can also be used to provide important insights into where particular turbulence models fail. This knowledge
can aid in the design of turbulence models that capture the physics of a certain flow, such as jets in crossflow, better than commonly used models which are optimized for different and simpler flows. The goal of the present work is to compare the models used to represent the terms in the exact $k - \varepsilon$ equations with the results obtained from DNS, for a representative film cooling case. Comparisons with the terms in the exact $k - \varepsilon$ equations are made with the results of others for a channel flow to validate the numerical method and the code used in the present work.

### 6.2 Turbulence Modeling

The Reynolds Averaged Navier-Stokes (RANS) equations (Eq. 6.1) are obtained by time averaging the Navier-Stokes equations (Equations 2.1 and 2.2). A similar equation (Eq. 6.2) can be obtained by time averaging the scalar equation (Eq. 2.3). Note that $t \gg$ turbulent time scale.

\[
\frac{\partial \overline{u_i}}{\partial t} + \frac{\partial \overline{(u_i u'_j)}}{\partial x_j} + \frac{\partial \overline{(u'_i u'_j)}}{\partial x_j} = -\frac{\partial \overline{p}}{\partial x_i} + \frac{1}{Re} \frac{\partial^2 \overline{u_i}}{\partial x_j \partial x_j}, \quad t \gg \text{turbulence scale} \quad (6.1)
\]

\[
\frac{\partial \overline{s}}{\partial t} + \frac{\partial \overline{(u'_j s)}}{\partial x_j} + \frac{\partial \overline{(u'_j s')}}{\partial x_j} = \frac{1}{Re P_r} \frac{\partial^2 \overline{s}}{\partial x_j \partial x_j}, \quad t \gg \text{turbulence scale} \quad (6.2)
\]

\[
-u'_i u'_{i,j} = \frac{1}{Re} - 2 S_{i,j} - \frac{2}{3} k \delta_{i,j} \quad \text{where} \quad S_{i,j} = \frac{1}{2} \left( \frac{\partial \overline{u_i}}{\partial x_j} + \frac{\partial \overline{u_j}}{\partial x_i} \right) \quad \text{and} \quad k = \frac{1}{2} u'_i u'_i \quad (6.3)
\]

The goal of turbulence modeling is to derive an expression for $u'_i u'_{i,j}$ and $u'_j s'$. The Boussinesq approximation (Eq. 6.3) is used in the standard $k - \varepsilon$ model and most other two equation turbulence models to represent the turbulent Reynolds stresses $(u'_i u'_{i,j})$ that appear in the Reynolds Averaged Navier-Stokes. The Boussinesq approximation requires that the eddy viscosity $\left( \frac{1}{Re} \right)$ be specified. In the standard $k - \varepsilon$ model, the eddy viscosity is defined as $C_{\mu} \frac{k^2}{\varepsilon} Re$, where $k$ is the turbulent kinetic energy and $\varepsilon = u'_i u'_i u'_{i,k}$ is the dissipation of turbulence. Equations for $k$ and $\varepsilon$ can be derived from the Navier-Stokes equations, details of which are given in Appendix B. As a result of the non-linearity of the Navier-Stokes equations, these equations for $k$ and $\varepsilon$ contain numerous high order correlations, which must be modeled. The present work models these correlations using the standard $k - \varepsilon$ model of (Launder & Spalding, 1974) (see Appendix C for details), which results in modified equations (Eq. 6.6 and Eq. 6.7) for $k$ and $\varepsilon$. 

120
6.2.1 Exact $k - \varepsilon$ Equations

6.2.1.1 Exact $k$ Equation

The exact $k$ equation is given by Eq. 6.4.

$$\frac{\partial k}{\partial t} + \nabla \cdot \left( \mu \frac{\partial k}{\partial x_j} \right) = -u_i' u_j' \frac{\partial u_i}{\partial x_j} - \frac{1}{Re} u_i' u_j' \frac{\partial^2 u_i}{\partial x_j} \frac{\partial u_j}{\partial x_j} + \frac{\partial}{\partial x_j} \left[ \frac{1}{Re} \frac{\partial k}{\partial x_j} - \frac{1}{2} u_i' u_j' - \frac{\rho}{u_i' u_j'} \right] \quad \text{where} \quad k = \frac{1}{2} u_i' u_i'$$

(6.4)

The terms of Eq. 6.4 are commonly referred to by the following names, which reflect their predominant effect on $k$:

<table>
<thead>
<tr>
<th>Term</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convective transport of $k$</td>
<td>$\frac{\partial k}{\partial t} + \nabla \cdot \left( \mu \frac{\partial k}{\partial x_j} \right)$</td>
</tr>
<tr>
<td>Production of $k$</td>
<td>$-u_i' u_j' \frac{\partial u_i}{\partial x_j}$</td>
</tr>
<tr>
<td>Dissipation of $k$</td>
<td>$-\frac{1}{Re} u_i' u_j' \frac{\partial^2 u_i}{\partial x_j} \frac{\partial u_j}{\partial x_j}$</td>
</tr>
<tr>
<td>Molecular diffusion of $k$</td>
<td>$\frac{\partial}{\partial x_j} \left[ \frac{\partial k}{\partial x_j} - \frac{1}{2} u_i' u_j' - \frac{\rho}{u_i' u_j'} \right]$</td>
</tr>
<tr>
<td>Turbulent diffusion of $k$</td>
<td>$-\frac{1}{Re} \frac{\partial k}{\partial x_j}$</td>
</tr>
</tbody>
</table>

Note that Equations 6.4 and 6.5 use the comma notation to indicate differentiation (i.e. $u_{k,m} = \frac{\partial u_k}{\partial x_m}$) and $u_{i,k,m} = \frac{\partial u_i'}{\partial x_k \partial x_m}$.

6.2.1.2 Exact $\varepsilon$ Equation

The exact $\varepsilon$ equation is given by Eq. 6.5.

$$\frac{\partial \varepsilon}{\partial t} + \nabla \cdot \left( \mu \frac{\partial \varepsilon}{\partial x_j} \right) = -2 \left[ u_i' u_j' + u_i' u_j' \right] \frac{\partial u_i}{\partial x_j} - 2u_k' u_i' \frac{\partial^2 u_i}{\partial x_k \partial x_j} - 2u_i' u_{i,m} u_k' u_{k,m} \quad \text{where} \quad \varepsilon = u_i' u_i'$

(6.5)

The terms of Eq. 6.5 are commonly referred to by the following names, which reflect their predominant effect on $\varepsilon$:

<table>
<thead>
<tr>
<th>Term</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convective transport of $\varepsilon$</td>
<td>$\frac{\partial \varepsilon}{\partial t} + \nabla \cdot \left( \mu \frac{\partial \varepsilon}{\partial x_j} \right)$</td>
</tr>
<tr>
<td>Production of $\varepsilon$</td>
<td>$-2 \left[ u_i' u_j' + u_i' u_j' \right] \frac{\partial u_i}{\partial x_j}$</td>
</tr>
<tr>
<td>Destruction of $\varepsilon$</td>
<td>$-2u_i' u_{i,m} u_k' u_{k,m} \frac{1}{Re}$</td>
</tr>
<tr>
<td>Molecular diffusion of $\varepsilon$</td>
<td>$\frac{\partial}{\partial x_j} \left[ \frac{\partial \varepsilon}{\partial x_j} - \frac{1}{2} u_i' u_i' \frac{\partial^2 u_i}{\partial x_j} \frac{\partial u_i}{\partial x_j} \right]$</td>
</tr>
<tr>
<td>Turbulent diffusion of $\varepsilon$</td>
<td>$-\frac{1}{Re} \frac{\partial \varepsilon}{\partial x_j}$</td>
</tr>
</tbody>
</table>
6.2.2 Modeled \( k - \varepsilon \) Equations

6.2.2.1 Modeled \( k \) Equation

The modeled \( k \) equation (Launder & Spalding, 1974) is given by Eq. 6.6.

\[
\frac{\partial k}{\partial t} + \frac{u_j}{\partial x_j} = C_{\mu} \frac{k^2}{\varepsilon} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \left. \frac{\partial u_i}{\partial x_j} \right|_{Re} - \varepsilon \frac{1}{Re} \frac{\partial k}{\partial x_j} + \frac{\partial k}{\partial x_j} \left[ \frac{1}{Re} \frac{\partial k}{\partial x_j} + \frac{C_{\mu} k^2}{\sigma_k} \frac{\partial k}{\partial x_j} \right]_{(6.6)}
\]

where \( \varepsilon = \frac{u'_i u'_i}{u'_i u'_i} \).

The terms of Eq. 6.6 are commonly referred to by the following names, which reflect their predominant effect on \( k \) (the terms that are modeled are indicated):

- **Convective transport of \( k \)**: \( \frac{\partial k}{\partial t} + \frac{u_j}{\partial x_j} \)
- **Production of \( k \)**: \( C_{\mu} \frac{k^2}{\varepsilon} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \left. \frac{\partial u_i}{\partial x_j} \right|_{Re} \)
- **Dissipation of \( k \)**: \( -\frac{1}{Re} \frac{\partial k}{\partial x_j} \)
- **Molecular diffusion of \( k \)**: \( \frac{1}{\partial x_j} \frac{\partial k}{\partial x_j} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \)
- **Turbulent diffusion of \( k \)**: \( \frac{1}{\partial x_j} \frac{\partial k}{\partial x_j} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \left[ \frac{1}{Re} \frac{\partial k}{\partial x_j} + \frac{C_{\mu} k^2}{\sigma_k} \frac{\partial k}{\partial x_j} \right] \)

Taking into account the following identities:

\[
\frac{1}{4} \frac{\partial u_i}{\partial x_j} \frac{\partial u_j}{\partial x_i} + \frac{1}{4} \frac{\partial u_i}{\partial x_j} \frac{\partial u_j}{\partial x_i} = \frac{1}{2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \frac{1}{2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right)
\]

The production term can be written as:

\[
C_{\mu} \frac{k^2}{\varepsilon} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \left. \frac{\partial u_i}{\partial x_j} \right|_{Re} = 2C_{\mu} \frac{k^2}{\varepsilon} S_{i,j} S_{i,j} Re, \quad \text{where} \quad S_{i,j} = \frac{1}{2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right)
\]

As this term is always positive it acts as a source term for \( k \). The values of the various constants appearing in Eq. 6.6 are:

\[
C_{\mu} = .09 \quad \sigma_k = 1
\]

6.2.2.2 Modeled \( \varepsilon \) Equation

The modeled \( \varepsilon \) equation (Launder & Spalding, 1974) is given by Eq. 6.7.

\[
\frac{\partial \varepsilon}{\partial t} + \frac{u_j}{\partial x_j} \frac{\partial \varepsilon}{\partial x_j} = C_{\varepsilon 1} C_{\mu} k \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \left. \frac{\partial u_i}{\partial x_j} \right|_{Re} - C_{\varepsilon 2} \frac{\varepsilon^2}{kRe} + \frac{1}{\partial x_j} \left[ \frac{1}{Re} \frac{\partial \varepsilon}{\partial x_j} + \frac{C_{\mu} Re k^2}{\sigma_\varepsilon} \frac{\partial \varepsilon}{\partial x_j} \right]
\]

where \( \varepsilon = \frac{u'_i u'_i}{u'_i u'_i} \).
The terms of Eq. 6.7 are commonly referred to by the following names, which reflect their predominant effect on \( \varepsilon \) (the terms that are modeled are indicated):

<table>
<thead>
<tr>
<th>Term</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convective transport of ( \varepsilon )</td>
<td>( \frac{\partial \varepsilon}{\partial t} + u_j \frac{\partial \varepsilon}{\partial x_j} )</td>
</tr>
<tr>
<td>Production of ( \varepsilon )</td>
<td>( C_\varepsilon C_\mu k \left( \frac{\sigma <em>{ij}}{\partial x_j} + \frac{\sigma <em>{ji}}{\partial x_i} \right) \frac{\partial \varepsilon}{\partial x_j} Re ) (models (-2u</em>{i,k}^' u</em>{j,k}^' + u_{k,i}^' u_{k,j}^' \frac{\partial \varepsilon}{\partial x_j}))</td>
</tr>
<tr>
<td>Destruction of ( \varepsilon )</td>
<td>(-C_\varepsilon^2 \frac{\varepsilon^2}{kRe} ) (models (-2u_{i,k}^' u_{i,m}^' u_{k,m}^' - 2u_{i,km}^' u_{i,km}^' \frac{1}{Re}))</td>
</tr>
<tr>
<td>Molecular diffusion of ( \varepsilon )</td>
<td>( \frac{\partial \varepsilon}{\partial x_j} \frac{\sigma <em>{ij}}{\sigma <em>e} \frac{1}{\partial x_j Re} ) (models ( \frac{\partial \varepsilon}{\partial x_j} \left[ -u</em>{i,m}^' u</em>{i,m}^' - 2p_{i,m}^' u_{i,m}^' \right] ))</td>
</tr>
<tr>
<td>Turbulent diffusion of ( \varepsilon )</td>
<td>( \frac{\partial \varepsilon}{\partial x_j} \frac{\sigma <em>{ij}}{\sigma <em>e} \frac{1}{\partial x_j Re} ) (models ( \frac{\partial \varepsilon}{\partial x_j} \left[ -u</em>{i,m}^' u</em>{i,m}^' - 2p_{i,m}^' u_{i,m}^' \right] ))</td>
</tr>
</tbody>
</table>

The production term can be written as:

\[
C_\varepsilon C_\mu k \left( \frac{\sigma _{ij}}{\partial x_j} + \frac{\sigma _{ji}}{\partial x_i} \right) \frac{\partial \varepsilon}{\partial x_j} Re = 2C_\varepsilon C_\mu k S_{i,j} S_{i,j} Re, \quad \text{where } S_{i,j} = \frac{1}{2} \left( \frac{\partial \mu _i}{\partial x_j} + \frac{\partial \mu _j}{\partial x_i} \right)
\]

As this term is always positive it acts as a source term for \( \varepsilon \). The values of the various constants appearing in Eq. 6.7 are:

\[
C_\varepsilon^1 = 1.44 \quad C_\varepsilon^2 = 1.92 \quad C_\mu = 0.09 \quad \sigma _\varepsilon = 1.3
\]

### 6.2.3 Eddy Viscosity and Turbulent Prandtl Number

The Boussinesq approximation (Eq. 6.3) is used in the standard \( k - \varepsilon \) model and most other two equation turbulence models to represent the turbulent Reynolds stresses \( \langle u_i^' u_j^' \rangle \) that appear in the Reynolds Averaged Navier-Stokes equations (Eq. 6.1). As the eddy viscosity appears frequently in numerous terms in two equation turbulence models including the standard \( k - \varepsilon \) model, it is examined in detail in the present work. The eddy viscosity, which equals \( \frac{1}{Re_{\tau}} \), can be computed from Eq. 6.3 using the results of a DNS to calculate \(-u_{i,j}^' u_{i,j}^'\), \( S_{i,j} \) and \( k \). It is desired to obtain the value for \( \frac{1}{Re_{\tau}} \) from Eq. 6.3, which will minimize the error in representing \(-u_{i,j}^' u_{i,j}^'\) by the right hand side of Eq. 6.3. This is done by minimizing the square of the error with respect to \( \frac{1}{Re_{\tau}} \).

Advantage is not taken of the symmetry of \( u_{i,j}^' u_{i,j}^' \) in defining the error in Eq. 6.3. The rationale for this is that in the Reynolds Averaged Navier-Stokes equations, the off diagonal terms appear twice as often as the diagonal. Therefore it is natural to preserve this ratio in minimizing the error.
associated with representing $\overline{u_i'u_j'}$ by the Boussinesq approximation (Eq. 6.3).

$$\text{error} = \sum_{j=1}^{3} \sum_{i=1}^{3} \left( \frac{1}{Re_T} 2S_{i,j} - \frac{2}{3} k \delta_{i,j} + \overline{u_i'u_j'} \right)^2$$

$$\frac{\partial (\text{error})}{\partial \left( \frac{1}{Re_T} \right)} = 0 = \sum_{j=1}^{3} \sum_{i=1}^{3} 2 \left( \frac{1}{Re_T} 2S_{i,j} - \frac{2}{3} k \delta_{i,j} + \overline{u_i'u_j'} \right) 2S_{i,j}$$

which results in:

$$\frac{1}{Re_T} = \frac{\sum_{j=1}^{3} \sum_{i=1}^{3} \left( \frac{2}{3} k \delta_{i,j} S_{i,j} - \overline{u_i'u_j'} S_{i,j} \right)}{\sum_{j=1}^{3} \sum_{i=1}^{3} 2S_{i,j} S_{i,j}} \tag{6.8}$$

Note that as the expression for the error is a quadratic polynomial in $\frac{1}{Re_T}$, there is a guaranteed global minimum, the value of $\frac{1}{Re_T}$ at this global minimum being given by Eq. 6.8. Likewise, it is desired to obtain the value for the turbulent Prandtl number ($P_{\tau\tau}$) from Eq. 6.9 which will minimize the error in representing $-\overline{u_is}$ in Eq. 6.2. This is done by minimizing the square of the error with respect to $P_{\tau\tau}$.

$$-\overline{u_is'} = \frac{1}{Re_T P_{\tau\tau}} \frac{\partial \sigma}{\partial x_i} \tag{6.9}$$

$$\text{error} = \sum_{i=1}^{3} \left( \frac{1}{Re_T P_{\tau\tau}} \frac{\partial \sigma}{\partial x_i} + \overline{u_i's'} \right)^2$$

$$\frac{\partial (\text{error})}{\partial (P_{\tau\tau})} = 0 = \sum_{i=1}^{3} 2 \left( \frac{1}{Re_T P_{\tau\tau}} \frac{\partial \sigma}{\partial x_i} + \overline{u_i's'} \right) \left( -\frac{P_{\tau\tau}^{-2}}{Re_T} \frac{\partial \sigma}{\partial x_i} \right)$$

rearranging,

$$-\frac{P_{\tau\tau}^{-2}}{Re_T} 2 \sum_{i=1}^{3} \left( \frac{1}{Re_T P_{\tau\tau}} \frac{\partial \sigma}{\partial x_i} + \overline{u_i's'} \right) \left( \frac{\partial \sigma}{\partial x_i} \right) = 0 = \sum_{i=1}^{3} \left( \frac{1}{Re_T P_{\tau\tau}} \frac{\partial \sigma}{\partial x_i} + \overline{u_i's'} \right) \left( \frac{\partial \sigma}{\partial x_i} \right)$$

which results in:

$$P_{\tau\tau} = \frac{1}{\sum_{i=1}^{3} \left( -\overline{u_i's' \frac{\partial \sigma}{\partial x_i}} \right) \left( \frac{\partial \sigma}{\partial x_i} \frac{\partial \sigma}{\partial x_j} \right) \frac{1}{Re_T}} \tag{6.10}$$

Note that on a wall boundary on which $u_i'$ is zero, $\frac{1}{Re_T}$ is zero while $P_{\tau\tau} = 0$. It should be noted that the above approach for minimizing the error is different from the traditional approach where, due to the idealized 2-D situations considered, only one component of $\overline{u_i'u_j'}$ and $S_{i,j}$ is important. For more complex 3-D flows, the above noted minimization process appears to be the most logical approach for computing $\frac{1}{Re_T}$ and $P_{\tau\tau}$. 

124
6.3 Collecting Statistics

In order to characterize turbulent flows, statistical information such as mean values ($\overline{u}$) and second-order products of fluctuating quantities such as $u'_i u'_j$ are needed. These second-order products are needed in order to understand the structure and degree of turbulence. In addition, they appear in the Reynolds Averaged Navier-Stokes equations (Eq. 6.1) and are the terms that all turbulence models attempt to represent. If it is desired to improve Reynolds Averaged Navier-Stokes models by examining the terms of the modeled equations, for instance the $k - \varepsilon$ equations (Eq. 6.4 and Eq. 6.5), then it is necessary to compute higher order products of fluctuating quantities, some of which involve derivatives. Two ways that this statistical information can be obtained are described.

6.3.1 Computing the Mean First and Fluctuating Components Second

Any unsteady quantity $u(t)$ can be decomposed into the following two parts: $u(t) = u'(t) + \overline{u}$, where $\overline{u}$ indicates time averaged or mean part and $u'(t)$ the fluctuating part. Consider the following equation for second-order statistics.

$$\overline{u'_i u'_j} = (u_i - \overline{u_i})(u_j - \overline{u_j})$$

The above product of fluctuating quantities can be easily obtained if the mean quantities are known. The mean quantities must first be obtained by running the simulation for a period of time and stopping when the mean statistics are considered to be converged. Once the mean values are computed, the computation of terms involving the fluctuating components can begin. This method has the serious disadvantage in that the simulation must be run twice in order to collect statistics. Another disadvantage is that the error associated with stopping the averaging of the mean components after a finite averaging period will not go to zero as the time period over which the fluctuating components are collected goes to infinity.
6.3.2 Running Average

Consider the following equation for second-order statistics:

$$\overline{u_i' u_j'} = (u_i - \overline{u_i})(u_j - \overline{u_j}) = u_i u_j - \overline{u_i} \overline{u_j}$$

This equation shows that by computing running averages of $u_i, u_j$ and the product $u_i u_j$ the second-order statistic tensor $\overline{u_i' u_j'}$ can be computed. This method has the great advantage of a savings in time compared to the previous method of computing the mean first. This is because it is not necessary to first run a simulation to compute the mean. As differentiation commutes with the time averaging process the following is true:

$$\frac{1}{T} \int_0^T \frac{\partial u_i}{\partial x_m} dt = \frac{\partial}{\partial x_m} \left( \frac{1}{T} \int_0^T u_i dt \right) \quad \text{i.e.} \quad \overline{\frac{\partial u_i}{\partial x_m}} = \frac{\partial}{\partial x_m} \overline{u_i}$$

This means that it is not necessary to compute runtime averages of derivatives as these can be obtained from differentiating the mean quantities. It will however be necessary to compute runtime averages of products of differentiated quantities as shown below.

$$\frac{1}{T} \int_0^T \frac{\partial u_i}{\partial x_m} \frac{\partial u_m}{\partial x_i} dt \neq \frac{1}{T} \int_0^T \frac{\partial u_i}{\partial x_m} dt \frac{1}{T} \int_0^T \frac{\partial u_m}{\partial x_i} dt \quad \text{i.e.} \quad \overline{\frac{\partial u_i}{\partial x_m} \frac{\partial u_m}{\partial x_i}} \neq \frac{\partial}{\partial x_m} \overline{u_i} \frac{\partial}{\partial x_i} \overline{u_m}$$

In the present work the numerous correlations appearing in the exact $k$ and $\varepsilon$ equations (see Eq. 6.4 and Eq. 6.5) are computed as a running average. The terms that must be computed as a running average in order reconstitute the correlations are described in Appendix B.

6.4 Channel Flow

6.4.1 Problem Description

A channel flow at a Reynolds number of 180 based on the wall shear velocity $u_r = \sqrt{\nu \frac{\partial \overline{u}}{\partial y} |_{y=0}}$ and the channel half height $h$ is simulated. All quantities are non-dimensionalized by the channel half height and the wall shear velocity. The flow is driven by a constant body force term which is added to the $x$ momentum equation. A scalar equation is solved for the transport of a passive scalar, with a Prandtl number of .7. The boundary conditions for the scalar are zero at the lower wall and one at the upper wall. A schematic of the flow domain along with boundary
Table 6.1: Channel flow grid spacing and time step table

<table>
<thead>
<tr>
<th>Grid</th>
<th>$\Delta x$ (wall units)</th>
<th>First four points from the wall (wall units)</th>
<th>$\Delta z$ (wall units)</th>
<th>$\Delta t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>64 $\times$ 32 $\times$ 32</td>
<td>18</td>
<td>2.4614, 7.4907, 13.2934, 19.9561</td>
<td>18</td>
<td>.001</td>
</tr>
<tr>
<td>128 $\times$ 64 $\times$ 64</td>
<td>9</td>
<td>1.1679, 3.5274, 6.0588, 8.7719</td>
<td>9</td>
<td>.0005</td>
</tr>
<tr>
<td>256 $\times$ 128 $\times$ 128</td>
<td>4.5</td>
<td>0.5691, 1.7131, 2.8976, 4.1239</td>
<td>4.5</td>
<td>.00025</td>
</tr>
</tbody>
</table>

conditions and dimensions is given in Fig. 3.8. The computational domain is $L_x = 6.4h$ units in the streamwise direction ($x$), $2h$ units in the wall normal direction ($y$) and $L_z = 3.2h$ units in the cross-stream direction ($z$). Three grids, with dimensions of $64 \times 32 \times 32, 128 \times 64 \times 64$ and $256 \times 128 \times 128$ are used. Each grid is obtained from the previous grid by doubling the number of points in each direction. Evenly spaced grids are used in the $x$ and $z$ directions, while a stretched grid is used in the $y$ direction to concentrate grid points near the walls. Specifics of the grid spacing and time step are shown in Table 6.1 for the different grids. The time steps used resulted in a maximum CFL number of approximately .2, based on the $u$ velocity and grid spacing in the $x$ direction. The other CFL numbers based on the other two directions are smaller. It is necessary to use a method to trigger turbulence for this particular flow. The method used in the present work is to place two rows of wall jets on the top and bottom walls. These wall jets either inject or remove fluid from the domain, depending on their spatial location. The net sum of mass injected into the domain by the jets is approximately zero. These wall jets are turned on until the flow (based on the bulk velocity) has traveled $11.7h$, after which they are turned off. In addition, as an initial condition, the $v$ velocity at each interior point is set to a random number between $-2$ and 2. As the flow field is homogenous in the $x$ and $z$ dimensions, spatial averaging of statistical quantities is carried out in these directions. Statistics are computed after the flow (based on the bulk velocity) has traveled $167h$. Comparisons are made with (Kim et al., 1987), (Kawamura et al., 1998) and (Debusschere & Rutland, 2002).

6.4.2 Numerical Scheme

A sixth-order central difference convective scheme (Eq. A.35) with a monotonic TVD type
limiter is used. Sixth–order interpolation is used for the velocity in the convective terms e.g. for \( v \) in \( \nu \frac{\partial u}{\partial y} \) etc. A sixth-order central difference scheme is used for the diffusive terms. A third-order accurate explicit time integration scheme (Eq. A.41) is used to integrate the convective and diffusive terms in time. The pressure gradient and the continuity equation are represented by second-order centered schemes. The equation for the transport of the passive scalar is represented using the same order of schemes as for the momentum equations. Sixth order interpolation is used to interpolate the velocity from the staggered grid locations to the main grid points that the scalar is defined. Three subiterations are used in the colored SCGS method, with relaxation factors of \( \alpha_p = \alpha_u = \alpha_v = \alpha_w = .95 \). The simulations are run in single precision. In order to determine when sufficient time has passed so that the flow can be considered fully developed, the residual history and the integrated averages of the scalar over three planes are monitored as a function of time. Fig. 6.1 shows the residual history of the continuity equations as a function of time. Fig. 6.2 shows the history of the integrated averages of the scalar over the \( xz \) plane for the same cases. As statistics are collected after the flow has traveled approximately 9.3 units of time, it can be seen from these figures that the flow has reached a statistically steady state by that time.

6.4.3 Convergence of Budgets

The results of a numerical simulation of Equations 2.1, 2.2 and 2.3 will of course depend on the numerical scheme used. It will also depend on the grid used. If statistical data, which is averaged over a time period, is desired, then the time period over which the data is averaged will have an effect on the results. These last two issues are addressed in the next two subsections.

6.4.3.1 Convergence of Budgets for Different Time Averaging Periods

In this subsection the effect of the time averaging period is examined on the \( 128 \times 64 \times 64 \) grid. In the following plots, the time averaging period is successively doubled. Once the flow has become fully developed, the time averaging is begun. The time averaging is done using results from every time step at which Eq. 2.1, Eq. 2.2 and Eq. 2.3 are solved. Fig. 6.3(a & d) show
Figure 6.1: Average residual of continuity equation for channel flow

Figure 6.2: Average value of scalar in $xz$ plane for channel flow

the mean value of $u$ and the mean scalar respectively. It is difficult to distinguish between the results from different time averaging periods. The mean value of $v$ is theoretically zero due to symmetries of a channel flow. It can be seen (Fig. 6.3b) to decay to zero as the time averaging period increases. The same is true of the mean value of $w$; see Fig. 6.3c. The peak value of the
turbulent kinetic energy, which occurs near the wall, is a strong function of the time averaging period (see Fig. 6.4a). However, at the two longest time averaging periods, it changes very little. The turbulent fluctuations of the scalar can be seen to converge rather well at the three longest time averaging periods (Fig. 6.5d). The convective transport term in the $k$ equation (Eq. 6.4) is theoretically zero due to symmetries of a channel flow. It can be seen (Fig. 6.5a) to decay to zero as the time averaging period increases. At the three longest time averaging periods there is little difference in its value. The dissipation ($\varepsilon$) is essentially independent of the time averaging period over most of the flow, with the exception of a region close to the wall in which there is a sharp change in its rate of change (see Fig. 6.5b). In this region it converges at the three longest time averaging periods. The residuals of Eq. 6.4 and Eq. 6.5 can be seen in Fig. 6.5(c & d) to behave differently from the individual quantities in Eq. 6.4 and Eq. 6.5 in that the residuals do not exhibit any trend, as the time averaging period is increased. Based on the results of this subsection, it can be seen that the time averaging period needs to be at least 80000 time steps for this flow. This corresponds to the flow traveling $720h$ at the mean velocity, considering the time step used. This requirement is met in all the results presented in the present work for the budgets of Eq. 6.4 and Eq. 6.5.

### 6.4.3.2 Convergence of Budgets on Different Grids

In this subsection the effect of the grid on the statistics is examined for the grids in Table 6.1. The mean velocities and scalar are shown in Fig. 6.6. There is no discernible trend in the mean $v$ and $w$ resulting from different grids. The magnitude of $w$ (which is theoretically zero for this flow) is similar to that of (Kim et al., 1987). The trend in $u$ is somewhat strange in that the value computed on the $256 \times 128 \times 128$ grid is much closer to that obtained on the $64 \times 32 \times 32$ grid than to that obtained on the $128 \times 64 \times 64$ grid. The non-normal turbulent Reynolds stresses and turbulent kinetic energy are given in Fig. 6.7. The results of $\overline{ww'}$ on the
Figure 6.3: Mean velocity components and scalar, averaged over different time periods, channel flow, 128 × 64 × 64 grid

Figure 6.4: Turbulent kinetic energy and turbulent scalar fluctuations, averaged over different time periods, channel flow, 128 × 64 × 64 grid

128 × 64 × 64 and 256 × 128 × 128 grids are very close to each other and to the results of (Kim et al., 1987), (Kawamura et al., 1998) and (Debusschere & Rutland, 2002), while the 64 × 32 × 32 grid under predicts this term. The turbulent Reynolds stress components $\overline{u'w'}$ and
which are theoretically zero for this flow) exhibit no discernible trends on the different grids.

The prediction of $k$ is somewhat different in that no clear grid dependent trend emerges, although the $256 \times 128 \times 128$ grid provides results closest to that of (Kim et al., 1987), (Kawamura et al., 1998) and (Debusschere & Rutland, 2002). In Fig. 6.8a, the results of the production term in Eq. 6.4 are very close to that of (Kim et al., 1987) and (Kawamura et al., 1998) on the $128 \times 64 \times 64$ and $256 \times 128 \times 128$ grids, while the $64 \times 32 \times 32$ grid under predicts this term. The dissipation (see Fig. 6.8b) is under predicted on both the $64 \times 32 \times 32$ and $128 \times 64 \times 64$ grids, while the results on the $256 \times 128 \times 128$ grid are close to those of (Kim et al., 1987) and (Kawamura et al., 1998). The results in Fig. 6.8(c & d) of the production and destruction terms in Eq. 6.5 show a difference in the extrema near the wall on the $128 \times 64 \times 64$ and $256 \times 128 \times 128$ grids, although
the difference is much smaller relative to the results on the $64 \times 32 \times 32$. In Fig. 6.9a the turbulent diffusion term in Eq. 6.4 is very close to that of (Kawamura et al., 1998) on the $128 \times 64 \times 64$ and $256 \times 128 \times 128$ grids, while again the $64 \times 32 \times 32$ grid under predicts this term. Figure 6.9b shows the turbulent diffusion term in Eq. 6.5, in which results on the $128 \times 64 \times 64$ and $256 \times 128 \times 128$ grid are much closer to each other than to the $64 \times 32 \times 32$ grid. Figure 6.10 shows the residual of Eq. 6.4, while Fig. 6.11 shows the residual of Eq. 6.5, as computed on the different grids. Note that, as discussed earlier, as Eq. 6.4 and Eq. 6.5 are not derived from the discretized versions of Eq. 2.1 and Eq. 2.2, the residuals are not expected to be zero within machine accuracy. Both residuals can be seen to decrease on the finest grid.

Figure 6.6: Mean velocity components and scalar on different grids, channel flow
Figure 6.7: Non-normal turbulent Reynolds stresses and turbulent kinetic energy on different grids, channel flow
Figure 6.8: Quantities from the exact $k - \varepsilon$ equations on different grids, channel flow

Figure 6.9: Turbulent diffusion on different grids, channel flow
Figure 6.10: Residual of $k$ equation (Eq. 6.4) on different grids for channel flow
Figure 6.11: Residual of ε equation (Eq. 6.5) on different grids for channel flow
6.4.3.3 Effect of Residual

To examine the effect of the residual level (i.e. the residual from the iterative solution of the discretized versions of Eq. 2.2 and Eq. 2.1) four different numbers (1, 3, 30, 300) of subiterations are used in the colored SCGS method on the $128 \times 64 \times 64$ grid. This results in different residual levels as seen in Fig. 6.12. The maximum and average residuals of the discretized version of the continuity equation (Eq. 2.2) decrease by three orders of magnitude as the number of subiterations increases. For the $u$ velocity (Fig. 6.13a), only the results from one subiteration are different from the other numbers of subiterations. The mean value of $u$ is theoretically zero due to symmetries of a channel flow. It can be seen (Fig. 6.13b) to be larger using one and three subiterations in comparison to 30 and 300 subiterations, but to be larger using 300 subiterations versus 30 subiterations. The mean value of $w$ is also theoretically zero. In Fig. 6.14a the results are actually slightly worse as the residual level decreases. The results of the mean scalar (Fig. 6.14b) are indistinguishable for the different numbers of subiterations. The turbulent kinetic energy (Fig. 6.15a) is slightly larger using one subiteration, while the results from using 3, 30 and 300 subiterations are very close to each other. While the values of the turbulent scalar fluctuations (Fig. 6.15b) are somewhat different depending on the number of subiterations, there is no trend in the values. The convective transport term in the $k$ equation (Eq. 6.4) is theoretically zero in this channel flow. As the residual decreases, no trend can be observed for this term in Fig. 6.16a. The production term in Eq. 6.4 also does not exhibit any trend as the residual decreases. Neither the turbulent diffusion term (Fig. 6.17a) in Eq. 6.4 or the residual (Fig. 6.17b) of Eq. 6.4 show any significant changes as the residual decreases. The same lack of a trend concerning these terms in the exact $\varepsilon$ equation (Eq. 6.5) can be seen in Fig. 6.19. As the only significant difference in the results is observed for the residual level resulting from 1 subiteration, it is determined that at least 3 subiterations is sufficient to avoid the residual affecting the results.
6.4.4 Eddy Viscosity and Turbulent Prandtl Number

The eddy viscosity and turbulent Prandtl number are somewhat difficult to compute as they are ratios of quantities which may go to zero. The first simulations of a channel flow in the present work, including heat transfer, involved a different finite difference scheme and grid. The results for the eddy viscosity and turbulent Prandtl number are somewhat different and are described here.

6.4.4.1 First Simulation

The computational domain is $L_x = 6.4h$ units in the streamwise direction ($x$), $2h$ units in the wall normal direction and $L_z = 3.2h$ units in the cross-stream direction ($z$). The grid dimensions
Figure 6.14: Time averaged $w$ (a) and scalar $\phi$ (b) at different residual levels, channel flow, $128 \times 64 \times 64$ grid

Figure 6.15: $k$ (a) and scalar fluctuations (b) at different residual levels, channel flow, $128 \times 64 \times 64$ grid

are $128 \times 128 \times 128$. Evenly spaced grids are used in the $x$ and $z$ directions. A stretched grid is used in the $y$ direction. The first three main grid points in wall units are at $0.3582, 1.0791$ and $1.8356$ from the wall. The maximum grid spacing in wall units in $y$ is $5.043$ (at the channel center line). A third-order upwind convective scheme (Eq. A.25) is used along with fourth-order interpolation of velocity e.g. for $v$ in $v \frac{∂u}{∂y}$ etc. A fourth-order scheme is used for the diffusive terms. The pressure gradient and the continuity equation are represented by second-order centered schemes. The time integration is fully explicit second-order Adams-Bashforth. A scalar equation is solved using the same differencing scheme as the momentum equations. Fourth-order
interpolation is used to interpolate the velocity from the staggered grid locations to the main grid points at which the scalar is defined. The Prandtl number is .7. The time step is .00015. This results in a CFL number of approximately .062, based on the maximum $x$ velocity and grid spacing in the $x$ direction. The SCGS-PP method is used with the following relaxation parameters; $\alpha_u = \alpha_v = \alpha_w = .95$ and $\alpha_p = .999975$.

Comparisons with the DNS data of (Debusschere & Rutland, 2002) are shown in Fig. 6.20, Fig. 6.21 and Fig. 6.22. The top row of Fig. 6.22 compares the eddy viscosity i.e. $\frac{1}{\text{Re}_\tau}$ and turbulent Prandtl number obtained from Eq. 6.8 and Eq. 6.10 with that obtained by only considering
the stresses and gradients which are non-zero as a result of symmetries for this flow. This simplification results in \( \frac{1}{Re_T} = -\frac{u_0 u_2}{\partial x_2} \) and \( Pr_T = \frac{1}{Re_T} \frac{\partial \sigma}{\partial x_2} - \frac{1}{u_2'}. \) This simplification is made in (Debusschere & Rutland, 2002). As it is desired to compute the eddy viscosity and turbulent Prandtl number for flows in which there are no non-zero stresses and gradients, the method cannot make these simplifications. The reason for the large discrepancy in the two methods is due to the error involved in the time averaging process. This error is the result of a finite time averaging period. As a result of this error, the stresses and gradients which theoretically are zero as a result of symmetries are not exactly zero. It should be noted that at the channel center line \(-\overline{u_1' u_2'}\) and
\frac{\partial u_1}{\partial x_2} are theoretically zero, which makes it difficult to converge the eddy viscosity near the channel center line. In the region near the channel center line the errors in the terms which are zero as a result of symmetries are of the same order of magnitude as \(-u_1'u_2'\) and \(\frac{\partial u_1'}{\partial x_2}\). The result is ill defined behavior for \(\frac{1}{Re_x}\) and the Prandtl number which depends on it, when they are computed from Eq. 6.8 and Eq. 6.10.

**6.4.4.2 Last Simulation**

It can be seen in Figures 6.23, 6.24 and 6.26 that there is little difference in the results of the eddy viscosity obtained from Eq. 6.8 and the simplified one. This is also true for the turbulent Prandtl number as can be seen in Figures 6.27, 6.28 and 6.29. The reason for this has not been investigated thoroughly. In the author’s opinion the most likely reason is that the grid used for the
first simulation contained cells with a much higher aspect ratio. The convergence of the grids can be seen in regions away from the channel center line. As mentioned in the previous section, convergence is more difficult to achieve near the channel center line. The large over prediction of the eddy viscosity in the standard $k - \varepsilon$ model is shown in Figures 6.25 and 6.26. Figure 6.29 shows the near wall behavior of the turbulent Prandtl number as compared to the curve fit of (Kays & Crawford, 1993). The curve fit of (Kays & Crawford, 1993) over predicts the turbulent Prandtl number near the wall in comparison to the DNS result in the present work and the of (Debusschere & Rutland, 2002). Note that the curve fit of (Kays & Crawford, 1993) requires an eddy viscosity; the eddy viscosity used for Fig. 6.29 is obtained from Eq. 6.8 instead of using $C_\mu \frac{k^2}{\varepsilon} Re$. 

Figure 6.21: $s's', s'w', s'w'$, channel flow
Figure 6.22: Eddy viscosity and turbulent Prandtl number, channel flow
Figure 6.23: Eddy viscosity from Eq. 6.8, channel flow

Figure 6.24: Eddy viscosity using only $-\overline{u_1' u_2'}$ and $\frac{\partial u_1}{\partial x_2}$, channel flow
Figure 6.25: Eddy viscosity computed from Eq. 6.8 and modeled eddy viscosity, channel flow

Figure 6.26: Eddy viscosity, wall units, channel flow
Figure 6.27: Turbulent Prandtl number from Eq. 6.10, channel flow

Figure 6.28: Simplified turbulent Prandtl number, channel flow
Figure 6.29: Turbulent Prandtl number, wall units, channel flow

\[ \text{Turbulent Prandtl number} \]
\[ \text{Turbulent Prandtl number simplified} \]
\[ \text{Debusschere & Rutland DNS} \]
\[ \text{Kays and Crawford} \]

256x129x128 grid, \( N_{\text{statistics}} = 471559 \)
6.4.5 Exact \( k \) Equation

Fig. 6.30 shows the various terms in the exact \( k \) equation (Eq. 6.4) for the finest grid. It can be seen that for this flow the production and dissipation terms dominate. The term involving fluctuations in pressure is the smallest term, aside from the convective transport term which is theoretically zero for this flow. The residual term is the next largest and is relatively small compared to other terms.

6.4.6 Exact \( \varepsilon \) Equation

Fig. 6.31 shows the various terms in exact \( \varepsilon \) equation (Eq. 6.5) for the finest grid. It can be seen that for this flow there are three dominant terms. The term involving fluctuations in pressure is the smallest term, aside from the convective transport term which is theoretically zero for this flow. The residual term is the next largest and is relatively small compared to other terms.
6.4.7 Modeled $k$ Equation

A comparison of the DNS computed production term ($-u_i u_j \partial u_i / \partial x_j$) with two models, $2C_{\mu} \left[ \frac{k^2}{\varepsilon} S_{i,j} \right] Re$ and $2Re^{-1} S_{i,j} S_{i,j}$, is given in Fig. 6.32. Note that the only difference between the two models in Fig. 6.32 is the eddy viscosity, which is defined to be $C_{\mu} \frac{k^2}{\varepsilon} Re$ (standard $k-\varepsilon$ model) or $Re^{-1}$ (as obtained from Eq. 6.8). It can be seen that using the eddy viscosity computed from Eq. 6.8 results in almost identically the same value as the DNS computed production term ($-u_i u_j \partial u_i / \partial x_j$). In comparison, the use of $C_{\mu} \frac{k^2}{\varepsilon} Re$ for the eddy viscosity, results in a profile with the right shape but which drastically overshoots, particularly near the wall. The differences between the models and the DNS computed production term ($-u_i u_j \partial u_i / \partial x_j$) are strictly the result of the differences in representing the Reynolds stresses, as they are the only part modeled. Note that the eddy viscosity computed from Eq. 6.8 is positive throughout the flow; as a result the
modeled term $2Re^{-1}_\tau S_{i,j}S_{i,j}$ is positive throughout the flow, as is the DNS computed production term $(-u_i'u_j'\partial U_i/\partial x_j)$. The fact that the production term $(-u_i'u_j'\partial U_i/\partial x_j)$ is positive throughout the flow, is a result of the specific flow and may not be the case for other flows. Note that the model $2C_\mu k^2 S_{i,j}S_{i,j} Re$ is always positive, whereas the model $2Re^{-1}_\tau S_{i,j}S_{i,j}$ may be negative if the eddy viscosity is.

The turbulent diffusion terms $(\partial/\partial x_j) [-\frac{1}{2} u_i'u_i' - p' u_j']$ in the exact $k$ equation are lumped together and modeled as a single diffusion process in the standard $k-\varepsilon$ model. Note that for this flow, the term involving fluctuations of pressure is small (see Fig. 6.30). Figure 6.33 compares the DNS computed value of turbulent diffusion $(\partial/\partial x_j) [-\frac{1}{2} u_i'u_i' - p' u_j']$, with two models $\partial/\partial x_j \left[ \frac{C_\mu k^2}{\sigma_h} \frac{\partial k}{\partial x_j} Re \right]$ and $\partial/\partial x_j \left[ Re^{-1}_\tau \frac{\partial k}{\partial x_j} \right]$, the first of which uses $C_\mu k^2 Re$, the second $Re^{-1}_\tau$ (obtained from Eq. 6.8), as the eddy viscosity. The model using an eddy viscosity of $C_\mu k^2 Re$
under and over-shoots the DNS computed value by more than an order of magnitude. If $Re_{-1}$ is used in place of $C_{\mu} \frac{k^2}{\varepsilon} Re$, the result is much better, although this model slightly exaggerates the extrema in the profile. These results demonstrate that the problem with the modeling of this term is with the choice of the eddy viscosity. Therefore this term can be modeled well as a diffusion process, if the eddy viscosity is chosen appropriately. The residual of the modeled $k$ equation (Fig. 6.34) illustrates the large errors in the standard $k - \varepsilon$ model. These errors are particularly large near the wall.

6.4.8 Modeled $\varepsilon$ Equation

Figure 6.35 compares the production term of the exact $\varepsilon$ equation (Eq. 6.5) with two models, $2C_{\varepsilon 1} C_{\mu} k S_{i,j} S_{i,j} Re$ and $2C_{\varepsilon 1} \varepsilon k^{-1} Re_{-1}^{-1} S_{i,j} S_{i,j}$, used to represent it. As previously, the only difference between these two models is that, for the eddy viscosity, the first uses $C_{\mu} \frac{k^2}{\varepsilon} Re$ while
the second uses $Re_{r}^{-1}$ (as obtained from Eq. 6.8). It can be seen that the model using $C_{\mu}k^{2}Re$ as the eddy viscosity greatly over predicts the production term. The model using $Re_{r}^{-1}$ (obtained from Eq. 6.8) results in values much closer to that computed from the exact $\varepsilon$ equation, although it slightly under predicts the profile near the wall. Note that both of the models predict a zero value for production on the wall. However, for this flow, the computed production is also zero at the wall. For the standard $k - \varepsilon$ model ($2C_{\varepsilon 1}C_{\mu}kS_{i,j}S_{i,j}Re$) this zero value on the wall is a result of $k$ being zero on the wall. For the model using $Re_{r}^{-1}$ for the eddy viscosity, it is more complicated due to the term $k^{-1}Re_{r}^{-1}$, which results in $\frac{\theta}{6}$ on the wall. In the present work, this has been defined to be zero, which is validated by the behavior of this term as the wall is approached.

As the destruction term in the exact $\varepsilon$ equation is assumed to be primarily negative, the standard $k - \varepsilon$ model represents this term by an expression ($-C_{\varepsilon 2}\frac{\theta^{2}}{kRe}$) that is always less than zero. Figure 6.36 shows that this term, as computed from the exact $\varepsilon$ equation, is negative throughout the flow.

Figure 6.34: All terms in modeled $k$ equation Eq. 6.6, channel flow
The model predicts the destruction term well, although it predicts slightly smaller values away from the wall at \( y \approx \frac{h}{4} \).

Figure 6.37 compares the turbulent diffusion term of the exact \( \varepsilon \) equation
\[
\left( \frac{\partial}{\partial x_j} \left[ -u'_j u'_{1,m} u'_{1,m} - 2p'_{m} u'_{j,m} \right] \right)
\]
with two models, \( \frac{\partial}{\partial x_j} \left[ \frac{C_{\mu} R e \ k^2}{\varepsilon} \ \frac{\partial \varepsilon}{\partial x_j} \right] \) and \( \frac{\partial}{\partial x_j} \left[ \frac{Re^{-1}_{\tau}}{\varepsilon} \ \frac{\partial \varepsilon}{\partial x_j} \right] \), used to represent it. Again, the only difference between these two models is that, for the eddy viscosity, the first uses \( C_{\mu} \frac{k^2}{\varepsilon} R e \) while the second uses \( Re^{-1}_{\tau} \) (obtained from Eq. 6.8). The model using \( C_{\mu} \frac{k^2}{\varepsilon} R e \) as the eddy viscosity greatly under and over predicts turbulent diffusion, while the model using \( Re^{-1}_{\tau} \) predicts it much better. However, both models qualitatively predict the behavior of this term reasonably well, as both predict the sign changes that occur near the wall, with the exception of the last peak in the profile close to the wall, which both models fail to predict. These results demonstrate that the major problem with the modeling of the production and turbulent diffusion terms lies with the choice of the eddy viscosity. The residual of the modeled \( \varepsilon \) equation (Fig. 6.38) illustrates the large errors in the standard \( k - \varepsilon \) model. These errors are particularly large near the wall.
Figure 6.36: Destruction term from Eq. 6.5 vs modeled term, channel flow
Turbulent diffusion \( \varepsilon \) vs modeled term, channel flow

Figure 6.37: Turbulent diffusion term from Eq. 6.5 vs modeled term, channel flow

All terms in modeled \( \varepsilon \) equation Eq. 6.7, channel flow

Figure 6.38: All terms in modeled \( \varepsilon \) equation Eq. 6.7, channel flow
6.5 Budgets of Film Cooling Jets

The efficiency of turbine engines improves as the operating temperature of the working fluid in the turbine increases. Higher fluid temperatures in turbines leads to blade failure resulting from allowing material limits to be exceeded. A common means of cooling turbine blades is film cooling, in which the goal is to protect the blade surface from the hot crossflow by a film of cooler fluid injected through holes in the blade surface. These film cooling holes must be designed in such a way that the coolant jet covers and remains near the blade surface and does not penetrate into the crossflow excessively. The film cooling jets consume process air and therefore a design goal is to maximize the cooling and minimize the mass flow through the jets. For these reasons it is important to have accurate models for the flow and associated heat transfer when designing turbine blades.

As turbine engines consist of a number of stages, each of which contains a row of turbine blades, the incoming flow for the later stages is highly turbulent, due to the wake effect of upstream blades and stators. This wake effect is the result of large scale structures, which are shed from the trailing edge of the blades and stators, impinging on downstream blades. This has a significant impact on the film cooling effectiveness (Teng et al., 2000). In addition, large scale structures also result from the interaction of the film cooling jet with the crossflow. These large scale structures make it very difficult to predict the flow and associated heat transfer using the Reynolds Averaged Navier-Stokes equations. The most commonly used turbulence model (the standard $k - \varepsilon$ model) is well known to have difficulty in film cooling applications. This model typically over predicts the distance the jet penetrates into the crossflow and under predicts the jet spreading rate (Hoda & Acharya, 2000). It is well known (Wilcox, 1993), to have difficulties in predicting flows with adverse pressure gradients as occurs in the recirculation region behind a film cooling jet. Despite these difficulties, the standard $k - \varepsilon$ model has been applied to film cooling
flows in (He et al., 1995), (Ajersch et al., 1995) and (Hoda & Acharya, 2000). (He et al., 1995) reported difficulties in predicting film cooling effectiveness at higher blowing ratios, which they attributed to the failure of the turbulence model to correctly predict jet penetration. In a combined experimental and computational work, (Ajersch et al., 1995) found an over prediction of the streamwise and vertical velocity. They also reported significantly larger extrema for a Reynolds turbulent shear stress. An over prediction of the turbulent kinetic energy was seen in (Hoda & Acharya, 2000). (Berhe & Patankar, 1996) found an under prediction of lateral film cooling effectiveness (in comparison to experimental data) which they attributed to the under prediction of the lateral spreading of the jet. (Berhe & Patankar, 1997) found a greater jet penetration and a correspondingly reduced lateral spreading of the jet. In a computational and experimental study (Harrington et al., 2001) found a significant over prediction of jet penetration and a significant under prediction of jet lateral spreading. Using the standard $k-\varepsilon$ model (Radomsky & Thole, 2000) found a large over prediction of the turbulent kinetic energy and one of the Reynolds turbulent shear stresses for flow in a turbine vane passage. In a series of papers on film cooling flows (Medic & Durbin, 2002b) (Medic & Durbin, 2002a) obtained significant improvements in prediction of turbulent kinetic energy and heat transfer coefficients by the use of limiters in the standard $k-\varepsilon$ model.

There are very few studies in the literature concerning DNS or LES of film cooling flows. An unsteady simulation of a circular jet in crossflow was performed by (Hahn & Choi, 1997) on a $705 \times 161 \times 129$ grid using second-order central difference schemes. The Reynolds number was 1750 and the blowing ratio was 0.5. The delivery tube was not modeled in their work. They solved the incompressible Navier-Stokes equations but not the energy equation. Higher-order finite difference schemes were used in an unsteady simulation (Muldoon & Acharya, 1999) to study a normally injected film cooling jet which included the flow development in the delivery
They presented the unsteady interactions of the upstream crossflow and horseshoe vortex system with flow development in the delivery tube, and showed that this unsteady interaction resulted in a periodic pulsing of the jet at a Strouhal number of .44. An LES of a film cooling jet was done in a series of papers, (Tyagi & Acharya, 1999c), (Tyagi & Acharya, 1999b) and (Tyagi & Acharya, 1999a) which investigated the influence of various parameters on the flow physics of square or rectangular coolant jets issuing normally into the crossflow, similar to the experimental setup of (Ajersch et al., 1995). While the reported DNS/LES studies have shown improvements in predictive capabilities vs. Reynolds Averaged Navier-Stokes studies, due to the computational expense of DNS/LES, Reynolds Averaged Navier-Stokes methods are the predominant industrial design tool for film cooling flows. Therefore, it is important to improve the turbulence models used in Reynolds Averaged Navier-Stokes solutions to improve their predictive capabilities for film cooling flows.

The purpose of the present work is to show how the terms in the commonly used standard $k-\varepsilon$ model behave for the basic jet-in-crossflow configuration with jet parameters representative of film cooling. This is done by computing (using DNS) the terms in the exact $k$ and $\varepsilon$ equations (Eq. 6.4 and Eq. 6.5) and the standard $k-\varepsilon$ model (Eq. 6.6 and Eq. 6.7). The modeled terms are then compared with the terms they model in the exact $k$ and $\varepsilon$ equations. This comparison shows what parts of the exact $k$ and $\varepsilon$ equations are significant, along with how well the models for the various terms behave for this type of flow. This study will provide guidance for turbulence modelers to develop turbulence models that are designed for film cooling jets in crossflow. In addition, a new wall damping function suitable for film cooling flows is presented.

**6.5.1 Problem Description**

The modeled flow is similar to that of (Kaszeta, 1998). In their experimental work, an inclined jet with a delivery tube and plenum was examined. As the goal is to obtain DNS data for a representative film cooling geometry, it has been decided to not model the plenum and
delivery tube in the DNS. This decision is a result of the stringent spatial and temporal resolution requirements of DNS. It has been decided that it is better to use the available points in the region of most interest (over the blade surface) where the accurate data is needed for use in improving turbulence models. The inclusion of the plenum and delivery tube will undoubtedly result in changes, but these changes are not expected to result in changes in the important physics of the flow. However, in order to model the effect of the delivery tube and plenum as closely as possible, the boundary condition at the jet exit is specified as the mean velocity obtained from a simulation that includes a plenum and delivery tube. Thus a preliminary simulation calculation is done including the plenum and delivery tube. From this simulation a mean jet exit velocity boundary condition is extracted and used in a second set of simulations in which all grid points are in the jet-crossflow region. All budgets are computed from this second set of simulations.

6.5.2 Simulation with Plenum and Delivery Tube

6.5.2.1 Problem Definition

A schematic of the flow domain along with boundary conditions and dimensions is given in Fig. 6.39. Referring to Fig. 6.39, $L_{up} = 6d$, $L_{down} = 16d$, $L_{freestream} = 5d$, $L_{tube} = 2d$, $L_{plenum} = 10d - L_{tube} \sin(35)$, $L_z = 3d$. The origin of the coordinate system is defined to be the center of the jet at the blade surface. All quantities are non-dimensionalized by the jet diameter $d$ and the maximum inlet crossflow velocity $U_0$. At $y = -L_{plenum}$ the $u$ and $w$ components of velocity are set to zero while $v$ is set to $\frac{1}{(L_{up} + L_{down})(L_z)}$. At the crossflow inlet boundary at $x \approx L_{up}$, the $v$ and $w$ components of velocity are set to zero while the following profile is used for the $u$ velocity.

$$u(y) = \begin{cases} U_0 \left(\frac{y}{2}\right)^{\frac{7}{2}} & 0 \leq y \leq 2 \\ U_0 & y > 2 \end{cases}$$

The Reynolds number based on the average velocity through the jet ($U_{jet}$) and jet diameter is 3068. The average blowing ratio, defined as the ratio $U_{jet}/U_0$ is .24 while the blowing ratio based on the maximum jet exit velocity is .53. The jet is injected at 35 degrees from the crossflow.
Periodic boundaries are used in the $z$ direction. The immersed boundary method (Mohd-Yusof, 1997) is used to model the blade surface, the plenum and the delivery tube. As a result of the immersed boundary method there is a slight mass flow through the blade surface equivalent to $1 - U_{jet}$. A symmetry boundary condition is applied at $y = L_{freestream}$. A convective outflow boundary condition (Eq. 3.20) is applied for all three velocities at $x = L_{down}$.

The grid dimensions are $462 \times 210 \times 102$. The grid is stretched in all three dimensions. This stretching is done to concentrate points near the jet and near the jet wall boundary. In particular the resolution decreases as the outflow boundary is approached. As the sole purpose of this simulation is to provide a mean velocity profile at the jet exit that simulates the effect of the delivery tube and plenum, this resolution is deemed sufficient. The time step used is .003125, which results in maximum CFL numbers of $\sim 1$, $\sim 0.078$ and $\sim 0.055$ in the $x,y$ and $z$ directions respectively.

6.5.2.2 Numerical Scheme

A sixth-order central difference convective scheme (Eq. A.35) with a monotonic TVD type limiter is used. Sixth-order interpolation is used for the velocity in the convective terms e.g. for $v$ in $v \frac{\partial u}{\partial y}$ etc.. A sixth-order central difference scheme is used for the diffusive terms. A third-order accurate explicit time integration scheme (Eq. A.41) is used to integrate the convective and diffusive terms in time. The pressure gradient and the continuity equation are represented by second-order centered schemes. The equation for the transport of the passive scalar is represented using the same order of schemes as for the momentum equations. Sixth order interpolation is used to interpolate the velocity from the staggered grid locations to the main grid points at which the scalar is defined. The accuracy of all finite difference stencils is maintained near non-periodic boundaries by shifting towards the boundary the point at which the stencil is evaluated. Three subiterations are used in the colored SCGS method at each time step. In order to determine when sufficient time has passed so that the flow can be considered fully developed (so that statistics can
be collected) the residual history and the integrated averages of the scalar over three planes are monitored as a function of time.

6.5.2.3 Immersed Boundary Method

The surfaces indicated in Fig. 6.39 are represented using the immersed boundary method of (Mohd-Yusof, 1997). A total of $3.9276 \times 10^5$ points are defined as immersed boundary points at which forcing is added. A second-order interpolation scheme is used. The forcing is defined using the pseudo velocities.

6.5.2.4 Results

Fig. 6.40 shows an instantaneous snapshot of the flow field in the vicinity of the jet exit. It can be seen that the angle of the delivery tube causes a separation region (evident in the negative values for the velocity in the jet inclination direction) to exist on the downstream side of the delivery tube. This causes a jetting effect on the upstream side of the jet exit. The result is a highly non-uniform velocity profile in the jet exit plane. This jetting effect appears also in the mean flow field (Fig. 6.42).
6.5.3 Simulation with Prescribed Jet Exit Condition

6.5.3.1 Problem Definition

A schematic of the flow domain along with boundary conditions and dimensions is given in Fig. 6.41. In Fig. 6.41 $L_{up} = 3.6d$, $L_{down} = 14d$, $L_{freestream} = 4.5d$, $L_z = 3d$. The boundary conditions in the region above the jet exit are identical to those used in the simulation that include the plenum and delivery tube. At the jet exit, the time averaged results obtained from the simulation including the plenum and delivery tube is used as a boundary condition for the velocity. This time averaged velocity should be symmetric about the $z$-axis. However, due to the finite time averaging period, the profile is not exactly symmetric. Symmetry is therefore enforced by averaging the profile around the $z$-axis. Figure 6.42 shows the contours of the three velocity components along with the velocity magnitude as interpolated onto the $574 \times 161 \times 142$ grid at the jet exit.
Figure 6.41: Inclined jet in crossflow with prescribed exit conditions 3D schematic, film cooling jet

Figure 6.42: Jet exit boundary condition as interpolated onto 574 × 161 × 142 grid, film cooling jet
Two grids, with dimensions of $286 \times 80 \times 70$ and $574 \times 161 \times 142$ are used. The $574 \times 161 \times 142$ grid is obtained by doubling the number of points in each dimension of the $286 \times 80 \times 70$ grid, while keeping the grid stretching parameters the same. The time steps used on the grids are .008 and .004 respectively. This results in maximum CFL numbers of $\sim .2$ in the $x$ direction, $\sim .1$ in the $y$ direction and $\sim .08$ in the $z$ direction. The results shown in the present work are obtained on the $574 \times 161 \times 142$ grid, with the $286 \times 80 \times 70$ grid being used for understanding the effect of deceasing the grid spacing.

6.5.3.2 Energy Spectra

A one-dimensional spatial energy spectrum in the $x$ direction of the $x$ component of velocity is given in Fig. 6.43. This spectra is obtained by interpolating the velocity on the uneven spatial grid onto an even grid which is defined by the smallest grid spacing in the uneven grid and then transforming into discrete wave number space. Regions in which the slope of the spectra matches the $5/3$ slope predicted by turbulence theory (Kolmogoroff, 1941) can be seen. Close agreement can be seen between the results obtained on the $286 \times 80 \times 70$ grid and on the $574 \times 161 \times 142$ grid. The good agreement in the results from the two grids may be due to the use of a monotonic convection scheme which adds viscosity in regions where the gradients are large in order to prevent overshoots and undershoots. As a result, on an unresolved grid, the scheme behaves as a Large Eddy Simulation model in which the model for the unresolved scales is the added viscosity. This effect has been termed Monotonic-Integrated Large-Eddy Simulation (MILES) by (Oran & Boris, 2001). However, it can be seen that energy pile-up occurs on the $286 \times 80 \times 70$ grid to a greater degree than on the $574 \times 161 \times 142$ grid. Fig. 6.44 shows one-dimensional spatial energy spectrums in the $z$ direction of the $x$ and $y$ components of velocity.

6.5.4 Flow Physics

Jets in crossflow contain a number of distinct flow structures (Fric & Roshko, 1994). These structures are a Counter Rotating Vortex Pair (CVP) of kidney shaped vortices, a horseshoe vortex,
shear layer vortices and upright wake vortices. As the jet is deflected by the crossflow it splits into the CVP. While it is difficult to distinguish the CVP in an instantaneous image of the flow field (Figures 6.45 and 6.47), it can be clearly seen if the time averaged flow field is examined, as
discussed later (Figures 6.48 and 6.52). The CVP entrains fluid from the crossflow into the jet, promoting mixing of the jet with the crossflow. As a result of vortex induction, the CVP causes the jet to lift away from the surface, which is undesirable for film cooling. The horseshoe vortex forms upstream of the jet, wraps around the jet, and reorients itself in the streamwise direction (Fig. 6.45). This vortex is actually a number of vortices (see Fig. 6.52) and remains close to the blade surface. Its role is to entrain coolant fluid from the jet and to cool the blade surface along its trajectory as seen in Fig. 6.45. Shear layer vortices form at the boundary between the jet and the crossflow (Fig. 6.45). These vortices begin small but grow into large scale structures by $3d - 4d$ downstream of the jet center. Fig. 6.46 clearly shows the regions where scalar transfer is highest. This occurs at the interface between the jet and crossflow and is associated with the shear layer vortices. Behind the jet a region of low pressure and velocity is formed. In this region a recirculation pattern emerges that entrains fluid into the jet. This region is evident in contours of the magnitude of velocity in Fig. 6.47. Wake vortices are not seen in this flow, likely as a result of the low blowing ratio and small jet inclination angle (which results in a jet close to the blade surface).
6.5.5 Comparison with Experimental Data

In this subsection, data from the present work is compared to that obtained experimentally by (Kaszeta, 1998). While the Reynolds number and jet inclination angle are the same in the present work as in (Kaszeta, 1998), there are three other differences. First, (Kaszeta, 1998) had a turbulent intensity of \(~12\%\) in the crossflow, while the present work has none. Second, the
domain in the present work does not include a delivery tube and plenum while the experiments of (Kaszeta, 1998) did. In the present work the boundary condition at the jet exit is the mean velocity obtained from a simulation that includes a plenum and delivery tube; this boundary condition does not change with time. Third, the blowing ratios are different; 1 in (Kaszeta, 1998) and .24 in the present work. These differences should be considered when comparisons are made between the present work and the results of (Kaszeta, 1998).

Fig. 6.48 compares the time averaged $u$. The time averaged $u$ in the present work appears to have slightly lower values than found by (Kaszeta, 1998). The horseshoe vortex, which is evident in the present work, cannot be seen in (Kaszeta, 1998), although this may be because of the small number of contour values in his plots. The shape of the contours of $\overline{u'v'}$ in the present work (see Fig. 6.49) matches that of (Kaszeta, 1998), with the exception again of the horseshoe vortex.

The shape of the contours of $\overline{u'w'}$ (see Fig. 6.50) is much more different. While the contours of $\overline{u'w'}$ in the present work are very symmetric about the $z$-axis, this is not so in (Kaszeta, 1998). However, the general shape of both plots is the same in that $\overline{u'w'}$ is negative on the positive $z$-axis and positive on the negative $z$-axis.
Figure 6.49: Comparison of $u'v'$ in $yz$ plane at $x = 5d$, film cooling jet

Figure 6.50: Comparison of $u'w'$ in $yz$ plane at $x = 5d$, film cooling jet
6.5.6 First and Second Order Statistics

The mean scalar is shown in various planes in Fig. 6.51c. Note that the maximum and minimum shown on figures in this work apply to the surfaces shown in the figures and not the entire field. The signature of the horseshoe vortex on the blade surface shows up distinctly in this figure. The horseshoe vortex entrains a small amount of fluid from the leading edge of the jet, and transports it downstream as the vortex wraps around the jet and orients itself in the streamwise direction. The effect of the horseshoe vortex in keeping the velocity magnitude small near the blade surface is evident in Fig. 6.51b, which shows a crossflow region between the horseshoe vortex and the CVP near the blade surface in which the velocity magnitude is large. A stagnation region in front of the jet, where the horseshoe vortex forms, and a low pressure region behind the jet can be seen in the mean pressure field in Fig. 6.51a. A region behind the jet at \( x \approx 2d \), in which the value of the scalar on the blade surface is higher than further downstream, can be seen in Fig. 6.51c. This likely is a result of the recirculation pattern in this region in which crossflow fluid is entrained into the jet wake. The CVP and the horseshoe vortex can be seen in Fig. 6.52, in which for clarity every second vector in each direction and half of the domain in the \( z \) direction is shown. The horseshoe vortex consists of two vortices with opposite rotation, while the CVP is seen to form two vortices with the same sense of rotation.

The normal Reynolds stresses are shown in Fig. 6.53. The \( u'u'\) component is larger than the other components and has its highest values in the shear layer where large velocity gradients exist. The \( u'u'\) component also dominates the horseshoe vortex, growing significantly in strength as the horseshoe vortex convects in the downstream direction. It starts to form a peak near the wall at \( x \approx 10d \). The smallest component is \( v'v'\), which spreads towards the wall as the downstream distance increases. A spike in the \( w'w'\) component exists in the recirculation region behind the jet at \( x \approx 2d \). This is where the crossflow collides in the \( z \) direction after wrapping around the jet.
Figure 6.51: Three-dimensional view of time averaged pressure, velocity magnitude and scalar, film cooling jet

The shear stress components of the Reynolds stress tensor are shown in Fig. 6.54. The strong anisotropy of the flow field can be seen in this figure. In particular the $\bar{v}w$ component experiences a sign change (which appears clearly in the $yz$ plane at $x \approx 5d$ in Fig. 6.54c), that occurs near the boundary between the jet and the crossflow.

Figure 6.55 shows that the scalar fluctuations are highest in the shear layer and reach a maximum at $x \approx 5d$. The scalar fluctuations are small in the horseshoe vortex. The turbulent kinetic energy has a different behavior, reaching a maximum further downstream at roughly $x \approx 7.5d$. It is significant in the horseshoe vortex. It can be seen that turbulent kinetic energy in the wake region is small except for a peak at $x \approx 2d$, which is due to the contribution of $\bar{w}w$. 

173
Figure 6.52: Time averaged velocity vectors and contours of scalar field in $yz$ plane at $x \approx 8d$, film cooling jet

Figure 6.53: Three-dimensional view of $u'u'$, $v'v'$, $w'w'$, film cooling jet
Figure 6.54: Three-dimensional view of $u'u'$, $u'w'$, $v'w'$, film cooling jet
Figure 6.55: Three-dimensional view of $p'$, $k'$, $s'$, film cooling jet
### 6.5.7 Eddy Viscosity and Turbulent Prandtl Number

The over prediction of the eddy viscosity by the model \( C_\mu \frac{k^2}{\varepsilon} \) can be seen quite clearly in Fig. 6.56. The eddy viscosity obtained from DNS (using Eq. 6.8) is actually negative in certain regions, which is not permitted by the model \( C_\mu \frac{k^2}{\varepsilon} \), in which the eddy viscosity is always positive. The eddy viscosity obtained from Eq. 6.8 enables some terms (as discussed later) to be modeled accurately, a factor which perhaps should be considered in the design of turbulence models, which traditionally assume that the eddy viscosity is positive. It should be noted that some LES models allow for a negative eddy viscosity which is intended to allow small scales to feed back energy to larger scales, a process known as backscatter. The turbulent Prandtl number is ill-behaved in many parts of the flow (Fig. 6.57); however, it is approximately .9 throughout the jet, although it is much smaller near the wall and in the freestream. The reason for this ill behavior can be seen in Fig. 6.58a which shows the numerator and denominator in Eq. 6.10. The numerator is well-behaved but near the wall the denominator fluctuates around zero, as can be seen in Fig. 6.58. This occurs as \( s'u' \) and \( \frac{\partial \tau}{\partial x} > 0 \), while \( s'u' < 0 \) and \( \frac{\partial \tau}{\partial y} > 0 \) (see Fig. 6.59a) and as a result, near the wall, the sum \( s'u' \frac{\partial \tau}{\partial x} + s'v' \frac{\partial \tau}{\partial y} \) is close to zero. The author believes that the problem is a result of the Neumann boundary condition for the scalar at \( y = 0 \). This condition results in small values (comparable to \( \frac{\partial \tau}{\partial x} \)) for \( \frac{\partial s}{\partial y} \) near the wall (Fig. 6.59a). Note that there is no difficulty in finding the turbulent Prandtl number near the wall for a channel flow using a Dirichlet boundary condition for the scalar. The eddy viscosity is not the cause of the ill behavior, as can be seen in Fig. 6.59b. In Fig. 6.60 the turbulent Prandtl number in the \( y \) direction (i.e. \( Re_\tau^{-1} \frac{\partial \tau}{s'v'} \frac{\partial \tau}{\partial y} \)) is well-behaved at this location.

### 6.5.8 Convergence of Budgets

Figure 6.61 shows the residual of the exact \( k \) equation (Eq. 6.4) computed on different grids. It can be seen that the residual is less on the finer grid, particularly near the wall. Fig. 6.62 shows the residual of the exact \( \varepsilon \) equation (Eq. 6.5) computed on different grids. In this case the residual
Figure 6.56: Eddy viscosity computed from Eq. 6.8 and modeled eddy viscosity, film cooling jet actually increases slightly in the region at the edge of the jet crossflow boundary on the fine grid relative to the coarse grid, although it decreases near the wall. However, the absolute value of the maximum and minimum residuals over the region plotted decreases on the finer grid. The residual of the $\varepsilon$ equation is highest on and near the wall. If the residual is examined over the entire field (instead of just the planes in Figures 6.61 and 6.62) the maximum of the absolute value of the residual of the exact $k$ equation is (0.2022, 0.0735) and of the exact $\varepsilon$ equation (17970, 13746) on the $286 \times 80 \times 70$ and $574 \times 161 \times 142$ grids respectively. Note that, as discussed earlier, as Eq. 6.4 and Eq. 6.5 are not derived from discretized versions of Eq. 2.1 and Eq. 2.2, the residuals

Figure 6.57: Turbulent Prandtl number computed from Eq. 6.10, film cooling jet
are not expected to be zero within machine accuracy. Figures 6.63 and 6.64 show the effect of the number of time steps (that the statistics are averaged over) on the production term of the exact \( k \) and \( \varepsilon \) equations respectively. The maximum change in the production term for \( \varepsilon \) changes by 3.9\%, while that of the exact \( k \) equation changes by 1.3\% as the number of time steps is more than doubled. This small change indicates that the temporal averaging period is sufficient. An issue concerning the discretization of the exact \( k \) and \( \varepsilon \) equations on inflow boundaries is described in Appendix D.

### 6.5.9 Exact \( k \) Equation

Figure 6.65 shows the various terms in the equation for \( k \) (Eq. 6.4) for the \( 574 \times 161 \times 142 \) grid. For this flow it can be seen that the production term is the largest with dissipation the next
largest term. Dissipation appears to be significant in the horseshoe vortex, while production appears to be less significant there. The dissipation term approximates a negative mirror of the production term in Fig. 6.67, with the exception of the region near the wall where the production
Figure 6.62: Three-dimensional view of residual of exact $\varepsilon$ equation (Eq. 6.5) on two different grids, film cooling jet

Figure 6.63: Three-dimensional view of production term in exact $k$ equation (Eq. 6.4) for two different time averaging periods, film cooling jet
Figure 6.64: Three-dimensional view of production term in exact $\varepsilon$ equation (Eq. 6.5) for two different time averaging periods, film cooling jet

is small while dissipation is large. At the wall dissipation is balanced by the diffusion terms

\[
\frac{\partial}{\partial x_2} \left[ \frac{1}{\Re} \frac{\partial k}{\partial x_2} - \frac{1}{2} u'_{i} u'_{j} \right],
\]

which are the only non-zero terms at the wall. Convective transport of $k$ is significant in the CVP, unlike a developed channel flow for which it theoretically is zero. Molecular diffusion of $k$ is insignificant over most of the flow, and except near the wall, is two orders of magnitude smaller than turbulent diffusion. Turbulent diffusion is highest along the outer side of the CVP and is small in the horseshoe vortex. The residual is the smallest term and is concentrated near the upper part of the jet. Figure 6.66 shows individually two terms, the sum of which is turbulent diffusion. These terms are of comparable magnitude, while for a channel flow the term involving fluctuations in pressure is much smaller. This suggests that models that have been tuned and work well for a channel flow will experience difficulties for a film cooling flow. A rapid change in

\[
\frac{\partial}{\partial x_2} \left[ -\frac{1}{2} u'_{i} u'_{j} \right]
\]

(in which the sign changes) can be seen at the top of the jet. This change in sign will be predicted by the gradient of $k$ (see Fig. 6.55), which is used when this term, along with

\[
\frac{\partial}{\partial x_2} \left[ -\bar{p}' u'_{j} \right]
\]

is modeled as proportional to the diffusion of $k$. However, this same change in sign does not occur in

\[
\frac{\partial}{\partial x_2} \left[ -\bar{p}' u'_{j} \right],
\]

which suggests that this part of the term will
Figure 6.65: All terms of exact $k$ equation (Eq. 6.4), $yz$ plane at $\approx 5d$, film cooling jet

not be modeled well by being made proportional to the diffusion of $k$.

### 6.5.10 Exact $\varepsilon$ Equation

Figure 6.68 shows the various terms in the exact $\varepsilon$ equation for (Eq. 6.5) for the $574 \times 161 \times 142$ grid. The destruction term is the most significant as it is extremely large near the wall. Production is the next significant term, and is large near the wall and at the interface between the jet and the crossflow. Convective transport reaches a maximum in the horseshoe vortex and a minimum near the wall in between the CVP and the horseshoe vortex. Molecular diffusion is insignificant over most of the flow, and except near the wall, is two orders of magnitude smaller than turbulent diffusion. Turbulent diffusion is large along the outer side of the CVP (where it changes sign) and is also significant within the horseshoe vortex. The residual is large in magnitude near the wall and near the wall has the same distribution as the destruction term.

Both parts of the turbulent diffusion of $\varepsilon$ term are shown in Fig. 6.69. The term involving
Figure 6.66: Three-dimensional view of turbulent diffusion terms in exact $k$ equation (Eq. 6.4), film cooling jet

Figure 6.67: Three-dimensional view of production and dissipation terms in exact $k$ equation (Eq. 6.4), film cooling jet
pressure has extrema nearly two orders of magnitude greater than the other term; however, these extrema are concentrated near the front edge of the jet on the wall (discussed later). In the rest of the flow, the terms are similar, except that the term \( \frac{\partial}{\partial x_j} [ -u_j u_i,m u_i,m ] \) changes sign twice in the vertical direction, while \( \frac{\partial}{\partial x_j} [ -2p_{,m} u_{j,m} ] \) changes sign once. This is unlike a channel flow for which the term involving fluctuations in pressure is much smaller. Note that this difference between the term involving pressure and the term which does not is also seen in the exact \( k \) equation.

In Fig. 6.70a the destruction term is positive in a large region in the jet away from the wall. This region correlates well with the distribution of \( k \) (see Fig. 6.55b). Near the wall the destruction term takes on large negative values. The production term (see Fig. 6.70b) is large in the recirculation region and is negative at the wall immediately behind the jet. The extrema in production are concentrated near the front edge of the jet on the wall and are discussed later.
Figure 6.69: Three-dimensional view of turbulent diffusion terms in exact $\varepsilon$ equation (Eq. 6.5), film cooling jet

Figure 6.70: Three-dimensional view of production and destruction terms in exact $\varepsilon$ equation (Eq. 6.5), film cooling jet
6.5.11 Modeled $k$ Equation

A comparison of the DNS computed production term $(-u'_i u'_j \frac{\partial u'_i}{\partial x_j})$ with two models, $2C_\mu \frac{k^2}{\varepsilon} S_{i,j} S_{i,j} Re$ and $2Re_\tau^{-1} S_{i,j} S_{i,j}$, is given in Fig. 6.71. Note that the only difference between the two models in Fig. 6.71 is the eddy viscosity, which is defined to be $C_\mu \frac{k^2}{\varepsilon} Re$ (standard $k - \varepsilon$ model) or $Re_\tau^{-1}$ (as obtained from Eq. 6.8). As Eq. 6.8 finds an eddy viscosity such that it minimizes the error in the approximation of the Reynolds stresses using Eq. 6.3, it should provide a better approximation of the production term than the standard $k - \varepsilon$ model, which uses $C_\mu \frac{k^2}{\varepsilon} Re$ to represent the eddy viscosity. As the errors in the modeled production term are strictly the result of errors in representing the Reynolds stresses, this provides a good measure of the ability of the minimization procedure (Eq. 6.8) in combination with the Boussinesq approximation, to represent the Reynolds stresses. It can be seen that using the eddy viscosity computed from Eq. 6.8 results in almost identically the same value as the DNS computed production term $(-u'_i u'_j \frac{\partial u'_i}{\partial x_j})$. In comparison, the use of $C_\mu \frac{k^2}{\varepsilon} Re$ for the eddy viscosity, results in an order of magnitude over prediction of the production term, particularly near the blade surface and in the jet region. However, the qualitative behavior of the production term is predicted using $C_\mu \frac{k^2}{\varepsilon} Re$. Note that the eddy viscosity computed from Eq. 6.8 (see Fig. 6.56) is negative in certain regions, which allows the modeled term $2Re_\tau^{-1} S_{i,j} S_{i,j}$ to be negative in certain regions, as is the DNS computed production term $(-u'_i u'_j \frac{\partial u'_i}{\partial x_j})$.

The turbulent diffusion terms $\left(\frac{\partial}{\partial x_j} \left[-\frac{1}{2} u'_i u'_j - p' u'_j\right]\right)$ in the exact $k$ equation are lumped together and modeled as a single diffusion process in the standard $k - \varepsilon$ model. Figure 6.72 compares the DNS computed value of turbulent diffusion $\left(\frac{\partial}{\partial x_j} \left[-\frac{1}{2} u'_i u'_j - p' u'_j\right]\right)$, with two models $\frac{\partial}{\partial x_j} \left[\frac{C_\mu k^2}{\sigma_k} \frac{\partial k}{\partial x_j} Re\right]$ and $\frac{\partial}{\partial x_j} \left[Re_\tau^{-1} \frac{\partial k}{\partial x_j}\right]$, the first of which uses $C_\mu \frac{k^2}{\varepsilon} Re$, the second $Re_\tau^{-1}$ (as obtained from Eq. 6.8), as the eddy viscosity. The model using an eddy viscosity of $C_\mu \frac{k^2}{\varepsilon} Re$, under-shoots the DNS computed value by an order of magnitude. If $Re_\tau^{-1}$ is used in place of
Figure 6.71: Production term from exact $k$ equation (Eq. 6.4) vs models, film cooling jet

$C_\mu \frac{k^2}{\varepsilon} \text{Re}$, the result is much better. These results demonstrate that the problem with the modeling of this term is with the choice of the eddy viscosity expression (at least for this flow). Therefore this term can be modeled well as a diffusion process, if the eddy viscosity is chosen appropriately.

The residual of the modeled $k$ equation (Fig. 6.73) illustrates the large errors in the standard $k - \varepsilon$ model. These errors are particularly large on the wall.

6.5.12 Modeled $\varepsilon$ Equation

Figure 6.74 compares the production term of the exact $\varepsilon$ equation (Eq. 6.5) with two models, $2C_{\varepsilon 1} C_\mu k S_{i,j} S_{i,j} \text{Re}$ and $2C_{\varepsilon 1} \varepsilon k^{-1} \text{Re}^{-1} S_{i,j} S_{i,j}$ used to represent it. As previously, the only difference between these two models is that, for the eddy viscosity, the first uses $C_\mu \frac{k^2}{\varepsilon} \text{Re}$ while the second uses $\text{Re}^{-1}$ (as obtained from Eq. 6.8). In Fig. 6.74, it appears that both models
under predict the production term, based on the maximum and minimum. However, this is only true on the wall at certain locations at the front edge of the jet, where the production term of the exact \( \varepsilon \) equation is very large (see Fig. 6.75a). In the jet region, it can be seen that the model using \( C_{\mu} \frac{k^2}{\varepsilon} Re \) as the eddy viscosity greatly over predicts the production term. The model
using $Re^{-1}_\tau$ (as obtained from Eq. 6.8) results in values much closer to that computed from the exact $\varepsilon$ equation, although it also over predicts the production in the jet region. Note that both of the models predict a zero value for production on the wall, while there is no reason why the actual term in the exact $\varepsilon$ equation will be zero (see Fig. 6.74a). For the standard $k - \varepsilon$ model $(2C_\mu k S_{i,j} S_{i,j} Re)$ this zero value on the wall is a result of $k$ being zero on the wall. For the model using $Re^{-1}_\tau$ for the eddy viscosity, it is more complicated due to the term $k^{-1} Re^{-1}_\tau$, which results in $0/0$ on the wall. In the present work, this has been defined to be zero. The production term is modeled as a positive term in the standard $k - \varepsilon$ model; however, on the wall in the recirculation region behind the jet, it assumes large negative values. Note that negative values are allowed in the model that uses $Re^{-1}_\tau$, as $Re^{-1}_\tau$ may be negative (it is in the part of the recirculation region).

As the destruction term in the exact $\varepsilon$ equation is assumed to be primarily negative, the standard $k - \varepsilon$ model represents this term by an expression $(-C_\varepsilon^2 \varepsilon^2 k Re)$ that is always less than zero. Figure 6.76 shows that this term, as computed from the exact $\varepsilon$ equation, has very large negative values next to and on the wall, but that it is positive (although small) in the jet region away from the wall. The model under predicts the destruction term in the recirculation region behind the jet. The destruction term, as computed from both the exact $\varepsilon$ equation and the model, shows significant negative values within the horseshoe vortex at $x \approx 5d$, although only in the model does this persist downstream. Note that as a result of division by $k$ the model is ill-behaved in parts of the flow, such as the freestream where $k$ approaches zero, near the wall where $k$ also approaches zero and at the wall where $k$ is zero.

Figure 6.77 compares the turbulent diffusion term of the exact $\varepsilon$ equation with two models, $\frac{\partial}{\partial x_j} \left[ C_\mu Re k^2 \frac{\partial \varepsilon}{\varepsilon \partial x_j} \right]$ and $\frac{\partial}{\partial x_j} \left[ \frac{Re^{-1}_\tau \partial \varepsilon}{\varepsilon \partial x_j} \right]$, used to represent it. Again, the only difference between these two models is that, for the eddy viscosity, the first uses $C_\mu k^2 Re$ while the second uses $Re^{-1}_\tau$.
Figure 6.74: Three-dimensional view of production term from exact $\varepsilon$ equation (Eq. 6.5) vs modeled term, film cooling jet

(as obtained from Eq. 6.8). As for the production term of the exact $\varepsilon$ equation, the maximum and minimum values of the turbulent diffusion term from the exact $\varepsilon$ equation occur on the wall at certain locations at the front edge of the jet (see Fig. 6.75b). In the jet region, the model using $C_{\mu} \frac{k^2}{\varepsilon} Re$ as the eddy viscosity greatly under and over predicts turbulent diffusion, while the model using $Re_{\tau}^{-1}$ predicts it much better. This over prediction by the model using $C_{\mu} \frac{k^2}{\varepsilon} Re$ is also seen in the horseshoe vortex system. However, both models qualitatively predict the behavior of this term correctly, as both predict a positive value region at the outer edge of the jet, a negative area inside this and another positive region inside the negative region.

6.5.13 Proposed Wall Damping Function

A serious problem with the standard $k-\varepsilon$ model is that it over predicts the eddy viscosity (see
Figures 6.75, 6.76 and 6.56). This error can be reduced by the use of what is termed a damping function, designed to reduce or damp the eddy viscosity. There are various damping functions in the literature (Rodi & Mansour, 1993). The damping function used in the present work is intended for use as in Eq. 6.11.

$$-\overline{u_i u_j} = f_{\mu}^{dns} C_{\mu} \frac{k^2}{\varepsilon} 2S_{i,j} - \frac{2}{3} k \delta_{i,j}, \quad f_{\mu}^{dns} = \frac{1}{Re_{\varepsilon}} \frac{k^2}{C_{\mu} \varepsilon} \tag{6.11}$$

The damping function in Eq. 6.11 could be a function of many different variables. For practical use, however, it must be a simple expression of variables available in two equation turbulence models. In the present work it has been made a function of a non-dimensional distance
The wall shear velocity is given by:

\[ u_\tau = \sqrt{\frac{\mu}{\rho} \frac{\partial u}{\partial y}} = \sqrt{\frac{\partial u}{\partial y}} \]

Using this, a length scale \( L_\tau \) can be defined by:

\[ L_\tau = \frac{1}{\sqrt{\frac{\partial u}{\partial y}} \frac{1}{\partial y}} = \frac{1}{\sqrt{\frac{1}{\partial y} \frac{1}{\partial y}}} \]

This enables a non-dimensional distance \( (y^+) \), known as wall units, to be defined:

\[ y^+ = \frac{y}{L_\tau} = y \sqrt{\frac{1}{\partial u} \frac{1}{\partial y}} \]

To find a value for \( f(y^+) \) that does not depend on \( x \) and \( z \), \( f(y^+) \) can be averaged in these directions. As the primary interest is in the jet downstream of the injection hole, the averaging
should be restricted to this region. Furthermore, as there is a recirculation region behind the jet, the region is restricted to \( x \geq 4d \), which is past the recirculation region. The spatial averaging is carried out in the region \( 4d \leq x \leq L_{down} \) and \(-1.5d \leq z \leq 1.5d\). Fig. 6.78a shows the spatial average of the eddy viscosity computed from Eq. 6.8 and the eddy viscosity in the standard \( k - \varepsilon \) model \( (C_{\mu} \frac{k^2}{\varepsilon}) \) over the entire distance in the \( y \) direction. The over prediction of the eddy viscosity by \( (C_{\mu} \frac{k^2}{\varepsilon}) \) can be seen quite clearly. The damping function \( f_{\mu}^{\text{dns}} = \frac{1}{C_{\mu} \frac{k^2}{\varepsilon}} \) is plotted on the top axis in Fig. 6.78a. Figure 6.78b shows the spatial average of the eddy viscosity computed from Eq. 6.8 and the eddy viscosity in the standard \( k - \varepsilon \) model \( (C_{\mu} \frac{k^2}{\varepsilon}) \), in a region closer to the wall. A curve fit (Eq. 6.13) to \( f_{\mu}^{\text{dns}} = \frac{1}{C_{\mu} \frac{k^2}{\varepsilon}} \) is shown plotted on the top axis in the same figure.

Fig. 6.79 compares the damping function (Eq. 6.12) proposed by (Rodi & Mansour, 1993) with the damping function (Eq. 6.13) proposed in the present work. It can be seen that the damping function of (Rodi & Mansour, 1993) does not reduce the eddy viscosity enough. The damping function proposed by (Rodi & Mansour, 1993) was obtained by fitting the curve to DNS data for a channel flow.

\[
f_{\mu}(y^+) = 1 - e^{(-0.0002(y^+)^2-0.00065(y^+)^2)} \quad (6.12)
\]

\[
f_{\mu}(y^+) = a_1 + a_2(y^+) + a_3(y^+)^2 + a_4(y^+)^3 + a_5(y^+)^4 + a_6(y^+)^5 \quad (6.13)
\]

where,
\[

da_1 = 3.861722160888E - 02 \\
da_2 = -3.773486227079E - 03 \\
da_3 = 5.063878167585E - 04 \\
da_4 = -9.34915933424E - 06 \\
da_5 = 6.358066501031E - 08 \\
da_6 = -1.627591785977E - 10
\]
Figure 6.78: Spatially averaged eddy viscosity, eddy viscosity from standard \( k-\varepsilon \) model and damping function, film cooling jet, \( 574 \times 161 \times 142 \) grid

Figure 6.79: Comparison of \( \frac{1}{Re_{\tau}} \), Eq. 6.13 and Eq. 6.12, film cooling jet
6.6 Conclusion

Using Direct Numerical Simulation (DNS) all terms in the $k$ and $\varepsilon$ equations have been computed and compared with the standard $k - \varepsilon$ model. This has been done for both a channel flow and a film cooling jet flow. For the case of a channel flow, for which other researchers ((Kim et al., 1987) and (Kawamura et al., 1998)) have also computed terms in the $k$ and $\varepsilon$ equations, comparisons have been made with their results. The good comparison between the results in the present work and that of (Kim et al., 1987), (Kawamura et al., 1998) and (Debusschere & Rutland, 2002) for a channel flow generates confidence in applying the numerical scheme and code used in the present work to a representative film cooling case. A grid-independent study has been done for both the channel and film cooling flows. The residuals (i.e. the budgets of the exact $k$ and $\varepsilon$ equations) decrease as the grid spacing decreases. The spatial power spectrum has been computed for both grids used for the film cooling flow. Its behavior indicates that the important scales have been resolved. Based on these checks, the author is confident that the results presented in the present work are an accurate representation of the exact $k$ and $\varepsilon$ equations.

For the film cooling flow, statistical convergence for the turbulent Prandtl number is found to be difficult to achieve, in particular near the wall. This is believed to be a result of the zero flux wall boundary condition for the scalar.

The eddy viscosity concept has been tested extensively by comparing the DNS terms with the modeled terms used in the standard $k - \varepsilon$ model and the modeled terms using an eddy viscosity obtained from the DNS results. For the film cooling jet flow, the greatest weakness in the standard $k - \varepsilon$ model has been shown to be the choice of the eddy viscosity. If the eddy viscosity is chosen by a method that applies a minimization procedure to the DNS data, the various models for the terms in the exact $k$ and $\varepsilon$ equations represent the terms relatively well.

Unlike a channel flow, the terms in the $k$ and $\varepsilon$ equations that contain correlations involving
pressure have been shown to be significant for a film cooling jet. In addition, the correlations involving pressure have been shown to behave somewhat differently than the correlations involving just the velocity, contrary to what is typically assumed. This is of interest as turbulence models commonly model these two terms as a single diffusion process. A damping function for the eddy viscosity that has been tuned to fit a representative film cooling flow has been proposed.
Chapter 7 DNS of Pulsed Jets

7.1 Introduction

Jets in crossflow occur in a number of different applications such as film cooling for turbine blades, thrust and noise control of Vertical Take Off and Landing (VTOL) aircraft, fuel-air mixing in gas turbine combustors and pollutant dispersion from chimney stacks. Experiments with non-reacting flows (Johari et al., 1999) and (Reynolds et al., 2003) have shown that pulsing of the jet can have a large effect on the mixing, spreading and penetration of the free jet. Therefore external modulation of the jet is a potential strategy for enhancing the mixing effectiveness of jets in crossflow and can be of interest in the context of improving performance in a number of applications including combustion (by improving fuel-air mixing) and film cooling (by enhancing film cooling effectiveness).

Jets in crossflow contain a number of distinct flow structures and the role of a predictive method is to accurately capture the evolution and transport of these structures. With external modulation, these flow structures can be manipulated, possibly leading to enhanced mixing or penetration. Shear layer vortices form (Fig. 7.10) at the boundary between the jet and the crossflow. As the jet is deflected by the crossflow a Counter Rotating Vortex Pair (CVP) develops. This CVP entrains flow from the crossflow into the jet region. While it is difficult to distinguish the CVP in an instantaneous snapshot of the flow field, it can be clearly seen if the time averaged flow field is examined. If the blowing ratio is high enough, vertically oriented vortices (known as wake vortices) may form downstream of the jet. These wake vortices are inherently unsteady and do not appear if the time averaged flow field is examined. They convect downstream in the direction of the crossflow. The horseshoe vortex that forms upstream of the jet is relatively steady and appears in the time averaged flow field. Behind the jet a region of low pressure is formed in which
a recirculation pattern emerges that entrains crossflow fluid into the jet.

The goal of the present study is to computationally analyze the effect of external modulation (pulsing) on the flow structures and the resulting mixing of a jet in crossflow. As the problem is inherently unsteady (due to the external pulsing), the key to the successful prediction of such flows is the ability to resolve the dynamics of all important flow structures resulting from the interaction of the unsteady pulsed jet with the main flow in the combustor. In addition, to understand why external excitation changes the flow, it is necessary to resolve and track the different dynamic flow structures resulting from different external excitations. As a result, time averaged Reynolds Averaged Navier-Stokes solutions provide very little insight into the issues of interest in the present work. A Direct Numerical Simulation (DNS) is therefore performed, and the results are used to examine the effects of forcing on the flow structures and on the mixing. To aid in quantifying the mixing, an equation governing the evolution of a passive scalar (Eq. 2.3) is solved along with the incompressible Navier-Stokes equations (Eq. 2.2 and Eq. 2.1). Another reason for performing a DNS of this flow (as opposed to an experimental study) is that a computational study can provide information that is very difficult to obtain in an experimental study. In the present work massless particles are released into the flow at various locations. These particles are colored by their seed locations and residence time, greatly aiding the understanding of the dynamics of the flow.

7.2 Previous Work

There is an extensive body of literature concerning unpulsed jets in crossflow. Much of this work, however, has involved film cooling applications in which the blowing ratio is rarely greater than four and the angle of injection is often less than 90 degrees. (Smith & Mungal, 1998) experimentally studied a circular jet in crossflow at blowing ratios of 10 and 20. The Mach number was .29 at a blowing ratio of 20. They found that the wake vortices did not contain
fluid from the jet at a blowing ratio of 10, but did so at a blowing ratio of 20. They observed asymmetries in the cross-stream \((yz)\) plane in the time averaged results. An experimental study of a jet in crossflow has been carried out by (Fric & Roshko, 1994) for a range of blowing ratios from 2 to 10 and Reynolds numbers from 3800 to 11400. The maximum Mach number of the cases they examined was .13. By means of smoke visualization they concluded that the wake vortices do not contain fluid from the jet.

Pulsed free jets have also received attention from a number of researchers. (Hilgers, 2000) coupled an optimization method with the output from a Large Eddy Simulation (LES) solution to optimize the mixing of a free circular jet issuing from a wall. The objective function used was the integral of the radial velocity over the entire computational domain. The rationale for choosing this as a measure of mixing was that increased jet spreading should correspond to increased mixing. It was concluded that the amplitude of the forcing was not an important influence on the objective function. The parameters that were varied in order to optimize the mixing were the axial and helical forcing Strouhal numbers. While performing a DNS of a jet at a Reynolds number greater than 6000 (the lowest used by (Hilgers, 2000)), is too expensive to couple with an optimization algorithm, this is not necessarily true for an LES solution. At a Reynolds number of 6000, a \(160 \times 120 \times 32\) grid was used and at a Reynolds number of 100000, a \(252 \times 150 \times 64\) grid was used. A pronounced improvement was found in the jet spreading at axial and helical Strouhal numbers of .79 and .36 respectively. (Reynolds et al., 2003) reviewed a number of studies concerning pulsed free jets. They found that significant changes in the jet structure and spreading could be obtained by varying the axial and helical forcings. (Danaila & Boersma, 2000) performed a DNS of a pulsed free jet at a Reynolds number of 1500 on a \(192 \times 128 \times 96\) grid. They found a dramatic change in the jet structure and spreading similar to that found experimentally by (Reynolds et al., 2003).
Studies on pulsed jets in crossflow are limited and have focused attention primarily on statistical measures of mixing. (Narayanan et al., 2003) experimentally studied the effect of pulsing on an isolated circular jet in crossflow. The flow was at a jet Reynolds number of 5000, with a low Mach number ($\approx .07$) and a blowing ratio of six. They introduced a passive scalar into the flow in order to quantify mixing. They found increased mixing for pulsing at Strouhal numbers in the range $.1 - .26$. (Blossey et al., 2001) performed a numerical study of a pulsed jet in crossflow at a jet Reynolds number of 3000 on a $256 \times 120 \times 96$ grid. They examined sinusoidal forcing at Strouhal numbers in the range $.1 - .64$. Increased spreading of the jet in the wall normal direction was found at Strouhal numbers in the range $.1 - .26$. A slight increase in jet penetration, in comparison to the unpulsed jet, was found at all Strouhal numbers, with the exception of a Strouhal number of $.64$. (Johari et al., 1999) conducted an experimental study (using water) of pulsed jets in crossflow at blowing ratios of 5 and 10. The Reynolds number was $2250 \pm 50$ for a blowing ratio of 5 and $4500 \pm 100$ for a blowing ratio of 10. The Strouhal number in their experiments was relatively low, ranging from $9. \times 10^{-4}$ to $2.5 \times 10^{-2}$. They found increased jet penetration over the unpulsed jet for all pulsed jet cases. (M’Closkey et al., 2002) experimentally studied pulsed jets in crossflow at a blowing ratio of 2.58 and a jet Reynolds number of 1500. Active control of the jet pulsing mechanism was used in order to maintain a constant amplitude of forcing at the jet exit over the range of forcing frequencies studied. They present instantaneous images generated by smoke visualization, which show a splitting of the jet in the wall normal direction at Strouhal numbers of $.133, .181, .205, .266$. This splitting, which results in greatly increased jet penetration, was not seen at a Strouhal number of $.532$. They state that a significant increase in jet penetration was not found at Strouhal numbers greater than $.338$.

7.3 Problem Description and Boundary Conditions

In the present work, a DNS of a pulsed jet is performed, motivated by the need to understand the role of external forcing on the dynamics of the flow structures and on the statistical measures
of jet mixing and vertical and lateral penetration. The problem setup is similar to the experimental work of (Narayanan et al., 2003). A schematic of the flow domain along with boundary conditions and dimensions is given in Fig. 7.1. The primary differences between the present work and (Narayanan et al., 2003) are in the boundary conditions. Both the crossflow and the jet exit in (Narayanan et al., 2003) are turbulent. In the present work the boundary condition for the crossflow is steady, i.e. it is not a function of time. In addition, the pulsing in (Narayanan et al., 2003) was generated by a mechanical spinning valve, with mating holes between the spinning disk and the valve housing. In the present work, the modulation amplitude is such that the instantaneous mass flow through the jet exit is always between 80% and 120% of the time averaged mean mass flow through the jet exit. This modulation amplitude is selected in order to match the amplitudes in the measurements.

All quantities in the present work are non-dimensionalized by the jet diameter $d$ and the maximum time averaged jet velocity $U_0$. The Reynolds number based on $U_0$ and jet diameter is 5000. The Prandtl number of the passive scalar is 1. The blowing ratio, defined as $U_0$ divided by the maximum crossflow inlet velocity, is 6. Referring to Fig. 7.1, $L_y = 17d$, $L_p = 10d$, $L_{up} = 7d$, and $L_{down} = 25d$. The origin of the coordinate system in the $xz$ plane is defined to be the center of the jet. The jet is injected at 90 degrees to the crossflow. At the crossflow inlet boundary the scalar is set to a value of zero. Periodic boundaries are used in the $z$ direction. Except for the jet, a no slip wall boundary is applied in the jet exit ($xz$) plane. At this boundary, with the exception of the jet exit, the flux of the scalar normal to the wall is set to zero. At the jet exit, which is defined by a circle of diameter one centered at $x = 0$, $z = 0$ in the $xz$ plane, the scalar is set to a value of one. The $u$ and $w$ components of velocity are set to zero in the jet exit plane. The $v$ velocity in the jet exit is set by Eq. 7.1.

$$v_{jet}(t) = (U_0 + U_p(t)) e^{(2r)^n} \text{ where } 0 \leq r \leq \frac{1}{2}$$ (7.1)
The mean jet exit boundary velocity profile in Eq. 7.1 is used to approximate the profile in a pipe flow. The pulsing is accomplished by varying \( U_p \) in Eq. 7.1 as a function of time. A sine wave, with an amplitude of \( 0.2U_0 \), is used. The jet velocity \( v_{\text{jet}}(t) \) fluctuates around \( U_0 \) to maintain the same time averaged mass flow rate from the jet regardless of the amplitude and frequency chosen. This makes it easier to compare the mixing for the various cases studied.

The far field boundary condition at \( y = L_y \) is \( v = w = 0, u = \frac{U_0}{6} \) and the flux of the scalar normal to this boundary set to zero. A convective outflow boundary condition (Eq. 3.20) is applied for all three velocities and the scalar at \( x = L_{\text{down}} \). At the crossflow inlet boundary at \( x = L_{\text{up}} \), the \( v \) and \( w \) components of velocity are set to zero while the following profile is used for
the $u$ velocity.

$$u(y) = \begin{cases} \frac{U_0}{d} \left( \frac{y}{15d} \right)^{\frac{3}{2}} & y \leq 15d \\ \frac{U_0}{d} & y > 15d \end{cases}$$

This crossflow inlet profile assumes a turbulent boundary layer thickness of $15d$ and a crossflow freestream velocity of $\frac{U_0}{d}$.

The forcing frequency can be characterized by a dimensionless jet Strouhal number $St = \frac{kd}{U_0}$ where $k$ is the forcing frequency in Hz. In the course of the present work, a range of Strouhal numbers (.025, .05, .1, .2, .4, .6, .8 and 1.) is examined for sine, square and spiked wave forms on a coarse ($205 \times 140 \times 66$) grid. This is done to inexpensively identify combinations of Strouhal numbers and wave forms for which there are significant changes in the flow. Based on this, a DNS is performed of the sine wave forms at three Strouhal numbers (.2, .4 and .6), in addition to the unpulsed case. These Strouhal numbers were chosen because large changes in the jet mixing, spreading and penetration were observed at these frequencies on the coarse grid. Results from these cases are presented in this work.

### 7.4 Numerical Issues

Solutions are obtained on three different grids. The dimensions of the grids are $205 \times 140 \times 66$, $410 \times 280 \times 130$, $820 \times 560 \times 260$. Each larger grid is obtained by doubling the number of grid points in each dimension of the next smallest grid. The results in the present work are obtained on the $410 \times 280 \times 130$ grid, with the other grids being used to examine the effect of the grid spacing on the solution. The grids are stretched in all three dimensions to concentrate points near the jet exit. In particular, the resolution decreases as the outflow boundary is approached. The time steps used are $0.02 \frac{d}{U_0}, 0.01 \frac{d}{U_0}, 0.004 \frac{d}{U_0}$ for the $205 \times 140 \times 66$, $410 \times 280 \times 130$, $820 \times 560 \times 260$ grids respectively. This results in maximum CFL numbers at any grid point in the domain of approximately .3.

A sixth-order central difference convective scheme (Eq. A.35) with a monotonic limiter is used. Fourth-order interpolation for the velocity in the convective terms e.g. for $v$ in $v \frac{\partial u}{\partial y}$ etc. is
used. A fourth-order central difference scheme is used for the diffusive terms. A third-order accurate explicit time integration scheme (Eq. A.41) is used to integrate the convective and diffusive terms in time. The pressure gradient and the continuity equation are represented by second-order centered schemes. The equation for the transport of the passive scalar is represented using the same order and type of schemes used for the momentum equations. Three subiterations per physical time step are used in the colored SCGS method. The accuracy of all finite difference stencils is maintained near non-periodic boundaries by shifting toward the boundary the point at which the stencil is evaluated.

As the boundary conditions are spatially symmetric in the $z$ direction, the symmetry of the flow field is broken in the $z$ direction in order to hasten the development of asymmetric turbulence. This is accomplished by setting, as an initial condition, the $u$ velocity equal to .25 of its value inside of a sphere of diameter $\frac{1}{2}d$ centered at $(-1d, 2d, .2d)$. In order to determine when sufficient time has passed so that the flow can be considered fully developed and statistically independent of the initial conditions, the residual history and the integrated averages of the scalar over three planes are monitored as a function of time. Figure 7.2 shows the residual history for the unpulsed case and for sine wave pulsing at $St = .4$. Figure 7.3 shows the history of the integrated averages $\left( \int_{y}^{z} s\left(\frac{1}{2} \left( L_{up} + L_{down}\right), y, z\right) dy dz, \int_{x}^{z} s\left(x, \frac{L_y}{2}, z\right) dx dz, \int_{x}^{y} s\left(x, y, 0\right) dx dy \right)$ of the scalar over the $yz$, $xz$ and $xy$ planes for the same cases. All simulations run on the $410 \times 280 \times 130$ grid are run for 28800 time steps, which results in the flow (based on the maximum crossflow velocity i.e. $\frac{U_{0}}{6}$) traveling $48d$. It can be seen in Fig. 7.2 and Fig. 7.3 that the flow can be considered fully developed when this time step has been reached. At this point statistics are collected at each time step until the flow had traveled a further $534d$. As statistics are collected at every time step, this results in statistics being averaged over 160201 distinct realizations of the flow field.
average residual of continuity equation
average residual of X momentum equation
average residual of Y momentum equation
average residual of Z momentum equation

Figure 7.2: Residual history, 410 × 280 × 130 grid, (a) unpulsed (b) $St = .4$
Figure 7.3: Scalar history, $410 \times 280 \times 130$ grid, (a) unpulsed (b) $St = .4$
7.5 Measures of Mixing

Since scalar mixing is of interest, three measures relating to the mean scalar distribution are outlined here.

7.5.1 Jet Trajectory

The jet trajectory can be defined by considering the first moment of the scalar field in $yz$ planes, resulting in the centriodal axis of the scalar distribution in these planes. Using an analogy from mechanics, this can be thought of as the center of mass of the scalar field in the $yz$ plane.

$$C_y(x) = \frac{\int \int y s(x,y,z) dy dz}{\int \int s(x,y,z) dy dz} \quad C_z(x) = \frac{\int \int z s(x,y,z) dy dz}{\int \int s(x,y,z) dy dz}$$

7.5.2 Jet Spreading

Jet spreading can be characterized by the second moments of the scalar field and can be defined by the moments of inertia of the scalar field in the $yz$ plane. In order to be meaningful these second moments must be determined relative to the centriodal axis of the cross section under consideration. The spreading of the jet in the $y$ and $z$ directions respectively is given by:

$$S_y(x) = \frac{\int \int (y - C_y(x))^2 s(x,y,z) dy dz}{\int \int s(x,y,z) dy dz} \quad (7.2)$$

$$S_z(x) = \frac{\int \int (z - C_z(x))^2 s(x,y,z) dy dz}{\int \int s(x,y,z) dy dz} \quad (7.3)$$

Another measure of mixing can be defined by considering the mean square of the difference of the scalar field with the scalar value in the limit of fully mixed conditions ($s_\infty$). Smaller values of $S_\infty(x)$ imply better mixing. This measure is determined over $yz$ planes making it a function of the downstream distance.

$$S_\infty(x) = \int \int (s(x,y,z) - s_\infty)^2 dy dz \quad (7.4)$$

where

$$s_\infty = \frac{\int \int_{crossflow \ inlet} s(x_{crossflow \ inlet}, y, z) u(x_{crossflow \ inlet}, y, z) dy dz}{\int \int_{crossflow \ inlet} u(x_{crossflow \ inlet}, y, z) dy dz + \int \int_{jet \ exit} v(x, y_{jet \ exit}, z) dx dz}$$

This measure of mixing is not unique as the value of $s_\infty$ changes and tends towards zero as the
freestream boundary is extended in the $y$ direction. It also depends on the mass flow out of the jet. However, given a finite freestream boundary, it provides a measure that can quantify the effect of applying forcing to cases that are otherwise identical.

7.6 Visualization of Flow Structures

In order to visualize flow structures, massless passive particles are released at various locations in the domain. These particles are particularly useful in uncovering structures in the flow, as they simulate and extend the capabilities of the smoke traces commonly used in experiments to visualize flow structures. At each time step particles are released at the seed locations shown in Fig. 7.4. Thirty-one particles are released at the red locations which are in a circular plane within the jet in the jet exit plane. One hundred particles are released at the blue locations, which is a line that extends from $-1.5$ to $1.5$ in the $z$ direction, at $x = -1.5$, $.5$ from the jet exit plane in the $y$ direction. The cyan locations are identical except that the line is $.05$ from the jet exit plane in the $y$ direction. One hundred particles are released at the pink locations, which is a line that extends $.05$ from the jet exit plane to ~6 in the $y$ direction, at $x = -1.5$, $z = 0$. One hundred and fourteen particles are released at the yellow locations, which are in a plane at $x = -1$ that extends from $y = .05$ to $y = .5$ from the jet exit plane and $z = -.85$ to $z = .85$. One hundred and thirty-five particles are released at the green locations, which are in a plane $.2$ from the jet exit plane in the $y$ direction which extends from $x = 1$ to $x = 4$ and $z = -1$ to $z = 1$.

Another means of visualizing the flow is that of isosurfaces of various quantities. Due to storage and computational constraints the isosurfaces are determined by considering only every other point in each dimension on the $410 \times 280 \times 130$ grid. Once an isosurface is defined, other quantities can be interpolated onto it and used to color the isosurface. Both methods are used in this work to analyze the features of the flow.

In the course of this research a large number (~400) of movies have been made of various quantities. These movies have been analyzed in order to understand the dynamics and statistics of
selected frames from these movies are shown to illustrate various aspects of the flow.

7.7 Comparison with Experimental Data

While the blowing ratio and Reynolds number in the present work are the same as the experimental work of (Narayanan et al., 2003), the boundary conditions are different. (Narayanan et al., 2003) had a turbulent cross flow and jet exit condition, while the present work uses boundary conditions (aside from the applied pulsing) that are not a function of time. A comparison of the instantaneous jet scalar with the results of (Narayanan et al., 2003) is shown in Fig. 7.5. In both works, jet penetration, spreading and turbulent structure is qualitatively similar, with the shear layer vortices appearing as circular ring shaped structures. The mean scalar field in Fig. 7.6 is somewhat different in the region on the downstream side of the jet, in which (in the
present work) there is a region in which the scalar projects further downstream than in (Narayanan et al., 2003). However, the overall jet penetration is similar in each work. A noticeable degree of asymmetry in the scalar and velocity fields can be seen in both works in Fig. 7.7. The results of (Narayanan et al., 2003) show the asymmetry appearing in the CVP, while the present work finds a significant asymmetry in the crossflow beneath the CVP as well as in the CVP. This asymmetry is discussed further in a later section.
Figure 7.7: Time averaged scalar in $yz$ plane at $x = 4d, 6d$
7.8 Spatial Energy Spectra

A one-dimensional spatial energy spectrum in the $x$ direction is given in Fig. 7.8. The spectrum is shown in the jet center-plane at different distances in the $y$ direction from the jet exit. This spectrum is obtained by interpolating the velocity on the uneven spatial grid onto an even grid which is defined by the smallest grid spacing in the uneven grid and then transforming that velocity into discrete wave number space. Regions in which the slope of the spectra matches the $-5/3$ slope of the inertial subrange predicted by turbulence theory (Kolmogoroff, 1941) can be seen in Fig. 7.8. Close agreement can be seen between the results obtained on the $410 \times 280 \times 130$ grid and on the $820 \times 560 \times 260$ grid. This indicates that the $410 \times 280 \times 130$ grid is fine enough to resolve the important spectra. Energy pile-up can be seen as one moves further away from the jet (in the $y$ direction). This is not unexpected as the grid spacing is increased as one moves away from the jet in the $y$ direction. As a result, energy in the smallest scales cannot be resolved and is aliased back into the larger scales. This is generally not desirable, but it as it occurs relatively far away from the jet the effect should be minimal.

7.9 Unpulsed Jet

7.9.1 Flow Dynamics

Two views of the unpulsed jet are given in Fig. 7.9. A splitting of the jet can be seen
approximately $2d$ from the jet exit with the bulk of the fluid retained in the primary jet stream. As will be shown later, this splitting or bifurcation is enhanced by pulsing at certain frequencies and contributes to changes in the vertical and lateral penetration and spreading of the jet.

The leading edge shear layer vortices are shown in Fig. 7.10. The shear layer vortices can be seen to form at the front of the jet and to break down into complex three-dimensional structures within $2d - 3d$ of the jet exit.

Strong wake vortices form immediately behind the jet. (Fric & Roshko, 1994) note that the wake vortices do not contain any jet fluid. In their experimental work this was determined by placing smoke traces at various locations in the flow. Based on this they believe that, because the only other source of vorticity is the wall boundary layer, the wake vortices have their origin there. In the present work a different origin is proposed for the wake vortices. Based on particle traces it can be ascertained that the wake vortices do not contain any jet fluid. The core of the wake vortices has its origin in fluid (i.e. green particles) at the back of the jet close to the wall. This fluid is entrained into the leeward jet shear layer vortices. Part of this fluid, at the very back of the jet, is stripped off by the crossflow and forms the seed or core of the wake vortex. As the wake vortex moves downstream it grows in diameter as crossflow fluid, originating upstream of the jet, wraps around the seed or core of the vortex. A series of images illustrating this is in Fig. 7.11 and Fig. 7.12. As the wake vortices move downstream they feed fluid from the crossflow into the jet. The center of the vortices continues to be dominated by fluid originating at the back of the jet close to the wall. This fluid travels along the center of the wake vortices in a corkscrew fashion upwards into the jet. It is surrounded by fluid originating upstream of the jet, which also travels in a corkscrew fashion upwards into the jet. A schematic of the pattern and direction of rotation of the wake vortices is shown later in Fig. 7.25a. While the dominant direction of rotation is clockwise, very weak counter-clockwise rotating vortices exist in between the strong clockwise
rotating vortices. The wake region is more disordered than observed in the pulsed jets discussed later, indicating that the pulsing imposes a degree of order.

Figure 7.9: Instantaneous isosurface of scalar colored by magnitude of the gradient of the scalar, unpulsed

Figure 7.10: Particles released from pink locations in Fig. 7.4, colored by residence time, unpulsed
Figure 7.11: Wake vortex visualization by particle traces, unpulsed
Figure 7.12: Wake vortex visualization by particle traces, unpulsed

217
7.9.1.1 Asymmetry

There is an asymmetry about the $z$-axis in the time averaged results (which is discussed later) and the instantaneous flow. The instantaneous flow which leads to the time averaged asymmetry can be seen in Fig. 7.14a, in which the particles (on the right side looking upstream) of the line of blue particles released upstream at $y = .5d$ rise much higher than those on the other side. These particles wrap around the outside of the wake vortex and cause the wake vortex pattern to become asymmetric. In the animations made (which cover a time period over which the crossflow travels $47d$) the asymmetry is not observed to switch. Asymmetry can be seen in Fig. 7.14b as the pink particles (released in a vertical line at $z = 0$) migrate towards the negative side of the $z$-axis. The green particles, released symmetrically in the wake region, asymmetrically migrate in the negative $z$ direction (Fig. 7.14b). These asymmetries in the instantaneous flow lead to the development of a distinctly asymmetric flow in the time averaged results. This issue is discussed in greater detail in a later section.
Figure 7.14: Asymmetry visualization by particle traces, unpulsed
7.9.2 Temporal Spectra

The temporal frequency spectra of the vertical component of velocity for the unpulsed jet is shown at different $y$ locations along the upstream edge of the jet ($x = -0.5d$) in Fig. 7.13. The dominant preferred mode of the unpulsed jet is $St = .35$. At $y = 1d$, a distinct frequency can be seen at $St = .7$ which, due to vortex merging, leads to a frequency of $St = .35$ by $y = 2d$. Beyond $y = 3d$, only the $0.35$ frequency can be seen.

7.9.3 Statistics

Figures 7.15 and 7.16 show an isosurface of the time averaged scalar field onto which the turbulent kinetic energy and the turbulent scalar fluctuations have been interpolated respectively. The asymmetry observed in the instantaneous images is also seen in this isosurface. The scalar turbulent fluctuations have a different trend than the turbulent kinetic energy. In particular, the scalar turbulent fluctuations are highest at the front of the jet where the crossflow meets the jet and where the scalar gradients are high. This is consistent with the gradient approximation used in many turbulence models, which makes turbulent stresses (or fluctuations) proportional to a mean gradient. The turbulent kinetic energy however, is highest at the side of the jet. The small splitting of the jet observed in the instantaneous images (Fig. 7.9a) appears clearly in these time averaged figures, reflecting the steady nature of the bifurcation. Isosurfaces of the scalar fluctuations can be seen in (Fig. 7.17). In this figure the small splitting of the jet is quite evident, indicating that a significant degree of scalar fluctuations exists in the split jet. The asymmetry is very evident in this isosurface.

Figure 7.18 shows the mean velocity vectors in the $yz$ plane at different $x$ locations. A large asymmetry can be observed in the CVP. Below the CVP are two small vortices with the same sense of rotation (clockwise, looking upstream). The asymmetry becomes more pronounced as the distance downstream increases.
Figure 7.15: Isosurface of mean scalar colored by $k$, unpulsed

Figure 7.16: Isosurface of mean scalar colored by $s^0$, unpulsed

Figure 7.17: Isosurface of $s^0$ colored by $\|\nabla s^0\|$, unpulsed
Figure 7.18: Velocity vectors of mean field colored by mean scalar at $x = 3d$, $x = 5d$, $x = 8d$, $x = 15d$, roughly every 12th vector shown, unpulsed
7.10 Sine Wave Pulsing

7.10.1 Flow Dynamics

Figures 7.19, 7.20 and 7.21 present instantaneous isosurfaces of the scalar colored by its gradient for Strouhal numbers of .2, .4 and .6 respectively. A distinct difference is noticed for pulsing at these Strouhal numbers relative to each other and to the unpulsed case. At $St = .2$ significant spreading occurs in the vertical direction relative to the unpulsed case. The weak bifurcation noted for the unpulsed case (Fig. 7.9) is considerably strengthened by pulsing (Fig. 7.19). The primary and secondary jets, which spread apart in the vertical direction as the flow travels downstream, result in greatly increased vertical spreading. At $St = .4$ the jet breaks into three jets. This breakup occurs in the $xy$ plane and begins approximately $1d$ from the jet exit in the vertical direction. Two of the jets (primary and secondary) are large, while the third (tertiary), in the middle, is much smaller. The whole jet penetrates and spreads much more in the $y$ direction than the unpulsed case, but has significantly reduced spreading in the $z$ direction. While the maximum jet penetration at $St = .2$ and $St = .4$ is very similar, the secondary jet at $St = .4$ is larger and closer to the wall. At $St = .6$ the jet splits into two jets in the $xz$ plane (Fig. 7.21b). The result is a jet that spreads significantly in the $z$ direction but which penetrates and spreads less in the $y$ direction relative to the unpulsed case.

The frequency of the shear layer vortices which form at the front of the jet is that of the imposed pulsing. At $St = .2$ the shear layer vortices are less distinct than at $St = .4$ and $St = .6$, although they are more distinct than the unpulsed case. Particle tracking gives clear evidence of shear layer vortex merging at $St = .2$ and $St = .6$, although evidence of vortex merging at $St = .4$ appears in the temporal spectrum. A sequence of images showing this merging at $St = .2$ is given in Fig. 7.22. The merging occurs close to the jet exit at $y = 2.5d$. In Fig. 7.22, the merging vortices are labeled "1" and "2", while the circled vortices are shear layer vortices that do not merge. This pattern of a vortex that does not merge in between a pair of vortices that do
merge is consistently seen in the animations. Note that \( St = .2 \) is close to the first subharmonic of the preferred mode. Shear layer vortex merging with a different pattern is seen at \( St = .6 \) in Fig. 7.24. Here every shear layer vortex merges, although at a greater distance (~4.25\(d\)) from the jet exit. This merging of the shear layer vortices at the front of the jet occurs at approximately the same location that the jet bifurcates in the \( z \) direction and is likely related to the bifurcation. Very clear and distinct (until they begin to break up at \( y = 4.25d \)) shear layer vortices at \( St = .4 \) are shown in Fig. 7.23. As \( St = .4 \) is close to the lowest natural frequency (\( St = .35 \)) the pulsing regularizes the vortex formation. No clear evidence of vortex merging is seen until \( y = 5d \) and \( 6d \), where the temporal spectrum (discussed later) shows several lower frequencies, indicating complex collective interaction.

Wake vortices are seen at \( St = .2 \) and \( St = .4 \) (see Appendix F) but not at \( St = .6 \). The origin of the wake vortices is the same as for the unpulsed jet in that it consists of crossflow fluid (not necessarily from the boundary layer) which is entrained into the jet and then stripped off the
Figure 7.20: Instantaneous isosurface of scalar colored by magnitude of the gradient of the scalar, $St = .4$

Figure 7.21: Instantaneous isosurface of scalar colored by magnitude of the gradient of the scalar, $St = .6$
Figure 7.22: Vortex merging, $St = .2$

Figure 7.23: Particles released from pink locations in Fig. 7.4, colored by residence time, $St = .4$
Figure 7.24: Vortex merging, $St = .6$

Figure 7.25: Wake vortex schematic

back of the jet by the crossflow. A schematic describing the positions and sense of rotation of the wake vortices is given in Fig. 7.25. For the unpulsed case and at $St = .2$, the wake vortex pattern is asymmetric, with the wake vortices appearing only on the positive side of the $z$-axis. Figures 7.27 and F.1 (see Appendix F) present a series of images illustrating the formation of the wake vortices close to the jet exit at $St = .2$. The wake vortices occur in what appear to be alternating
pairs, but which is actually one vortex bent into a U-shaped loop (Fig. 7.29). The side of the U-shaped vortex closest to the $z$-axis contains fluid mostly from the crossflow (yellow particles), while the side furthest from the $z$-axis contains the wake fluid (green particles) as its core, with entrained crossflow boundary fluid (yellow particles) around the periphery of the vortex. This is likely a result of the asymmetry of the flow in the region behind the jet, close to the wall. At $St = .2$ and $St = .4$ as the wake vortices move downstream, the center of the vortices continues to be dominated by fluid originating behind the jet close to the wall. This fluid, and the crossflow wrapped around it, travels along the center of the wake vortices in a corkscrew fashion upwards into the jet, representing a mechanism (absent at $St = .6$) by which the jet is diluted by the surrounding flow. The effect of this mechanism can be seen in Fig. 7.32a, where at $St = .6$ many more of the green particles remain near the wall and spread much further in the downstream and cross-stream direction in comparison to $St = .2$ (Fig. 7.28a) and $St = .4$ (Fig. 7.31a). Such spanwise spreading may potentially be very useful in film cooling applications.

7.10.1.1 Asymmetry

There is an asymmetry about the $z$-axis in the time averaged results at $St = .2$, but not at $St = .4$ or $St = .6$. The instantaneous flow leading to the time averaged asymmetry can be clearly seen in Fig. 7.28a in which the blue particles on the negative side of the $z$-axis of the line of blue particles released upstream at $y = .5d$ rise much higher than those on the other side. These particles wrap around the outside of the wake vortex and cause the wake vortex pattern to become asymmetric. Asymmetry can be seen in Fig. 7.28b as the pink particles migrate around the jet towards the negative side of the $z$-axis. Asymmetry is evident very close (.05$d$) to the wall in Fig. 7.26a, which shows the particles released at the cyan locations in Fig 7.4. For comparison, at $St = .4$ the flow is symmetric even after being entrained into the back of the jet (Fig. 7.26b). The wake vortices are tilted in the positive $z$ direction (Fig. 7.30), while the green particles seeded behind the jet that remain near the wall (Fig. 7.28b) move in the opposite direction. This tilting
Figure 7.26: Particles released $0.05d$ from wall colored by residence time, (a) $St = 0.2$, (b) $St = 0.4$

results in movement of the wake vortices away from the wake flow immediately behind the jet, which results in their liftoff and separation from the wake flow (Fig. 7.29). In the animations made (which cover a time period over which the crossflow travels $47d$) the asymmetry is not observed to switch. This asymmetry is very similar to the unpulsed case although the wake vortices have a different pattern (Fig. 7.25b). No such dramatic asymmetry in the instantaneous flow is observed at Strouhal numbers of 0.4 (Fig. 7.31) and 0.6 (Fig. 7.32).
Figure 7.27: Wake vortex visualization by particle tracking, $St = .2$
Figure 7.28: Wake region visualization by particle tracking, $St = .2$
Figure 7.29: Wake vortex visualization by particle tracking, $St = .2$

Figure 7.30: Wake vortex visualization by particle tracking, $St = .2$
Figure 7.31: Wake region visualization by particle tracking, $St = A$
Figure 7.32: Wake region visualization by particle traces, \( St = .6 \)
7.10.2 Temporal Energy Spectra

Figures 7.33, 7.34 and 7.35 present the temporal frequency spectra of $v$ at the leading edge of the jet at various $y$ locations for Strouhal numbers of .2, .4, and .6 respectively. At all Strouhal numbers the dominant frequency corresponds to the jet pulsing frequency. In all cases higher harmonics of the jet pulsing frequency can be seen. At $St = .2$ evidence of weak shear layer vortex merging was observed in Fig. 7.22 and is evident in the temporal spectra at around $y = 5d$. At $St = .4$ subharmonics with significant energy appear at $y = 5d$ and $6d$, indicating interaction between vortical structures resulting in a number of vortex mergings. At $St = .6$, a distinct subharmonic equal to half the imposed pulsing frequency can be seen. This subharmonic is a result of the vortex merging shown in Fig. 7.24. It reaches its maximum value, among the locations shown, at $y = 4d$, which is very close to the location ($y \approx 4.25$) where the vortex pairing is observed to occur in Fig. 7.24. The higher harmonics are very weak at this Strouhal number.

7.10.3 Statistics

The $y$ component of the jet trajectory is shown in Fig. 7.36. Pulsing at Strouhal numbers of .2 and .4 results in greater penetration in the $y$ direction than the unpulsed case, with both cases penetrating $1.25d$ more than the unpulsed case at $x = 20d$. Pulsing at $St = .6$, which causes the jet to bifurcate in the $z$ direction, results in a jet which penetrates less than the non-pulsed case, and much less than pulsing at Strouhal numbers of .2 and .4. The asymmetry of the unpulsed and the $St = .2$ jet appears in the $z$ component of the jet trajectory shown in Fig. 7.38, in which the jet center moves as far away as $6d$ (unpulsed) and $4d$ ($St = .2$) from the $z$-axis. The cases in which the jet penetrates furthest are also the cases which exhibit the largest spreading (see Fig. 7.37) in the $y$ direction (as defined by Eq. 7.2). Thus, the greatest spreading in the $y$ direction is obtained at $St = .4$ followed by the $St = .2$, unpulsed, and $St = .6$ cases. Spreading in the $z$ direction (defined by Eq. 7.3) increases in the reverse order of spreading in $y$ (Fig. 7.39).

The measure of mixing given by Eq. 7.4 is shown in Fig. 7.40. By this measure the non-pulsed
case has less mixing than any of the pulsed cases, with $St = .4$ and $St = .6$ having the best mixing.

Figure 7.33: Temporal frequency spectra of $v$ at $x = - .5$, $z = 0$, $St = .2$

Figure 7.34: Temporal frequency spectra of $v$ at $x = - .5$, $z = 0$, $St = .4
Figure 7.35: Temporal frequency spectra of $v$ at $x = -.5, z = 0, St = .6$

Figure 7.36: Jet trajectory in $y$, unpulsed and pulsed
Figure 7.37: Jet spreading in $y$, unpulsed and pulsed

Figure 7.38: Jet trajectory in $z$, unpulsed and pulsed
Figure 7.39: Jet spreading in $z$, unpulsed and pulsed

Figure 7.40: $S_\infty(x)$, unpulsed and pulsed
The bifurcation ($St = .2$) and trifurcation ($St = .4$) in the vertical direction and the bifurcation in the horizontal direction ($St = .6$) can be seen in the mean scalar isosurfaces (colored by the turbulent kinetic energy) of Figures 7.41, 7.42 and 7.43 respectively. Asymmetry at ($St = .2$) is observed, particularly of the secondary jet nearest the wall. For $St = .2$ and $St = .4$ the kinetic energy of the secondary jets is less than that of the main jet. At $St = .4$ the turbulent kinetic energy is maximum at the front of the jet at $y \approx 5d$, while at $St = .2$ it is maximum more towards the side of the jet and lower ($y \approx 3.5d$). The lateral spreading is the greatest and the turbulent kinetic energy is least intense at $St = .6$.

Isosurfaces of the mean scalar field colored by the turbulent scalar fluctuations are shown in Figures 7.44, 7.45 and 7.46. At $St = .2$ and $St = .4$ the turbulent scalar fluctuations are largest at the front and side of the jet, while at $St = .6$ they are large at $y \approx 3.5d$, which is where vortex merging is observed (Fig. 7.24). On this isosurface the turbulent scalar fluctuations are least intense at $St = .6$. In all cases there is a region at the back of the jet where the turbulent scalar fluctuations are large.

If isosurfaces of the scalar fluctuations are examined, the splitting of the jet into three jets at $St = .4$ is even more pronounced (Fig. 7.48) compared to the unpulsed case (Fig. 7.17). At $St = .4$ the secondary jet nearest the wall extends significantly further downstream than the main jet, which does not happen for $St = .2$, which indicates that a significant degree of turbulence exists in the secondary jet. The asymmetry of the secondary jet at $St = .2$ is quite apparent in Fig. 7.47. At $St = .6$ the splitting of the jet into two distinct jets in the $z$ direction can be seen quite clearly in Fig. 7.49. These two jets correspond to the CVP.

For $St = .4$ and $St = 6$ the isosurfaces are very symmetric, indicating that the asymmetry observed in the unpulsed and $St = .2$ cases is not the result of an insufficient time averaging period.
Figure 7.41: Isosurface of mean scalar colored by $k$, $St = .2$

Figure 7.42: Isosurface of mean scalar colored by $k$, $St = .4$

Figure 7.43: Isosurface of mean scalar colored by $k$, $St = .6$
Figure 7.44: Isosurface of mean scalar colored by $s'$, $St = .2$

Figure 7.45: Isosurface of mean scalar colored by $s'$, $St = .4$

Figure 7.46: Isosurface of mean scalar colored by $s'$, $St = .6$
Figure 7.47: Isosurface of $s' s'$ colored by magnitude of the gradient of $\overline{\sigma}$, $St = .2$

Figure 7.48: Isosurface of $s' s'$ colored by magnitude of the gradient of $\overline{\sigma}$, $St = .4$

Figure 7.49: Isosurface of $s' s'$ colored by magnitude of the gradient of $\overline{\sigma}$, $St = .6$
Figure 7.50: Velocity vectors of mean field colored by mean scalar at $x = 3d$, $x = 5d$, $x = 8d$, $x = 15d$, roughly every 12th vector shown, $St = .2$
Figure 7.51: Velocity vectors of mean field colored by mean scalar at $x = 3d$, $x = 5d$, $x = 8d$, $x = 15d$, roughly every 12th vector shown, $St = .4$
Figure 7.52: Velocity vectors of mean field colored by mean scalar at $x = 3d, x = 5d, x = 8d, x = 15d$, roughly every 12th vector shown, $St = .6$
7.11 Asymmetry

Figure 7.50 shows mean velocity vectors in the $yz$ plane at different $x$ locations at $St = .2$. A strong asymmetry can be seen beneath the CVP although, unlike the unpulsed case (Fig. 7.18), the CVP is relatively symmetric. Immediately below the CVP (on the left side) is a counterclockwise rotating vortex that greatly increases in size as the distance downstream increases. Below the CVP near the wall are two small vortices with the same sense of rotation (clockwise, looking upstream). The asymmetry becomes more pronounced as the distance downstream increases. There is no significant asymmetry at Strouhal numbers of .4 (Fig. 7.51) and .6 (Fig. 7.52). The asymmetry in $\overline{w}$ can be clearly seen in Fig. 7.53, in which the asymmetry is more pronounced in the unpulsed case than at $St = .2$. If symmetry exists at the jet center-plane ($z = 0$), then for this flow $\overline{w^'u^'} = \overline{w^'v^'} = 0$ in the $xy$ plane at $z = 0$ (see Appendix E for proof). Figures 7.54 and 7.55 show that, for the unpulsed and $St = .2$ cases, these turbulent stresses are nearly as large as the $\overline{u^'v^'}$ turbulent stress (Fig. 7.56), while for the symmetric cases $\overline{w^'u^'}$ and $\overline{w^'v^'}$ are essentially zero.

While it is often assumed that symmetry will exist in the time averaged results, this may not be the case. (Smith & Mungal, 1998) found asymmetry in $yz$ planes at blowing ratios of 10 and 20. They determined asymmetries by considering the distribution of a scalar injected into the
Figure 7.54: $\overline{uw'}$ at $z = 0$

Figure 7.55: $\overline{w'v'}$ at $z = 0$

Figure 7.56: $\overline{uw'}$ at $z = 0$
jet. As a result, they were unable to determine asymmetries associated with the crossflow, such as the small vortices that are observed close to the wall in Figures 7.18 and 7.50. (Narayanan et al., 2003) found asymmetries for the unpulsed jet. They suggested the asymmetries were due to an insufficient time averaging period. In the present work, the asymmetry exists only for the unpulsed jet and for sine wave pulsing at \( St = 0.2 \). While it is impossible in an experiment to remove all asymmetries regarding elements such as the incoming flow, the physical geometry of the jet nozzle etc., much greater control exists in computations. In the present work, there are two sources of asymmetry - roundoff error, and symmetry breaking in the initial conditions. The roundoff error is a source of asymmetry because, while the same numbers are in a finite difference stencil on both sides of the \( z \)-axis, the numbers are added together in a different order on each side of the \( z \)-axis. No asymmetry is introduced by the boundary conditions, as they are symmetrical about the \( z \)-axis. The asymmetry caused by the roundoff error is very small, while the asymmetry resulting from the initial conditions should decay in a time averaged sense if physically an asymmetry does not exist. This asymmetry may disappear if a longer time averaging period is used. If so, the observed asymmetry is simply an extremely low frequency oscillation. In order to determine if this is the case, it would be necessary to run the simulation and time average the results over a much longer length of time than in the present work. This has not been done due to the large computational expense. It should be noted that the statistics in the present work have been averaged over 160201 consecutive time steps, a length of time that corresponds to the flow traveling 534\( d \), based on the maximum crossflow velocity.

Fig. 7.57 shows the results of different symmetry breaking in the initial conditions. Fig. 7.57a shows the result of the symmetry breaking used in all cases in this work. Fig. 7.57b shows the result of a mirror image (about the \( z \)-axis) of the symmetry breaking used as an initial condition. Fig. 7.57c shows the result of no symmetry breaking. It can be seen that the flow can assume
either of these two apparently mirror image states. Therefore it can be concluded that for cases in which asymmetry is seen, the flow is very susceptible to asymmetries and that one of the two asymmetric states would appear in an experiment, in which asymmetries in the initial and boundary conditions would likely be larger than the computations.

![Figure 7.57: Velocity vectors of mean field colored by mean scalar at $x = 15d$, different symmetry breaking initial conditions, unpulsed](image)

### 7.12 Conclusion

A Direct Numerical Simulation of incompressible circular pulsed jets in crossflow at a blowing ratio of 6 is performed with the goal of examining the effect of sinusoidal pulsing on the dynamic flow structures and resultant statistical measures of mixing. Quantification of mixing is accomplished by solving an equation governing evolution of a passive scalar. Particle tracking is used to understand the formation and evolution of the shear and wake layer vortices.

The shear layer vortices are strongly affected by pulsing, with different vortex merging patterns depending on the pulsing frequency. A different origin for the formation of the wake vortices than that proposed by (Fric & Roshko, 1994) has been discovered for both pulsed and unpulsed jets. The formation of the wake vortices is found to be initiated by the entrainment into the jet of fluid from the region behind the jet and its subsequent stripping from the back of the jet by the crossflow. Pulsing is shown to drastically change the jet spreading and penetration and to increase
the mixing of the jet with the crossflow. In particular, pulsing at \( St = 0.4 \) causes the jet to split into three jets in the vertical direction and to penetrate much further in the vertical direction. Pulsing at \( St = 0.6 \) causes the jet to split in the lateral direction and to significantly reduce penetration in the vertical direction. In addition, pulsing at \( St = 0.6 \) is seen to eliminate the wake vortices. As these wake vortices provide a mechanism by which fluid originating behind the jet near the wall is fed into the jet, pulsing may have practical applications in suppressing the transport to the jet of pollutants from this region.

Isosurfaces of the scalar and the turbulent scalar fluctuations are used to illustrate the effects of pulsing on the time averaged scalar and velocity fields. Measures of mixing and spreading are applied to the time averaged scalar field, with the finding that pulsing improves mixing and greatly effects jet spreading.

A significant asymmetry primarily affecting the wake vortices has been found for only the unpulsed and pulsing at \( St = 0.2 \) cases. The finding of this asymmetry, which can occur in one of two mirror images, confirms that found in the experimental works of (Smith & Mungal, 1998) and (Narayanan et al., 2003). This asymmetry is clearly seen in both the instantaneous and time averaged scalar and velocity fields.
References


Appendix A  Finite Difference Methods

A.1 One Dimension

The finite difference method is a subset of the method of weighted residuals. In this case the weight functions are Dirac delta functions centered at the grid points. The finite difference stencils used in the present work can be obtained from fitting polynomials to the function values at grid points.

Define a grid \(x_{i-2}, x_{i-1}, x_i, x_{i+1}\) with associated function values \(u_{i-2}, u_{i-1}, u_i, u_{i+1}\). Polynomial theory states that there is only one unique polynomial of degree three which passes through these four points. It is the Lagrange polynomial given below.

\[
u(x) = \frac{u_{i-2}(x - x_{i-1})(x - x_i)(x - x_{i+1})}{(x_{i-2} - x_{i-1})(x_{i-2} - x_i)(x_{i-2} - x_{i+1})} + \frac{u_{i-1}(x - x_{i-2})(x - x_i)(x - x_{i+1})}{(x_{i-1} - x_{i-2})(x_{i-1} - x_i)(x_{i-1} - x_{i+1})} + \frac{u_i(x - x_{i-2})(x - x_{i-1})(x - x_{i+1})}{(x_{i-2} - x_{i-1})(x_{i-1} - x_i)(x_{i-1} - x_{i+1})} + \frac{u_{i+1}(x - x_{i-2})(x - x_{i-1})(x - x_i)}{(x_{i+1} - x_{i-2})(x_{i+1} - x_{i-1})(x_{i+1} - x_i)}\]

For clarity, an even grid (in which the spacing between grid points is constant throughout the grid) is defined by setting \(x_{i-2} = -2\Delta x, x_{i-1} = -\Delta x, x_i = 0, x_{i+1} = \Delta x\). Differentiating with respect to \(x\) results in:

\[
\frac{\partial u}{\partial x} = \frac{1}{6} \left[ \frac{-12u_{i-2}x\Delta x - 3u_{i-2}x^2 + 9u_{i-1}x^2 - 9u_i x^2 + 3u_i^2 \Delta x}{\Delta x^3} + \frac{3u_{i+1}x^2 + u_{i-2}^2 \Delta x - 6u_{i-1}^2 \Delta x - 2u_{i+1}^2 \Delta x + 6u_{i-1}x \Delta x + 2u_{i+1}x \Delta x}{\Delta x^3} \right]
\]

If this expression is evaluated at \(x = 0\) the result is:

\[
\frac{\partial u}{\partial x} = \frac{1}{6\Delta x} (u_{i-2} - 6u_{i-1} + 3u_i + 2u_{i+1})
\]

If this expression is evaluated at \(x = \Delta x\) the result is:

\[
\frac{\partial u}{\partial x} = \frac{1}{6\Delta x} (-2u_{i-2} + 9u_{i-1} - 18u_i + 11u_{i+1})
\]

If this expression is evaluated at \(x = \frac{\Delta x}{2}\) the result is:

\[
\frac{\partial u}{\partial x} = \frac{1}{\Delta x} (u_{i-2} + 27u_i - 27u_{i-1} - u_{i+1})
\]
These difference schemes can also be obtained by Taylor series expansions. Expanding at $x = 0$.

\[ u_{i+1} = u_i + \Delta x \frac{\partial u}{\partial x} + \frac{(\Delta x)^2}{2!} \frac{\partial^2 u}{\partial x^2} + \frac{(\Delta x)^3}{3!} \frac{\partial^3 u}{\partial x^3} + O((\Delta x)^4 + \ldots + (\Delta x)^n) \]  

(A.5)

\[ u_{i-1} = u_i - \Delta x \frac{\partial u}{\partial x} + \frac{(\Delta x)^2}{2!} \frac{\partial^2 u}{\partial x^2} - \frac{(\Delta x)^3}{3!} \frac{\partial^3 u}{\partial x^3} + O((-\Delta x)^4 + \ldots + (-\Delta x)^n) \]

\[ u_{i-2} = u_i - 2\Delta x \frac{\partial u}{\partial x} + \frac{(2\Delta x)^2}{2!} \frac{\partial^2 u}{\partial x^2} + \frac{(-2\Delta x)^3}{3!} \frac{\partial^3 u}{\partial x^3} + O((-2\Delta x)^4 + \ldots + (-2\Delta x)^n) \]

The solution to this set of linear equations for $\frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}, \frac{\partial^3 u}{\partial x^3}$ will result in the same expression for $\frac{\partial u}{\partial x}$ obtained above by differentiating the Lagrange polynomial which passes through these four points and evaluating it at $x = 0$. A benefit from deriving the expression through Taylor series expansions is that an explicit expression for the error is obtained. The error involved in expressing $\frac{\partial u}{\partial x}$ is:

\[ error = \frac{-2O((\Delta x)^4 + \ldots + (\Delta x)^n) + 6O((-\Delta x)^4 + \ldots + (-\Delta x)^n)}{\Delta x} \]

If the first two error terms are expanded out the following expression is obtained for the error.

\[ error = \frac{-2}{\Delta x} \left( \frac{(\Delta x)^4}{4!} \frac{\partial^4 u}{\partial x^4} + \frac{(\Delta x)^5}{5!} \frac{\partial^5 u}{\partial x^5} \right) + \frac{6}{\Delta x} \left( \frac{(-\Delta x)^4}{4!} \frac{\partial^4 u}{\partial x^4} + \frac{(-\Delta x)^5}{5!} \frac{\partial^5 u}{\partial x^5} \right) \]

\[ - \frac{1}{\Delta x} \left( \frac{(-2\Delta x)^4}{4!} \frac{\partial^4 u}{\partial x^4} + \frac{(-2\Delta x)^5}{5!} \frac{\partial^5 u}{\partial x^5} \right) \]

\[ error = \frac{-12}{4!} \frac{(\Delta x)^3}{\partial x^4} + \frac{26}{5!} \frac{(\Delta x)^4}{\partial x^4} + \frac{(-2\Delta x)^3}{5!} \frac{\partial^5 u}{\partial x^5} \]

It can be seen that the leading order error term is diffusive as it contains an even derivative.

This diffusivity helps to stabilize Eq. 2.1 and Eq. 2.3 although accuracy will be compromised if it is of the same magnitude as the physical diffusion. On an uneven grid, i.e. a grid in which the distance between grid points varies, the resulting expressions for derivatives are much more complicated in terms of the grid spacing. In general, each coefficient for a function value will depend on the grid spacing of all the function values. This is not a problem in practice as the coefficients for the function values can be obtained to machine accuracy using Eq. A.5. In
the present work, the coefficients for the function values in one dimension are obtained by the
algorithm described in (Fornberg, 1988). The algorithm described in (Fornberg, 1988) is reputed
to be an efficient method although this has not been verified by the author. The efficiency is not of
concern in the present work, as the grid does not change with time and therefore the coefficients
for the function values can be computed and stored in a preprocessing stage.

A.2 Two Dimensions

While Eq. 2.1, Eq. 2.2 and Eq. 2.3 do not contain mixed derivatives, mixed derivatives are
needed when computing certain terms in the exact $\varepsilon$ and $\varepsilon'$ equations (Eq. 6.5 and Eq. 6.4).

A.2.0.1 Polynomial Fitting

Consider a collection of $N$ points in three dimensions denoted by

$$x_{i,j},$$

where $i$ denotes the dimension and $j$ the point

It is desired to define a method by which to interpolate a scalar $u$ defined at this collection of
points. One method is to create a function defined as follows:

$$U(x) = \sum_{i=1}^{M} a_i f_i(x) \text{ where the } f_i(x) \text{ are unspecified basis functions} \quad (A.6)$$

If polynomials are chosen as basis functions the result can be written as Eq. A.7:

$$U(x) = \sum_{k=0}^{\beta_3} \sum_{m=0}^{\beta_2} \sum_{n=0}^{\beta_1} a_{[1+n(m+1)(k+1)+m(k+1)+k]} x_{1,j}^n x_{2,j}^m x_{3,j}^k \quad (A.7)$$

(this equation requires that $0^0$ be defined as one)

Expanding for the case of $\beta_3 = 0, \beta_2 = \beta_1 = 2$.

$$U(x) = a_1 + a_2 x_{1,j} + a_3 x_{1,j}^2 + a_4 x_{2,j} + a_5 x_{1,j} x_{2,j} + a_6 x_{1,j}^2 x_{2,j} + a_7 x_{2,j}^2 + a_8 x_{1,j} x_{2,j}^2 + a_9 x_{1,j}^2 x_{2,j}^2 \quad (A.8)$$

$U(x)$ can be defined as the following (which incidentally cannot be obtained directly from Eq.
A.7):

$$U(x) = a_1 + a_2 x_{1,j} + a_3 x_{1,j} + a_4 x_{1,j}^2 + a_5 x_{2,j}^2 + a_6 x_{1,j} x_{2,j} \quad (A.9)$$

A system of six equations (Eq. A.10) can be formed by setting Eq. A.9 equal to the value of $u$ at
Each grid point.

\[
\begin{pmatrix}
1 & x_{1,1} & x_{1,2} & x_{1,3} & x_{1,4} & x_{1,5} & x_{1,6} \\
1 & x_{2,1} & x_{2,2} & x_{2,3} & x_{2,4} & x_{2,5} & x_{2,6} \\
1 & x_{3,1} & x_{3,2} & x_{3,3} & x_{3,4} & x_{3,5} & x_{3,6} \\
1 & x_{4,1} & x_{4,2} & x_{4,3} & x_{4,4} & x_{4,5} & x_{4,6} \\
1 & x_{5,1} & x_{5,2} & x_{5,3} & x_{5,4} & x_{5,5} & x_{5,6} \\
1 & x_{6,1} & x_{6,2} & x_{6,3} & x_{6,4} & x_{6,5} & x_{6,6}
\end{pmatrix}
\begin{pmatrix}
a_1 \\
a_2 \\
a_3 \\
a_4 \\
a_5 \\
a_6
\end{pmatrix}
= \begin{pmatrix}
u_1 \\
u_2 \\
u_3 \\
u_4 \\
u_5 \\
u_6
\end{pmatrix}
= F a = u \quad \text{(A.10)}
\]

Eq. A.10 can be written compactly as:

\[
U(x) = \sum_{i=1}^{N} a_i f_i(x) = \sum_{i=1}^{N} (F^{-1} u)_i f_i(x)
\]

Expanding

\[
U(x) = \begin{pmatrix}
F_{1,1}^{-1} u_1 + F_{1,2}^{-1} u_2 + F_{1,3}^{-1} u_3 + \ldots + F_{1,N}^{-1} u_N \\
F_{2,1}^{-1} u_1 + F_{2,2}^{-1} u_2 + F_{2,3}^{-1} u_3 + \ldots + F_{2,N}^{-1} u_N \\
F_{3,1}^{-1} u_1 + F_{3,2}^{-1} u_2 + F_{3,3}^{-1} u_3 + \ldots + F_{3,N}^{-1} u_N \\
\vdots \\
F_{N,1}^{-1} u_1 + F_{N,2}^{-1} u_2 + F_{N,3}^{-1} u_3 + \ldots + F_{N,N}^{-1} u_N
\end{pmatrix}
^T
\begin{pmatrix}
f_1(x) \\
f_2(x) \\
f_3(x) \\
\vdots \\
f_N(x)
\end{pmatrix}
= \begin{pmatrix}
F_{1,1}^{-1} u_1 f_1(x) + F_{1,2}^{-1} u_2 f_1(x) + \ldots + F_{1,N}^{-1} u_N f_1(x) \\
F_{2,1}^{-1} u_1 f_2(x) + F_{2,2}^{-1} u_2 f_2(x) + \ldots + F_{2,N}^{-1} u_N f_2(x) \\
\vdots \\
F_{N,1}^{-1} u_1 f_N(x) + F_{N,2}^{-1} u_2 f_N(x) + \ldots + F_{N,N}^{-1} u_N f_N(x)
\end{pmatrix}
\]

Which can be written as:

\[
U(x) = \sum_{j=1}^{N} \left[ \sum_{i=1}^{N} F_{i,j}^{-1} f_i(x) \right] u_j \quad \text{(A.11)}
\]
The result (Eq. A.11) is an interpolation function. This function can be used in various ways. One way is as a finite element shape function. A unique solution exists if the determinant of \( F \neq 0 \). This imposes a restriction on the location of the grid points. A further restriction comes from the practical requirement that \( F \) not be ill-conditioned. Note that \( \sum_{i,j=1}^{N} F_{i,j}^{-1} f_i(x) \) in Eq. A.11 is a function only of the grid.

### A.2.0.2 Taylor Series Expansion

Consider two unique points in three-dimensional space \( \vec{x}_1, \vec{x}_2 \). The derivative of a scalar \( f \) at the point \( \vec{x}_1 \) in the direction \( v \) is given by the following:

\[
D_v(f) = \nabla f \cdot v = f_i v_i \quad \text{where} \quad v = \frac{\vec{x}_2 - \vec{x}_1}{\| \vec{x}_2 - \vec{x}_1 \|} \quad \text{and} \quad f_i \text{ is the derivative of } f \text{ in the } i \text{ direction}
\]

Likewise, the second derivative of a scalar \( f \) at the point \( \vec{x}_1 \) in the direction \( v \) is given by the following:

\[
D_{vv}(f) = D_v[D_v(f)] = \nabla (f_i v_i) \cdot v = (f_{i,j} v_i) v_j
\]

where \( f_{i,j} \) is the derivative of \( f \) in the \( i \) and then the \( j \) direction.

Using the above definitions, a Taylor series expansion can be written about the point \( \vec{x}_1 \).

\[
f(\vec{x}_2) = f(\vec{x}_1) + d D_v(f) + \frac{d^2}{2!} D_{vv}(f) + O(d^3) \quad \text{where} \quad d = \| \vec{x}_1 - \vec{x}_2 \|
\]

\[
f(\vec{x}_2) = f(\vec{x}_1) + df_i v_i + \frac{d^2}{2!} (f_{i,j} v_i) v_j + O(d^3)
\]

Expanding results in:

\[
f(\vec{x}_2) = f(\vec{x}_1) + d (f_1 v_1 + f_2 v_2 + f_3 v_3) + d^2 \left( \frac{1}{2} v_1^2 f_{1,1} + \frac{1}{2} v_2^2 f_{2,2} + \frac{1}{2} v_3^2 f_{3,3} + v_1 v_2 f_{1,2} + v_1 v_3 f_{1,3} + v_2 v_3 f_{2,3} \right)
\]

There are nine unknowns (the derivatives) in Eq. A.12. If Eq. A.12 is written for nine points then the possibility exists of solving the resulting system of equations for the derivatives. If difference
expressions are desired in two dimensions then the resulting set of equations is given by:

\[
f(x_i) = f(x_1) + d(f_{1v_{1,i}} + f_{2v_{2,i}}) + d^2 \left( \frac{1}{2} v_{1,i}^2 f_{1,1} + \frac{1}{2} v_{2,i}^2 f_{2,2} + v_{1,i} v_{2,i} f_{1,2} \right) \quad i \in 2, 6
\]

In matrix form:

\[
\begin{pmatrix}
  d_{2v_{1,2}} & d_{2v_{2,2}} & \frac{1}{2} d_{2v_{1,2}}^2 & \frac{1}{2} d_{2v_{2,2}}^2 & d_{2v_{1,2}v_{2,2}} \\
  d_{3v_{1,3}} & d_{3v_{2,3}} & \frac{1}{2} d_{3v_{1,3}}^2 & \frac{1}{2} d_{3v_{2,3}}^2 & d_{3v_{1,3}v_{2,3}} \\
  d_{4v_{1,4}} & d_{4v_{2,4}} & \frac{1}{2} d_{4v_{1,4}}^2 & \frac{1}{2} d_{4v_{2,4}}^2 & d_{4v_{1,4}v_{2,4}} \\
  d_{5v_{1,5}} & d_{5v_{2,5}} & \frac{1}{2} d_{5v_{1,5}}^2 & \frac{1}{2} d_{5v_{2,5}}^2 & d_{5v_{1,5}v_{2,5}} \\
  d_{6v_{1,6}} & d_{6v_{2,6}} & \frac{1}{2} d_{6v_{1,6}}^2 & \frac{1}{2} d_{6v_{2,6}}^2 & d_{6v_{1,6}v_{2,6}} \\
\end{pmatrix}
\begin{pmatrix}
  f_1 \\
  f_2 \\
  f_{11} \\
  f_{22} \\
  f_{12} \\
\end{pmatrix} =
\begin{pmatrix}
  f(x_2) - f(x_1) \\
  f(x_3) - f(x_1) \\
  f(x_4) - f(x_1) \\
  f(x_5) - f(x_1) \\
  f(x_6) - f(x_1) \\
\end{pmatrix}
\]

(A.13)

A unique solution exists if the determinant of the matrix in Eq. A.13 \( \neq 0 \). This imposes a restriction on the location of the grid points. A further restriction comes from the practical requirement that the matrix in Eq. A.13 not be ill-conditioned. If the coordinate system origin is set at the point \( x_1 \) in Eq. A.10 then Eq. A.10 reduces to:

\[
\begin{pmatrix}
  1 & 0 & 0 & 0 & 0 \\
  1 & x_{1,2} & x_{2,2} & x_{1,2} x_{1,2} & x_{2,2} x_{2,2} \\
  1 & x_{1,3} & x_{2,3} & x_{1,3} x_{1,3} & x_{2,3} x_{2,3} \\
  1 & x_{1,4} & x_{2,4} & x_{1,4} x_{1,4} & x_{2,4} x_{2,4} \\
  1 & x_{1,5} & x_{2,5} & x_{1,5} x_{1,5} & x_{2,5} x_{2,5} \\
  1 & x_{1,6} & x_{2,6} & x_{1,6} x_{1,6} & x_{2,6} x_{2,6} \\
\end{pmatrix}
\begin{pmatrix}
  a_1 \\
  a_2 \\
  a_3 \\
  a_4 \\
  a_5 \\
  a_6 \\
\end{pmatrix} =
\begin{pmatrix}
  u_1 \\
  u_2 \\
  u_3 \\
  u_4 \\
  u_5 \\
  u_6 \\
\end{pmatrix}
\]

Which can be rewritten as:

\[
\begin{pmatrix}
  x_{1,2} & x_{2,2} & x_{1,2} x_{1,2} & x_{2,2} x_{2,2} \\
  x_{1,3} & x_{2,3} & x_{1,3} x_{1,3} & x_{2,3} x_{2,3} \\
  x_{1,4} & x_{2,4} & x_{1,4} x_{1,4} & x_{2,4} x_{2,4} \\
  x_{1,5} & x_{2,5} & x_{1,5} x_{1,5} & x_{2,5} x_{2,5} \\
  x_{1,6} & x_{2,6} & x_{1,6} x_{1,6} & x_{2,6} x_{2,6} \\
\end{pmatrix}
\begin{pmatrix}
  a_2 \\
  a_3 \\
  a_4 \\
  a_5 \\
  a_6 \\
\end{pmatrix} =
\begin{pmatrix}
  u_2 - u_1 \\
  u_3 - u_1 \\
  u_4 - u_1 \\
  u_5 - u_1 \\
  u_6 - u_1 \\
\end{pmatrix}
\]

Now since \( d_{iv_i} = x_{j+1} - x_1 = x_{j+1} \)

\[
\begin{pmatrix}
  d_{2v_{1,2}} & d_{2v_{2,2}} & d_{2v_{1,2}}^2 & d_{2v_{2,2}}^2 & d_{2v_{1,2}v_{2,2}} \\
  d_{3v_{1,3}} & d_{3v_{2,3}} & d_{3v_{1,3}}^2 & d_{3v_{2,3}}^2 & d_{3v_{1,3}v_{2,3}} \\
  d_{4v_{1,4}} & d_{4v_{2,4}} & d_{4v_{1,4}}^2 & d_{4v_{2,4}}^2 & d_{4v_{1,4}v_{2,4}} \\
  d_{5v_{1,5}} & d_{5v_{2,5}} & d_{5v_{1,5}}^2 & d_{5v_{2,5}}^2 & d_{5v_{1,5}v_{2,5}} \\
  d_{6v_{1,6}} & d_{6v_{2,6}} & d_{6v_{1,6}}^2 & d_{6v_{2,6}}^2 & d_{6v_{1,6}v_{2,6}} \\
\end{pmatrix}
\begin{pmatrix}
  a_2 \\
  a_3 \\
  a_4 \\
  a_5 \\
  a_6 \\
\end{pmatrix} =
\begin{pmatrix}
  u_2 - u_1 \\
  u_3 - u_1 \\
  u_4 - u_1 \\
  u_5 - u_1 \\
  u_6 - u_1 \\
\end{pmatrix}
\]

This can now be seen to be the same expression as Eq. A.13 with \( \frac{dy}{x} = f_{11} \) and \( \frac{dy}{y} = f_{22} \). If a mixed derivative is desired, it can be obtained by differentiating Eq. A.11. As the origin of the coordinate system used in Eq. A.11 can be set at whatever point the mixed derivative is desired, the value of the mixed derivative (or other derivatives) is simply the value of the coefficient of the
appropriate term in Eq. A.11. This method (using Eq. A.11) is used in the present work in order to determine the weights in the finite difference stencils used to represent the mixed derivative terms which appear in the exact $k$ and $\varepsilon$ equations (Eq. 6.4 and Eq. 6.5).

A.3 Implemented Discretization Schemes in Tetra

Eq. 2.1 and Eq. 2.2 are solved on a staggered grid (see Fig. A.1). Note that the three components of velocity are at different spatial locations. If the computational domain is composed of a collection of Cartesian blocks that do not form a single Cartesian block, then the main grid points are centered between adjacent velocity locations. If this is not the case, the velocity is centered instead between the two adjacent pressure locations. Pressure is defined on main grid points, each interior point of which is surrounded by six velocity points. Pressure is not defined on non-periodic boundaries. The equations governing the $x$, $y$ and $z$ components of momentum are solved at the same grid points at which the $x$, $y$ and $z$ components of velocity are defined, respectively. The equation governing the conservation of mass is solved at the same grid points at which pressure is defined. The scalar is defined on the same grid on which the pressure is defined, with the exception that the scalar is defined on non-periodic boundaries (see Fig. A.2). The equation governing the evolution of the passive scalar (Eq. 2.3) is solved at the same locations at which the scalar is defined.

A.3.1 Continuity

The terms in the continuity equation are represented by either a two, four or six point centered stencil. The stencils below are shown for the particular case of an even grid. For the case of an uneven grid, the weights for the function values at the discrete grid points are determined by differentiating the unique Lagrange polynomial that passes through the points and evaluating the result at the location at which the derivative is desired.
Figure A.1: Staggered grid schematic

Figure A.2: Grid locations of scalar
\[
\frac{\partial u}{\partial x}\bigg|_i = \frac{1}{\Delta x} (u_i - u_{i-1}) + O(\Delta x^2)
\] (A.14)

\[
\frac{\partial u}{\partial x}\bigg|_i = \frac{1}{\Delta x} (u_{i-2} + 27u_i - 27u_{i-1} - u_{i+1}) + O(\Delta x^4)
\] (A.15)

\[
\frac{\partial u}{\partial x}\bigg|_i = \frac{1}{\Delta x} \left( 17280u_{i-3} + 240000u_{i-2} + 4320000u_i - 4320000u_{i-1} - 240000u_{i+1} - 17280u_{i+2} \right) + O(\Delta x^6)
\] (A.16)

The order of accuracy indicated assumes the grid is even; otherwise it is not possible to define an order of accuracy. The order of the stencils is retained as a non-periodic boundary is approached. In other words, at a non-periodic boundary using the sixth-order scheme, the following points would be used.

\[
\frac{\partial u}{\partial x}\bigg|_2 = a_1u_1 + a_2u_2 + a_3u_3 + a_4u_4 + a_5u_5 + a_6u_6 \text{ where } u_1 \text{ is on the boundary}
\]

The coefficients \(a_1, a_2, \ldots\) are obtained by differentiating the unique fifth-order Lagrange polynomial and evaluating the result at the location where \(\frac{\partial u}{\partial x}\) is needed. This of course results in a non-centered stencil. The option of maintaining a centered stencil at a non-periodic boundary is included in the computer program (Tetra) used in the present work. This will require a reduction in the number of points used and therefore of the formal order of accuracy of the representation of the derivative. This reduction in accuracy has not been used in any of the results presented in the present work.

A.3.2 Pressure Gradient

The pressure gradient terms in the momentum equations are represented by either a two, four or six point centered stencil. The stencils below are shown for the particular case of an even grid. For the case of an uneven grid, the weights for the function values at the discrete grid points are determined by differentiating the unique Lagrange polynomial that passes through the points and
evaluating the result at the location where the derivative is desired.

\[
\frac{\partial p}{\partial x} \bigg|_{i} = \frac{1}{\Delta x} (p_{i+1} - p_{i}) + O(\Delta x^2) \quad (A.17)
\]

\[
\frac{\partial p}{\partial x} \bigg|_{i} = \frac{1}{\Delta x} (p_{i-1} + 27p_{i+1} - 27p_{i} - p_{i+2}) + O(\Delta x^4) \quad (A.18)
\]

\[
\frac{\partial p}{\partial x} \bigg|_{i} = 10 \left( \frac{1728p_{i-2} + 24000p_{i-1} + 432000p_{i+1} - 432000p_{i} - 24000p_{i+2} - 1728p_{i+3}}{\Delta x} \right) + O(\Delta x^6) \quad (A.19)
\]

The order of accuracy indicated assumes the grid is even. As regards boundary conditions the same treatment is used as for the continuity equation. The only difference is that pressure is not defined on non-periodic boundaries (see Fig. A.1).

### A.3.3 Diffusive Terms

The diffusive terms in the momentum and scalar equations are represented by either a three, five or seven point centered stencil. The stencils below are shown for the particular case of an even grid. For the case of an uneven grid, the weights for the function values at the discrete grid points are determined by differentiating the unique Lagrange polynomial that passes through the points and evaluating the result at the location where the derivative is desired.

\[
\frac{\partial^2 u}{\partial x^2} \bigg|_{i} = \frac{1}{\Delta x^2} (u_{i-1} - 2u_{i} + u_{i+1}) + O(\Delta x^2) \quad (A.20)
\]

\[
\frac{\partial^2 u}{\partial x^2} \bigg|_{i} = \frac{1}{12\Delta x^2} (-u_{i-2} + 16u_{i-1} - 30u_{i} + 16u_{i+1} - u_{i+2}) + O(\Delta x^4) \quad (A.21)
\]

\[
\frac{\partial^2 u}{\partial x^2} \bigg|_{i} = \frac{1}{180\Delta x^2} \left( \frac{2u_{i-3} - 27u_{i-2} + 270u_{i-1} - 490u_{i} + 270u_{i+1} - 27u_{i+2} + 2u_{i+3}}{\Delta x^2} \right) + O(\Delta x^6) \quad (A.22)
\]

The order of accuracy indicated assumes the grid is even. As regards boundary conditions the same treatment is used as for the continuity equation.

### A.3.4 Convective Terms

The nonlinear convective terms in the momentum and scalar equations are evaluated in two parts - the derivatives, and the velocities that multiply them.
A.3.4.1 Interpolation of the Velocity for Momentum Equations

Due to the staggered grid, none of the velocity components are stored at the same location. However they are needed at the same location in order to compute the parts of $u_j \frac{\partial u_i}{\partial x_j}$ where $i \neq j$ in Eq. 2.1. They are obtained by fitting a two-dimensional centered Lagrange surface over the point in question (see Fig. A.3). The fitting uses products of two, four and six point Lagrange polynomials, which results in second, fourth and sixth-order accurate interpolation stencils. This results in a total number of points involved in the fitting of the surface of four, sixteen, and thirty-six respectively. The number of points involved is retained as non-periodic boundaries are approached; this results in a non-centered interpolation stencil. The resulting non-centered stencil is shown in Fig. A.4 for the case of a fourth-order interpolation stencil.
A.3.4.2 Interpolation of the Velocity for Scalar Equation

It is necessary to obtain the velocity at the main grid points (see Fig. A.2), in order to evaluate the convective terms in Eq. 2.3. This is done by interpolating $u$ in the $x$ direction, $v$ in the $y$ direction, and $w$ in the $z$ direction. The interpolation uses centered Lagrange polynomials involving two, four or six points. The order of the polynomial is maintained as non-periodic boundaries are approached; this results in a non-centered interpolation stencil. The resulting non-centered stencil is shown in Fig. A.5. This interpolation scheme is also used to obtain a velocity field needed for post-processing and for computing quantities involving derivatives, such as vorticity and certain correlations appearing in the exact $k$ and $\varepsilon$ equations (Eq. 6.4 and Eq. 6.5).
A.3.4.3 Upwind Schemes

In the computer program used in the present work, two, three, four, five and six point upwind biased stencils can be used for the convective terms. There are two types (resulting from different amounts of biasing in the upwind direction) of four, five and six point schemes. The stencils and their orders of accuracy are shown below for the case of upwind biased schemes (i.e. assuming that the flow is in the direction of increasing $i$ index). The coefficients in the stencils can be obtained by the polynomial fitting methods described previously. As regards boundary conditions, the same treatment is used as for the continuity equation. If monotonic limiters are not used, then the most commonly used scheme in the present work is Eq. A.25. This scheme is used primarily for reasons of stability as the higher-order schemes are more unstable.
These higher-order schemes can be made stable by the use of a monotonic limiter; however, if this approach is taken then central difference schemes, which are used in the present work in conjunction with monotonic limiters, are more efficient.

\[
\begin{align*}
\frac{\partial u}{\partial x} |_{i,j,k} &= a_1 u_{i-1,j,k} + a_2 u_{i,j,k} + O(\Delta x^1) \\
\frac{\partial u}{\partial x} |_{i,j,k} &= a_1 u_{i-2,j,k} + a_2 u_{i-1,j,k} + a_3 u_{i,j,k} + O(\Delta x^2) \\
\frac{\partial u}{\partial x} |_{i,j,k} &= a_1 u_{i-2,j,k} + a_2 u_{i-1,j,k} + a_3 u_{i,j,k} + a_4 u_{i+1,j,k} + O(\Delta x^3) \\
\frac{\partial u}{\partial x} |_{i,j,k} &= a_1 u_{i-3,j,k} + a_2 u_{i-2,j,k} + a_3 u_{i-1,j,k} + a_4 u_{i,j,k} + O(\Delta x^3) \\
\frac{\partial u}{\partial x} |_{i,j,k} &= a_1 u_{i-3,j,k} + a_2 u_{i-2,j,k} + a_3 u_{i-1,j,k} + a_4 u_{i,j,k} + a_5 u_{i+1,j,k} + O(\Delta x^4) \\
\frac{\partial u}{\partial x} |_{i,j,k} &= a_1 u_{i-4,j,k} + a_2 u_{i-3,j,k} + a_3 u_{i-2,j,k} + a_4 u_{i-1,j,k} + a_5 u_{i,j,k} + O(\Delta x^4) \\
\frac{\partial u}{\partial x} |_{i,j,k} &= a_1 u_{i-3,j,k} + a_2 u_{i-2,j,k} + a_3 u_{i-1,j,k} + a_4 u_{i,j,k} + a_5 u_{i+1,j,k} + a_6 u_{i+2,j,k} + O(\Delta x^5) \\
\frac{\partial u}{\partial x} |_{i,j,k} &= a_1 u_{i-4,j,k} + a_2 u_{i-3,j,k} + a_3 u_{i-2,j,k} + a_4 u_{i-1,j,k} + a_5 u_{i,j,k} + a_6 u_{i+1,j,k} + a_7 u_{i+2,j,k} + O(\Delta x^5)
\end{align*}
\]

The upwinding is implemented in software in one of two ways. Shown below are two ways upwinding can be implemented for Eq. A.25 on an even grid. The \textit{sign} function is a Fortran function that returns the absolute value of the first argument with the sign of the second argument.

\[
\begin{align*}
u_{i,j,k} \frac{\partial u}{\partial x} |_{i,j,k} &= \left[ (.5 + \text{sign}(.5, u_{i,j,k})) \left( \frac{u_{i,j,k}}{\Delta x} (u_{i-2,j,k} - 6u_{i-1,j,k} + 3u_{i,j,k} + 2u_{i+1,j,k}) \right) \\
&\quad + (.5 - \text{sign}(.5, u_{i,j,k})) \left( \frac{u_{i,j,k}}{\Delta x} (-2u_{i-1,j,k} - 3u_{i,j,k} + 6u_{i+1,j,k} - u_{i+2,j,k}) \right) \right]
\end{align*}
\]
\[ \text{if}(u_{i,j,k} \geq 0)\text{then} \]

\[ u_{i,j,k} \frac{\partial u}{\partial x} \bigg|_{i,j,k} = u_{i,j,k} \left( \frac{1}{6\Delta x} \left( u_{i-2,j,k} - 6u_{i-1,j,k} + 3u_{i,j,k} + 2u_{i+1,j,k} \right) \right) \]

\[ \text{else} \]

\[ u_{i,j,k} \frac{\partial u}{\partial x} \bigg|_{i,j,k} = u_{i,j,k} \left( \frac{1}{6\Delta x} \left( -2u_{i-1,j,k} - 3u_{i,j,k} + 6u_{i+1,j,k} - u_{i+2,j,k} \right) \right) \]

\[ \text{endif} \]

The first method involves more floating point operations but does not have a conditional branch. Which method is faster will depend on the machine used. The reason for this is that the second method means that the path of instructions executed by the computer depends on the solution. This causes delays because, until the if statement is evaluated, the computer cannot know which branch of the code to take. On a vector machine, such as the Cray C90 or T90, the conditional branch caused by the if statement will prevent the loop from vectorizing and thereby seriously degrade performance. If the machine is not a vector machine, then the first method may or may not be faster. For example, the first method is faster on the IBM Power3 processor. This processor is capable of four floating point instructions per clock cycle. The second method is faster on the Intel Pentium III processor. This processor is capable of only one floating point instruction per clock cycle, so reducing the amount of floating point operations is more critical for improving performance than on the IBM Power3 processor.

**A.3.4.4 Centered Schemes**

In the computer program used in the present work, three, five or seven point centered stencils can be used for the convective terms. On an even grid the order of accuracy is \(O(\Delta x^2), O(\Delta x^4), O(\Delta x^6)\) respectively. The stencils are given below. As regards boundary conditions, the same treatment is used as for the continuity equation. As no upwinding is done, fewer floating point operations are required than an upwind stencil with the same order of
\[
\frac{\partial u}{\partial x} |_{i,j,k} = a_1 u_{i-1,j,k} + a_2 u_{i,j,k} + a_3 u_{i+1,j,k} + O(\Delta x^2) \quad (A.33)
\]
\[
\frac{\partial u}{\partial x} |_{i,j,k} = a_1 u_{i-2,j,k} + a_2 u_{i-1,j,k} + a_3 u_{i,j,k} + a_4 u_{i+1,j,k} + a_5 u_{i+2,j,k} + O(\Delta x^4) \quad (A.34)
\]
\[
\frac{\partial u}{\partial x} |_{i,j,k} = a_1 u_{i-3,j,k} + a_2 u_{i-2,j,k} + a_3 u_{i-1,j,k} + a_4 u_{i,j,k} + a_5 u_{i+1,j,k} + a_6 u_{i+2,j,k} + a_7 u_{i+3,j,k} + O(\Delta x^6) \quad (A.35)
\]

### A.4 Time Integration Schemes

#### A.4.1 Multi Level Methods

After spatially discretizing the momentum equations (Eq. 2.1), the result is a system of coupled ordinary differential equations subject to the constraint that the divergence is zero. A general scheme to integrate Eq. 2.1 in time can be written as:

\[
u^{n+1} = \int_{t^n}^{t^{n+1}} (C(t) + D(t) + P(t)) \, dt + u^n,
\]

where \( C(t) \) represents the spatially discretized convection term, \( D(t) \) the spatially discretized diffusive term and \( P(t) \) the spatially discretized pressure gradient term. In the above equation, there are many possibilities for \( C(t) \), \( D(t) \) and \( P(t) \). These can be defined by assuming a polynomial form and an associated set of constraints to define the coefficients of the polynomial. Defining the origin of \( t \) to be at the \( n \) time level and assuming the following form for \( C(t) \) along with the associated set of constraints, a quadratic scheme can be written as:

\[
C(t) = a_1 + a_2 t + a_3 t^2
\]
\[
C^{n-1} = a_1 - a_2 \Delta t + a_3 (-\Delta t)^2
\]
\[
C^n = a_1 + a_2 0 + a_3 0^2
\]
\[
C^{n+1} = a_1 + a_2 \Delta t + a_3 (\Delta t)^2
\]
The solution of which is:

\[ a_1 = C^n, \quad a_2 = \frac{C^{n+1} - C^{n-1}}{2\Delta t}, \quad a_3 = \frac{C^{n-1} + C^{n+1} - 2C^n}{2(\Delta t)^2} \]

Carrying out the integral results in the following discrete time integration scheme:

\[
\int_0^\Delta t \left( C^n + \frac{C^{n+1} - C^{n-1}}{2\Delta t} t + \frac{C^{n-1} + C^{n+1} - 2C^n}{2(\Delta t)^2} t^2 \right) dt
= \frac{2}{3} C^n \Delta t + \frac{5}{12} \Delta t C^{n+1} - \frac{1}{12} \Delta t C^{n-1}
\]

Note that it is an implicit scheme as \( u^{n+1} \) depends on \( C^{n+1} \), which is itself a function of \( u^{n+1} \).

An explicit scheme can be defined by using a different set of constraints. Assuming the following form for \( C(t) \):

\[ C(t) = a_1 + a_2 t \]

a linear explicit scheme can be written as:

\[
C^{n-1} = a_1 - a_2 \Delta t
\]

\[ C^n = a_1 + a_2 t_0 \]

The solution of which is:

\[ a_1 = C^n, \quad a_2 = \frac{C^n - C^{n-1}}{\Delta t} \]

Carrying out the integral results in the following discrete time integration scheme:

\[
\int_0^\Delta t \left( C^n + \frac{C^n - C^{n-1}}{\Delta t} t \right) dt = \left( C^n t + \frac{C^n - C^{n-1}}{\Delta t} t^2 \right]\bigg|_0^\Delta t = \frac{\Delta t}{2} \left( -C^{n-1} + 3C^n \right)
\]

It is an explicit scheme in that \( u^{n+1} \) does not depend on \( C^{n+1} \), which is itself a function of \( u^{n+1} \).

Note that the region over which the polynomial is integrated is outside the region in which the constraints were set. This amounts to using the polynomial as an extrapolant instead of as an interpolant.
The following multi level time integration schemes have been implemented in Tetra.

\[
\int_0^{\Delta t} \beta(t) dt = \beta^n \Delta t \quad (A.37)
\]
\[
\int_0^{\Delta t} \beta(t) dt = \beta^{n+1} \Delta t \quad (A.38)
\]
\[
\int_0^{\Delta t} \beta(t) dt = \frac{\Delta t}{2} (-\beta^{n-1} + 3\beta^n) \quad (A.39)
\]
\[
\int_0^{\Delta t} \beta(t) dt = \frac{\Delta t}{12} (-\beta^{n-1} + 8\beta^n + 5\beta^{n+1}) \quad (A.40)
\]
\[
\int_0^{\Delta t} \beta(t) dt = \frac{\Delta t}{12} (5\beta^{n-2} - 16\beta^{n-1} + 23\beta^n) \quad (A.41)
\]
\[
\int_0^{\Delta t} \beta(t) dt = \frac{\Delta t}{24} (\beta^{n-2} - 5\beta^{n-1} + 19\beta^n + 9\beta^{n+1}) \quad (A.42)
\]

It is not necessary for the convection, diffusion and pressure gradient terms to be integrated with the same time integration scheme. Specifically, as there is no time derivative of pressure in the governing equations (Eq. 2.2 and Eq. 2.1), it would appear unreasonable to introduce the effect of pressure at previous time steps by using a multi level scheme that did so. For this reason, Eq. A.38 is used for the pressure gradient term in Eq. 2.1 in the present work.

**A.4.2 Discretization of the Time Derivative Term**

Instead of integrating Eq. 2.1 in time, it is possible to discretize the time derivative in these equations, using a backward-in-time discretization stencil. The pressure gradient, convection and diffusion terms are then evaluated at the current time level. As a result the schemes are implicit.

\[
\frac{\partial u}{\partial t} = \frac{u^{n+1} - u^n}{\Delta t} \quad \text{leading truncation error term is } O(\Delta t) \quad (A.43)
\]
\[
\frac{\partial u}{\partial t} = \frac{3u^{n+1} - 4u^n + u^{n-1}}{2\Delta t} \quad \text{leading truncation error term is } O(\Delta t^2)
\]
\[
\frac{\partial u}{\partial t} = \frac{11u^{n+1} - 18u^n + 9u^{n-1} - 2u^{n-2}}{6\Delta t} \quad \text{leading truncation error term is } O(\Delta t^3)
\]

Applying these schemes to the momentum equations (Eq. 2.1) results in:

\[
\frac{u^{n+1} - u^n}{\Delta t} = -\frac{\partial p}{\partial x_i}^{n+1} + \frac{1}{Re} \frac{\partial^2 u_i}{\partial x_j \partial x_j}^{n+1} - u_j \frac{\partial u_i}{\partial x_j}^{n+1} \quad (A.44)
\]
\[
\frac{3u_i^{n+1} - 4u_i^n + u_i^{n-1}}{2\Delta t} = -\frac{\partial p^{n+1}}{\partial x_i} + \frac{1}{Re} \frac{\partial^2 u_i^{n+1}}{\partial x_j \partial x_j} - u_j \frac{\partial u_i^{n+1}}{\partial x_j}
\] (A.45)

\[
\frac{11u_i^{n+1} - 18u_i^n + 9u_i^{n-1} - 2u_i^{n-2}}{6\Delta t} = -\frac{\partial p^{n+1}}{\partial x_i} + \frac{1}{Re} \frac{\partial^2 u_i^{n+1}}{\partial x_j \partial x_j} - u_j \frac{\partial u_i^{n+1}}{\partial x_j}
\] (A.46)
Appendix B  Exact Turbulent Kinetic Energy and Dissipation Equations

B.1  Exact \( k \) Equation

The exact \( k \) equation (where \( k \) is the turbulent kinetic energy i.e. \( \frac{1}{2}u_i^l u_j^l \)) is given by Eq. 6.4.

B.1.1  Computing as a Running Average

Expanding each term involving fluctuating components in the exact \( k \) equation results in the following expressions.

\[
\overline{u_i^l u_j^l} = (u_i - \overline{u_i})(u_j - \overline{u_j}) = \overline{u_i u_j} - \overline{u_i}u_j - u_i \overline{u_j} + u_i u_j = \overline{u_i u_j} - u_i \overline{u_j}
\]

\[
\overline{u_i^l u_j^l u_k} = (u_i - \overline{u_i})(u_k - \overline{u_k}) = \overline{u_i u_k} - \overline{u_i}u_k - \overline{u_k}u_i + \overline{u_i u_k} = \overline{u_i u_k} - u_i \overline{u_k}
\]

\[
\overline{u_i^l u_j^l u_k} = (u_i - \overline{u_i})(u_i - \overline{u_i})(u_k - \overline{u_k})
\]

\[
= [2\overline{u_i} \overline{u_k} u_i - 2\overline{u_i} u_k u_i - \overline{u_k} u_i u_i + \overline{u_i u_k} u_i + u_i u_i u_k] = 2\overline{u_i} \overline{u_k} u_i - 2\overline{u_i} u_k u_i - \overline{u_k} u_i u_i + u_i u_i u_k
\]

\[
\overline{p' u_j^l} = (p - \overline{p})(u_j - \overline{u_j}) = \overline{p u_j} - \overline{p} u_j - \overline{u_j} p + \overline{p u_j} = \overline{p u_j} - \overline{p} u_j
\]

Table B.1 shows the quantities that must be maintained as running averages in order to reconstruct all the terms in Eq. 6.4. The first column contains the terms in Eq. 6.4 that involve correlations. The second column gives the addition, if any, to the quantities which must be computed as a running average in order to reconstruct all the terms in Eq. 6.4. It is possible for the same running average quantity to appear in more than one term in Eq. 6.4.
### Table B.1: Running average terms in \( k \) equation

<table>
<thead>
<tr>
<th>Term in 6.4</th>
<th>Average term</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>( u'_i u'_j = u_i u_j - u_i u_j )</td>
<td>( u_i u_j )</td>
<td>second-order symmetric tensor</td>
</tr>
<tr>
<td>( u'_i u'_k u'_k = u_i u_k u_k - u_i u_k u_k )</td>
<td>( u_i u_k u_k )</td>
<td>scalar</td>
</tr>
<tr>
<td>( { u'_i u'_k }_k = 2u_i u_k u_i - 2u_i u_k u_i )</td>
<td>( u_i u_k u_i )</td>
<td>scalar</td>
</tr>
<tr>
<td>( -u_i u_i u_k + u_i u_i u_k } )</td>
<td>( u_i u_i u_k )</td>
<td>first-order tensor</td>
</tr>
<tr>
<td>( \rho' u'_j = \rho u_j - \rho' u_j )</td>
<td>( \rho u_j )</td>
<td>first-order tensor</td>
</tr>
</tbody>
</table>

### B.2 Exact \( \varepsilon \) Equation

The exact \( \varepsilon \) equation (where \( \varepsilon = \nu u'_{i,k} u'_{i,k} \) is the dissipation of turbulence) is given in dimensional form (Wilcox, 1993) by:

\[
\frac{\partial \varepsilon}{\partial t} + \rho \mathbf{w}_j \frac{\partial \varepsilon}{\partial x_j} =
\]

\[
-2\mu \left( u'_{i,k} u'_{j,k} + u'_{k,i} u'_{j,k} \right) \frac{\partial u_i}{\partial x_j} - 2\mu u'_i u'_j \frac{\partial^2 u_i}{\partial x_k \partial x_j} - 2\mu u'_{i,k} u'_{i,m} u'_{k,m}
\]

\[
-2\mu' \frac{u'_{i,k} u'_{i,k} u'_{i,k,m} U^2}{L} + \frac{\partial}{\partial x_j} \left[ \mu \frac{\partial \varepsilon}{\partial x_j} - \mu u'_j u'_{i,m} u'_{i,m} - 2\nu \frac{u'_{j,m} U}{L^2} \right], \text{ where } \varepsilon = \nu u'_{i,k} u'_{i,k}
\]

This equation can be non-dimensionalized by a reference velocity \( U \) and a reference length \( L \).

\[
\frac{\rho}{\rho' \rho L^2} \frac{\partial \varepsilon}{\partial t} + \frac{\rho \mathbf{w}_j}{\rho L} \frac{\partial \varepsilon}{\partial x_j} =
\]

\[
-2\mu' \frac{u'_{i,k} u'_{j,k} + u'_{k,i} u'_{j,k}}{L^2} \frac{U^2}{L} \frac{\partial u_i}{\partial x_j} - 2\mu u'_i u'_j \frac{U^2}{L} \frac{\partial^2 u_i}{\partial x_k \partial x_j} - 2\mu u'_{i,k} u'_{i,m} \frac{U^3}{L^3}
\]

\[
-2\mu' \frac{u'_{i,k} u'_{i,k} u'_{i,k,m} U^2}{L} + \frac{\partial}{\partial x_j} \left[ \mu \frac{\partial \varepsilon}{\partial x_j} \frac{U^2}{L^2} - \mu u'_j u'_{i,m} u'_{i,m} - 2\nu \frac{u'_{j,m} U}{L^2} \right]
\]

Multiplying through by \( \frac{L^3}{U^3} \),

\[
\frac{\rho}{\rho' \rho} \frac{\partial \varepsilon}{\partial t} + \rho \mathbf{w}_j \frac{\partial \varepsilon}{\partial x_j} =
\]

\[
-2\mu' \frac{u'_{i,k} u'_{j,k} + u'_{k,i} u'_{j,k}}{L} \frac{1}{U} \frac{\partial u_i}{\partial x_j} - 2\mu u'_i u'_j \frac{1}{U} \frac{\partial^2 u_i}{\partial x_k \partial x_j} - 2\mu u'_{i,k} u'_{i,m} u'_{k,m}
\]

\[
-2\mu' \frac{u'_{i,k} u'_{i,k} u'_{i,k,m}}{U} \frac{1}{L} + \frac{\partial}{\partial x_j} \left[ \mu \frac{1}{\partial x_j} \frac{1}{U} - \mu u'_j u'_{i,m} u'_{i,m} - 2\nu \frac{u'_{j,m} U}{L^2} \right]
\]

Dividing by \( \rho \).
Simplifying since $\frac{1}{Re} = \frac{\mu}{\rho \nu L}$,

$$\frac{\partial \varepsilon}{\partial t} + u_j \frac{\partial \varepsilon}{\partial x_j} = -2\nu \left[ u_{i,k} u'_{j,k} + u'_{k,i} u_{j,k} \right] \frac{\partial u_i}{\partial x_j} - 2\nu u'_{i,k} u'_{j,i} \frac{\partial^2 u_i}{\partial x_k \partial x_j} - 2\nu u'_{i,k} u'_{i,m} u'_{k,m}$$

$$-2\nu u'_{i,k,m} u'_{i,k,m} \frac{1}{Re} + \frac{\partial}{\partial x_j} \left[ \frac{\partial \varepsilon}{\partial x_j} Re - \nu \varepsilon \left( u'_{i,k} u'_{i,k} - 2\nu \varepsilon u'_{i,k} \right) \right]$$

Now since $\varepsilon = \nu u'_{i,k} u'_{i,k}$ redefine $\varepsilon = \overline{u'_{i,k} u'_{i,k}}$ and divide through by $\nu$. The result is a non-dimensional expression for $\varepsilon$ (Eq. 6.5). Note that $\varepsilon$ as used in the present work is defined as $\varepsilon = \overline{u'_{i,k} u'_{i,k}}$.

### B.2.1 Computing as a Running Average

Expanding each term involving fluctuating components in the exact $\varepsilon$ equation (Eq. 6.5) results in the following expressions.

$$u'_{i,k} u'_{j,k} = (u_{i,k} - \overline{u_{i,k}})(u_{j,k} - \overline{u_{j,k}})$$

$$= u_{i,k} u_{j,k} - u_{i,k} u_{j,k} - u_{i,k} u_{j,k} + u_{i,k} u_{j,k} = u_{i,k} u_{j,k} - u_{i,k} u_{j,k}$$

$$u'_{k,i} u'_{j,k} = (u_{k,i} - \overline{u_{k,i}})(u_{j,k} - \overline{u_{j,k}})$$

$$= u_{k,i} u_{j,k} - u_{k,i} u_{j,k} - u_{k,i} u_{j,k} + u_{k,i} u_{j,k} = u_{k,i} u_{j,k} - u_{k,i} u_{j,k}$$

$$u'_{k,i} u'_{i,m} u'_{k,m} = (u_{k,i} - \overline{u_{k,i}}) (u_{i,m} - \overline{u_{i,m}}) (u_{k,m} - \overline{u_{k,m}})$$

$$= \left[ u_{i,m} u_{k,m} u_{k,m} - u_{i,m} u_{i,m} u_{k,m} - u_{i,m} u_{k,m} u_{i,m} + u_{i,m} u_{k,m} u_{i,m} \right]$$

$$= 2u_{i,m} u_{k,m} u_{i,m} - u_{i,m} u_{i,m} u_{k,m} - u_{i,m} u_{i,m} u_{k,m} - u_{i,m} u_{i,m} u_{k,m} + u_{i,m} u_{i,m} u_{k,m}$$

$$u'_{i,k,m} u'_{i,k,m} = (u_{i,k,m} - \overline{u_{i,k,m}})(u_{i,k,m} - \overline{u_{i,k,m}})$$

$$= u_{i,k,m} u_{i,k,m} - u_{i,k,m} u_{i,k,m} - u_{i,k,m} u_{i,k,m} + u_{i,k,m} u_{i,k,m}$$

$$= u_{i,k,m} u_{i,k,m} - u_{i,k,m} u_{i,k,m}$$
Table B.2: Running average terms in \( \varepsilon \) equation

<table>
<thead>
<tr>
<th>Term in Eq. 6.5</th>
<th>Average term</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>( u_{i,k} u_{j,k} = u_{i,k} u_{j,k} - u_{i,k} u_{j,k} )</td>
<td>( u_{i,k} u_{j,k} )</td>
<td>a second-order symmetric tensor</td>
</tr>
<tr>
<td>( u_{k,i} u_{k,j} = u_{k,i} u_{k,j} - u_{k,i} u_{k,j} )</td>
<td>( u_{k,i} u_{k,j} )</td>
<td>a second-order symmetric tensor</td>
</tr>
<tr>
<td>( u_{k,j} u_{k,i} = u_{k,j} u_{k,i} - u_{k,j} u_{k,i} )</td>
<td>( u_{k,j} u_{k,i} )</td>
<td>a third-order tensor</td>
</tr>
<tr>
<td>( { u_{i,m}^t u_{i,m}^t u_{k,m} = 2u_{i,m}^t u_{k,m} u_{i,m} } )</td>
<td>( u_{i,k} u_{i,m} u_{k,m} )</td>
<td>a second-order tensor</td>
</tr>
<tr>
<td>( { u_{i,m}^t u_{i,m}^t u_{k,m} = 2u_{i,m}^t u_{k,m} u_{i,m} } )</td>
<td>( u_{i,k} u_{i,m} u_{k,m} )</td>
<td>a scalar</td>
</tr>
<tr>
<td>( p_m^t u_{j,m} = p_m^t u_{j,m} - p_m^t u_{j,m} )</td>
<td>( p_m^t u_{j,m} )</td>
<td>a first-order tensor</td>
</tr>
</tbody>
</table>

\[
\begin{align*}
  u_{j,m}^t u_{i,m}^t u_{j,m}^t &= (u_j - u_j)(u_i - u_i)(u_i - u_i) \\
  &= \left[ u_j u_i - u_j u_i - u_j u_i + u_j u_i u_i + u_i u_i u_i u_i \right] \\
  &= \left[ u_j u_i u_i - u_j u_i u_i + u_j u_i u_i + u_i u_i u_i \right] \\
  &= 2u_j u_i u_i - 2u_i u_i u_i + 2u_i u_i u_i + u_i u_i u_i \\
  p_m^t u_{j,m}^t &= (p_m - p_m)(u_j - u_j) \\
  &= p_m u_j - u_j p_m - u_j p_m + p_m u_j \\
  &= p_m u_j - p_m u_j
\end{align*}
\]

Table B.2 is analogous to Table B.1. Running average terms appear only once in the second column. For instance \( u_{i,k}^t u_{j,k}^t \) contains the term \( u_{i,k} u_{j,k} \) which also appears in the running average equivalent of \( u_{i,k}^t u_{i,m}^t u_{k,m} \) (in the term \( u_{i,k} u_{i,m} u_{k,m} \)). Therefore \( u_{i,k} u_{j,k} \) appears only once in Table B.2.

### B.3 Computational Work

The computational work involved in computing as running averages all products of fluctuating components appearing in Equations 6.4 and 6.5 can now be estimated. From the \( k \) equation (Eq. 6.4) we have:
Average term | Computational work ( ) → if symmetry is used
--- | ---
$u_i u_j$ | 9 (6) multiplications and 8 (5) additions, 9 (6) storage locations
$u_i u_{i,k}$ | 9 multiplications and 8 additions, 1 storage location
$u_\tau^\tau$ | 3 multiplications and 2 additions, 1 storage location
$u_i u_{\tau} u_{\tau}$ | 12 multiplications and 6 additions, 3 storage locations
$p u_j$ | 3 multiplications, 3 storage locations
**total** | 36 (33) multiplications, 27 (24) additions, 17 (14) storage locations

From the $\varepsilon$ equation we have:

<table>
<thead>
<tr>
<th>Average term</th>
<th>Computational work ( ) → if symmetry is used</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_i u_{i,k}$</td>
<td>27 (18) multiplications and 18 (12) additions, 9 (6) storage locations</td>
</tr>
<tr>
<td>$u_{i,j} u_{k,j}$</td>
<td>27 (18) multiplications and 18 (12) additions, 9 (6) storage locations</td>
</tr>
<tr>
<td>$u_i u_{i,j}$</td>
<td>27 multiplications and 26 additions, 27 storage locations</td>
</tr>
<tr>
<td>$u_{i,k} u_{k,m}$</td>
<td>27 multiplications and 18 additions, 9 storage locations</td>
</tr>
<tr>
<td>$u_{i,k} u_{i,m} u_{k,m}$</td>
<td>54 multiplications and 26 additions, 1 storage location</td>
</tr>
<tr>
<td>$u_{i,k} u_{i,m} u_{k,m}$</td>
<td>27 multiplications and 26 additions, 1 storage location</td>
</tr>
<tr>
<td>$u_{j} u_{i,m} u_{i,m}$</td>
<td>27 multiplications and 24 additions, 3 storage locations</td>
</tr>
<tr>
<td>$p,m u_{j,m}$</td>
<td>9 multiplications and 6 additions, 3 storage locations</td>
</tr>
<tr>
<td><strong>total</strong></td>
<td>225 (207) multiplications and 152 (140) additions, 65 (59) storage locations</td>
</tr>
</tbody>
</table>

The total amount of computational work involved in computing the running averages for all terms in Equations 6.4 and 6.5 is 261 (240) multiplications, 179 (164) additions and 82 (73) storage locations. In addition, the following derivatives must be computed before running averages are made of products involving them.

<table>
<thead>
<tr>
<th>Average term</th>
<th>Computational work</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p,m$</td>
<td>15 multiplications and 12 additions, 3 storage locations</td>
</tr>
<tr>
<td>$u_{j,m}$</td>
<td>45 multiplications and 36 additions, 9 storage locations</td>
</tr>
<tr>
<td>$u_{i,km}$</td>
<td>135 multiplications and 104 additions, 27 storage locations</td>
</tr>
</tbody>
</table>

The number of multiplications and additions required to compute the derivative terms depends on the finite difference scheme used. A five point finite difference stencil is assumed in order to obtain the numbers above. If the finite difference grid is not staggered then all of the derivative terms above (except for the mixed derivatives) would also be needed to solve the Navier-Stokes...
equations and potentially could be reused. However, in the present work a staggered grid is used. One evaluation of the Navier-Stokes equations using a five point diffusive scheme and four point schemes for all other terms requires 399 floating point operations for an implicit time integration scheme. The grand total number of floating point operations involved in computing the running averages is 787 (751). Total amount of storage is 121 (112). The small increase in computational savings possible from taking advantage of the symmetries of certain terms indicates that the ease of implementation from ignoring the symmetries will probably be more important.

One way to collect mean velocities (or other statistics) is to add up every time step in a run. However, this is computationally expensive. A more efficient method may be to add up a subset of the time series. This subset could be chosen in an optimal way so that for the number of subsets chosen it gives the optimal approximation to the mean. The separation between times in the subset could be based on the time it takes for two fields to be decorrelated. This has the potential of substantial computational savings, particularly as regards computing the many correlations involving derivatives in Equations 6.4 and 6.5. Questions that must be answered are how to choose the decorrelation time step and how the computed statistics change as the size of the subset increases. This approach has not been explored further in the present work.

In order for the residuals of Equations 6.4 and 6.5 to be zero within machine accuracy it would be necessary to derive Equations 6.4 and 6.5 in a discrete sense using the same finite difference scheme used to solve Equations 6.4 and 6.5. This would require that the changed stencils near boundaries used in Equations 6.4 and 6.5 be used when deriving discrete versions of Equations 6.4 and 6.5. Due to the high order stencils used in the present work to discretize Equations 6.4 and 6.5 this would require much larger stencils when derivatives are taken of derivatives in deriving Equations 6.4 and 6.5. This would require a large amount of data to be communicated between
processes. In addition, there is the daunting complexity of deriving discrete versions of Equations 6.4 and 6.5. Another difficulty is due to the staggered grid which would require interpolation. As a result, this approach has not been taken. The terms appearing in Equations 6.4 and 6.5 are discretized on the main grid points using stencils of the same order of accuracy as used in discretizing Equations 6.4 and 6.5.
Appendix C Standard Kinetic Energy and Dissipation Model

C.1 Modeled $k$ Equation

The modeled $k$ equation is given in dimensional form (Wilcox, 1993) by:

$$\rho \frac{\partial k}{\partial t} + \rho \mu_{ij} \frac{\partial k}{\partial x_j} = \left[ \rho C_\mu \frac{k^2}{\varepsilon} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \frac{2}{3} \rho k \delta_{ij} \right] \frac{\partial u_i}{\partial x_j}$$

$$- \rho \varepsilon + \frac{\partial}{\partial x_j} \left[ \left( \mu + \rho C_\mu \frac{k^2}{\sigma_k} \right) \frac{\partial k}{\partial x_j} \right]$$

where $\varepsilon = u_{i,k}^' u_{i,k}^'$. 

As $\frac{\partial u_i}{\partial x_j} = 0$, $\frac{2}{3} \rho k \delta_{ij} \frac{\partial u_i}{\partial x_j} = 0$ the above can be rewritten as:

$$\rho \frac{\partial k}{\partial t} + \rho \mu_{ij} \frac{\partial k}{\partial x_j} = \rho C_\mu \frac{k^2}{\varepsilon} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \frac{\partial u_i}{\partial x_j} - \rho \varepsilon + \frac{\partial}{\partial x_j} \left[ \left( \mu + \rho C_\mu \frac{k^2}{\sigma_k} \right) \frac{\partial k}{\partial x_j} \right]$$

Non-dimensionalizing by a reference velocity $U$ and a reference length $L$:

$$\rho \frac{\partial U^2}{\partial t} + \rho \mu_{ij} \frac{\partial k}{\partial x_j} =$$

$$\rho C_\mu \frac{k^2 U^4}{\varepsilon U^2 L^2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \frac{\partial u_i}{\partial x_j} - \rho U^2 L^{-2} + \frac{\partial}{\partial x_j} \left[ \left( \mu + \rho C_\mu \frac{k^2 U^4}{\sigma_k} \right) \frac{\partial k}{\partial x_j} \right]$$

Multiplying through by $U^{-3} L$,

$$\rho \frac{\partial k}{\partial t} + \rho \mu_{ij} \frac{\partial k}{\partial x_j} =$$

$$\rho C_\mu \frac{k^2}{\varepsilon} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \frac{\partial u_i U}{\partial x_j} - \rho U^{-1} L^{-1} + \frac{\partial}{\partial x_j} \left[ \left( \mu + \rho C_\mu \frac{k^2 U^4}{\sigma_k} \right) \frac{\partial k U}{\partial x_j} \right]$$

Dividing by $\rho$.

$$\frac{\partial k}{\partial t} + \mu_{ij} \frac{\partial k}{\partial x_j} = C_\mu \frac{k^2}{\varepsilon} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \frac{\partial u_i U}{\partial x_j} - \varepsilon U^{-1} L^{-1} + \frac{\partial}{\partial x_j} \left[ \left( \mu + \rho C_\mu \frac{k^2 U^4}{\sigma_k} \right) \frac{\partial k U}{\partial x_j} \right]$$

Now, recalling that $\varepsilon = \nu u_{i,k}^' u_{i,k}^'$, redefine $\varepsilon = u_{i,k}^' u_{i,k}^'$.

$$\frac{\partial k}{\partial t} + \mu_{ij} \frac{\partial k}{\partial x_j} = C_\mu \frac{k^2}{\varepsilon} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \frac{\partial u_i U}{\partial x_j} - \nu U^{-1} L^{-1} + \frac{\partial}{\partial x_j} \left[ \left( \mu + \rho C_\mu \frac{k^2 U^4}{\sigma_k} \right) \frac{\partial k U}{\partial x_j} \right]$$

After simplifying, the result is a non-dimensional expression for the modeled $k$ equation (Eq. 6.6).
C.2 Modeled \( \varepsilon \) Equation

The modeled \( \varepsilon \) equation is given in dimensional form (Wilcox, 1993) by:

\[
\frac{\partial \varepsilon}{\partial t} + \rho \frac{u_j}{\partial x_j} = C_{e1} \left[ \frac{\rho C_{\mu} k^2}{\varepsilon} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \frac{2}{3} \rho k \delta_{ij} \right] \frac{\partial u_i}{\partial x_j} \\
- \rho C_{e2} \frac{\varepsilon^2}{k} + \frac{\partial}{\partial x_j} \left[ \left( \mu + \frac{\rho C_{\mu} k^2}{\sigma_\varepsilon} \right) \frac{\partial \varepsilon}{\partial x_j} \right]
\]

where \( \varepsilon = \nu u'_{i,k} u'_{i,k} \)

As \( \frac{\partial \varepsilon}{\partial x_i} = 0 \), \( \frac{2}{3} \rho k \delta_{ij} \frac{\partial \varepsilon}{\partial x_j} = 0 \) the above can be rewritten as:

\[
\frac{\partial \varepsilon}{\partial t} + \rho \frac{u_j}{\partial x_j} = C_{e1} C_{\mu} \rho k \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \frac{\partial u_i}{\partial x_j} - \rho C_{e2} \frac{\varepsilon^2}{k} + \frac{\partial}{\partial x_j} \left[ \left( \mu + \frac{\rho C_{\mu} k^2}{\sigma_\varepsilon} \varepsilon \right) \frac{\partial \varepsilon}{\partial x_j} \right]
\]

Non-dimensionalizing by a reference velocity \( U \) and a reference length \( L \):

\[
\frac{\rho \varepsilon U^2 L^{-2}}{\partial t L U^{-1}} + \frac{\rho u_j}{\partial x_j} = C_{e1} C_{\mu} \rho U^2 \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \frac{\partial u_i}{\partial x_j} - \rho C_{e2} \frac{\varepsilon^2 U^4 L^{-4}}{k} + \frac{\partial}{\partial x_j} \left[ \left( \mu + \frac{\rho C_{\mu} k^2 U^2 L^2}{\sigma_\varepsilon} \varepsilon U^2 L^{-2} \right) \frac{\partial \varepsilon}{\partial x_j} \right]
\]

Multiplying through by \( U^{-3} L^3 \)

\[
\frac{\rho \varepsilon}{\partial t} + \frac{\rho u_j}{\partial x_j} = C_{e1} C_{\mu} k \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \frac{\partial u_i}{\partial x_j} - \rho C_{e2} \frac{\varepsilon^2 U^{-1} L^{-1}}{k} + \frac{\partial}{\partial x_j} \left[ \left( \mu + \frac{\rho C_{\mu} k^2 U L}{\sigma_\varepsilon} \varepsilon \right) \frac{\partial \varepsilon}{\partial x_j} \right]
\]

Dividing by \( \rho \).

\[
\frac{\partial \varepsilon}{\partial t} + \frac{u_j}{\partial x_j} = C_{e1} C_{\mu} k \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \frac{\partial u_i}{\partial x_j} - C_{e2} \frac{\varepsilon^2 U^{-1} L^{-1}}{k} + \frac{\partial}{\partial x_j} \left[ \left( \mu + \frac{\rho C_{\mu} k^2 U L}{\sigma_\varepsilon} \varepsilon \right) \frac{\partial \varepsilon}{\partial x_j} \right]
\]

Now since \( \varepsilon = \nu u'_{i,k} u'_{i,k} \), redefine \( \varepsilon = \nu u'_{i,k} u'_{i,k} \) and divide by \( \nu \).

\[
\frac{\partial \varepsilon}{\partial t} + \frac{u_j}{\partial x_j} = C_{e1} C_{\mu} k \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \frac{\partial u_i}{\partial x_j} - C_{e2} \frac{\varepsilon^2 U^{-1} L^{-1} \nu}{k} + \frac{\partial}{\partial x_j} \left[ \left( \mu + \frac{\rho C_{\mu} k^2 U L}{\sigma_\varepsilon} \varepsilon \right) \frac{\partial \varepsilon}{\partial x_j} \right]
\]

After simplifying, the result is a non-dimensional expression for the modeled \( \varepsilon \) equation (Eq. 6.7).
As $k = 0$ on the blade surface, but may be non-zero in the interior:

$$\frac{\partial k}{\partial x_1} = \frac{\partial k}{\partial x_3} = 0 \text{ but } \frac{\partial k}{\partial x_2} \neq 0$$

In the jet exit plane $u_2 \neq 0$. Taking into account the above, in the jet exit plane the exact $k$ equation (Eq. 6.4) reduces to:

$$\frac{u_2}{2} \frac{\partial k}{\partial x_2} = -\frac{1}{Re} \varepsilon + \frac{\partial}{\partial x_2} \left[ \frac{1}{Re} \frac{\partial k}{\partial x_2} - \frac{1}{2} u_i^' u_i^' u_2^' + p^' u_2^' \right] \text{ where } \varepsilon = u_i^' u_i^'$$

The term $\frac{u_2}{2} \frac{\partial k}{\partial x_2}$ physically represents convection of $k$ into the crossflow from the jet delivery tube. As $k = 0$ in the jet exit plane, this term should be zero. However, if a one-sided finite difference expression is used to represent $\frac{\partial k}{\partial x_2}$ (which is the only possibility on the blade surface) $\frac{\partial k}{\partial x_2}$ may not be zero and therefore $\frac{u_2}{2} \frac{\partial k}{\partial x_2}$ may not be zero. The basic problem is that $\frac{u_2}{2} \frac{\partial k}{\partial x_2}$ is being represented by a downwind finite difference scheme. If an upwind finite difference scheme was used instead, then there would be no problem as $k = 0$ in the jet delivery tube. This problem is avoided by setting $\frac{u_2}{2} \frac{\partial k}{\partial x_2} = 0$ in the jet exit plane. As the same issue arises at the crossflow inlet boundary plane in which $u_1^i$ and $\frac{\partial k}{\partial x_1}^i$ may not be zero, $\frac{u_1^i}{2} \frac{\partial k}{\partial x_1}^i$ is set to zero in the jet exit plane.

If the term $\frac{u_j}{2} \frac{\partial \varepsilon}{\partial x_j}$ in the exact $\varepsilon$ equation (Eq. 6.5) is written in a coordinate system one axis of which is aligned with the inclination angle, then as $\varepsilon$ assumed to be constant, this term is zero, although $\varepsilon$ itself may not be zero. Therefore $\frac{u_j}{2} \frac{\partial \varepsilon}{\partial x_j}$ is set to zero in the jet exit and crossflow inlet boundary planes.
Appendix E  Proof that $u'_i u'_i = v'_i v'_i = 0$ at $z = 0$

The $z$ momentum component of the Reynolds Averaged Navier-Stokes equations is given by:

$$
\frac{\partial w}{\partial x} + \frac{\partial w}{\partial y} + \frac{\partial w}{\partial z} + \frac{\partial (w'u')}{\partial x} + \frac{\partial (w'v')}{\partial y} + \frac{\partial (w'w')}{\partial z} = -\frac{\partial p}{\partial z} + \frac{1}{Re} \left( \frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} + \frac{\partial^2 w}{\partial z^2} \right)
$$

Assuming no gradients in the $z$ direction with the additional assumption that $w = 0$, results in:

$$
\frac{\partial (w'u')}{\partial x} + \frac{\partial (w'v')}{\partial y} = 0
$$

At the wall $u'_i u'_i = 0$ and therefore $\frac{\partial (w'u')}{\partial x} = 0$. Note that this is true also in the jet exit region, as the jet exit profile is not a function of time. As $\frac{\partial (w'u')}{\partial x} = 0$ at the wall, $\frac{\partial (w'v')}{\partial y} = 0$ at the wall and as $u'_i u'_i = 0$ at the wall, $w'_i v' = 0$ for all $y$. As $u'_i u'_i = 0$ for all $y$, $\frac{\partial (w'u')}{\partial x} = 0$ for all $y$. At the cross flow inflow boundary $u'_i u'_i = 0$, as the cross flow profile is not a function of time. As $\frac{\partial (w'u')}{\partial x} = 0$ for all $y$, $w'_i u'_i = 0$ for all $y$. Therefore $u'_i u'_i = v'_i v'_i = 0$ in $xy$ planes for which there are no gradients in the $z$ direction and $\overline{w} = 0$. 

285
Appendix F  Wake Vortex Formation

Figure F.1: Wake vortex visualization by particle traces, $St = 0.2$
Figure F.2: Wake vortex visualization by particle traces, $St = 0.4$
Figure F.3: Wake vortex visualization by particle traces, $St = .4$
Frank Herbert Muldoon was born in Ann Arbor, Michigan in 1971. He was home schooled with the exception of his last year of high school, when he graduated from Bedford High School, Bedford, Massachusetts, in 1989. He came to Louisiana in 1989 when his father was assigned to England Air Force Base in Alexandria. He was a member of the Louisiana Army National Guard from August 1989 to August 1996. He attended Louisiana State University at Alexandria from fall 1990 to spring 1992. In August 1992 he moved to Louisiana State University in Baton Rouge, receiving his bachelor’s degree in mechanical engineering in 1996. He then started graduate school, working for a year as a Computational Fluid Dynamics analyst in the Math Modeling group at Dow Chemical in Plaquemine, Louisiana. In the summer of 2000 he worked as a Computational Fluid Dynamics analyst at the Air Force Jet Propulsion Laboratory at Wright-Patterson AFB, Ohio.