A Sliding Interface Method for Unsteady Unstructured Parallel Flow Simulations

Eric Lindsay Blades

Follow this and additional works at: https://scholarsjunction.msstate.edu/td

Recommended Citation
https://scholarsjunction.msstate.edu/td/256

This Dissertation is brought to you for free and open access by the Theses and Dissertations at Scholars Junction. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of Scholars Junction. For more information, please contact scholcomm@msstate.libanswers.com.
A SLIDING INTERFACE METHOD FOR UNSTEADY UNSTRUCTURED PARALLEL FLOW SIMULATIONS

By

Eric Lindsay Blades

A Dissertation
Submitted to the Faculty of Mississippi State University in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in Mechanical Engineering in the Department of Mechanical Engineering

Mississippi State, Mississippi

December 2004
A SLIDING INTERFACE METHOD FOR UNSTEADY UNSTRUCTURED PARALLEL FLOW SIMULATIONS

By

Eric Lindsay Blades

Approved:

David L. Marcum
Professor of Mechanical Engineering
(Director of Dissertation and Major Professor)

James C. Newman, III
Associate Professor of Aerospace Engineering
(Committee Member)

D. Keith Walters
Assistant Professor of Mechanical Engineering
(Committee Member)

W. Roger Briley
Professor of Computational Engineering
The University of Tennessee at Chattanooga
(Committee Member)

Rogelio Luck
Associate Professor of Mechanical Engineering and Graduate Coordinator

David L. Whitfield
Associate Dean of the College of Engineering and Computer Science
The University of Tennessee at Chattanooga
(Committee Member)

W. Glenn Steele
Interim Dean of the Bagley College of Engineering
The primary objective of this study is to develop a sliding interface method for simulations involving relative rotational grid motion suitable for unstructured grid topologies. The present method alleviates computationally expensive grid deformation, remeshing, and hole cutting procedures. Rotational motion is accomplished by rigidly rotating a subdomain representing the moving component. At the subdomain interface boundary, the faces along the interfaces are extruded into the adjacent subdomain to create new volume elements and provide a one-cell overlap. These new volume elements close the control volumes for the nodes on the interface surface and allow a flux to be computed across the subdomain interface. An interface flux is computed independently for each subdomain. The values of the solution variables and other quantities for the nodes created by the extrusion process are found by interpolation. The extrusion is done so that the interpolation will maintain information as localized as possible. A parallel
implementation of the neighbor search is used to find the extruded points in the adjacent subdomain.

The method has been implemented in a parallel, node-centered finite volume, high-resolution viscous flow solver. The method does not impose any restrictions on the subdomain interface aside from the axisymmetric limitation required for rotational motion. In addition, the grid on the subdomain interface is arbitrary. The boundary surfaces between the two subdomains can have independent grids from one another. They do not have to connect in a one-to-one manner and there are no symmetry or pattern restrictions placed on the surface grid.

A variety of numerical simulations were performed on several small-scale model problems to examine conservation of the interface flux. Overall flux conservation errors were found to be comparable to that for fully connected and fully conservative simulations. In addition, excellent agreement was obtained with both theoretical and experimental results.

Three large-scale applications were also used to validate the method and highlight some of the advantages of the sliding interface method compared to the current state-of-the-art for unstructured grid applications. This sliding interface method requires no geometric modifications and has significantly shorter run times. Furthermore, there were no apparent adverse effects on the numerical solutions by not strictly enforcing flux conservation at the subdomain boundary.
DEDICATION

I would like to dedicate this work to my wife, Susan, for enduring the countless hours spent in the pursuit of this endeavor and to my children, Hannah, Samuel, and Luke, for always reminding me what are truly the most important things in life.
ACKNOWLEDGEMENTS

The author would like to express his sincere appreciation to Dr. David Marcum for his guidance and understanding throughout this endeavor and for his advice on technical as well as non-technical issues. As Director of the SimCenter, he is also due my thanks and gratitude for providing financial support during the course of this work. I would also like to thank all the members of my committee, namely, Dr. James Newman, III, Dr. Keith Walters, Dr. Roger Briley, and Dr. David Whitfield, for offering their valuable comments, advice, and suggestions.

I would like to thank the members of the SimCenter for their support and encouragement, and in particular, Dr. Mark Janus for being an extremely helpful source of information. I would like to extend special thanks to two former SimCenter members, Dr. Kidambi Sreenivas and Dr. Daniel Hyams, who both provided many valuable suggestions and took the time to answer my never ending supply of questions. Sree’s idea started this work and he served as a sounding board for many of my ideas and Daniel continues to offer advice and suggestions. Thanks are also due to former SimCenter member Brent Mitchell for cleaning up the missile geometries. Special thanks are also due to Dr. Monty Hughson for having taken the time end effort to do a thorough review of this manuscript and provide many helpful suggestions.

I would like to acknowledge Dr. Chunhua Sheng for providing the grid and solutions for the high-speed centrifugal compressor case. In addition, I would like to
thank Dr. Tor Nygaard of ELORET and Dr. Robert Meakin of the U.S. Army
Aeroflightdynamics Directorate at NASA Ames Research Center for providing the
OVERFLOW-D simulation results for comparison and to Dr. Scott Murman of ELORET
and Mr. Michael Aftosmis of NASA Ames Research Center for providing the canard
command schedule used for the comparison to the experimental data.

Eric Lindsay Blades
Mississippi State University
October 2004
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEDICATION</td>
<td>ii</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENTS</td>
<td>iii</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>viii</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>x</td>
</tr>
<tr>
<td>CHAPTER</td>
<td></td>
</tr>
<tr>
<td>I. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Structured Grid Topology Grid Motion Strategies</td>
<td>2</td>
</tr>
<tr>
<td>1.2 Unstructured Grid Topology Grid Motion Strategies</td>
<td>5</td>
</tr>
<tr>
<td>1.3 UVI Methodology</td>
<td>8</td>
</tr>
<tr>
<td>1.4 Objectives and Approach</td>
<td>13</td>
</tr>
<tr>
<td>1.5 Outline</td>
<td>15</td>
</tr>
<tr>
<td>II. NUMERICAL DISCRETIZATION AND SOLUTION ALGORITHM</td>
<td>16</td>
</tr>
<tr>
<td>2.1 Governing Equations</td>
<td>16</td>
</tr>
<tr>
<td>2.2 Spatial Discretization</td>
<td>23</td>
</tr>
<tr>
<td>2.2.1 Inviscid Contributions</td>
<td>24</td>
</tr>
<tr>
<td>2.2.2 Higher Order Methods</td>
<td>28</td>
</tr>
<tr>
<td>2.2.3 Gradient Estimation</td>
<td>29</td>
</tr>
<tr>
<td>2.2.4 Flux Limiting</td>
<td>31</td>
</tr>
<tr>
<td>2.2.5 Viscous Contributions</td>
<td>33</td>
</tr>
<tr>
<td>2.3 Temporal Discretization</td>
<td>34</td>
</tr>
<tr>
<td>2.3.1 Time Advancement Scheme</td>
<td>36</td>
</tr>
<tr>
<td>2.3.2 Linear System Solution Technique</td>
<td>39</td>
</tr>
<tr>
<td>2.3.3 Jacobian Calculations</td>
<td>40</td>
</tr>
<tr>
<td>2.4 Turbulence Modeling</td>
<td>42</td>
</tr>
<tr>
<td>2.5 Boundary Conditions</td>
<td>46</td>
</tr>
<tr>
<td>2.5.1 Inviscid Surface</td>
<td>46</td>
</tr>
<tr>
<td>2.5.2 Viscous Surface</td>
<td>47</td>
</tr>
<tr>
<td>2.5.3 Rotating Viscous Surface</td>
<td>47</td>
</tr>
<tr>
<td>2.5.4 Farfield Conditions</td>
<td>48</td>
</tr>
</tbody>
</table>
## Table of Contents

### CHAPTER 2.5.5 Symmetry Plane Boundary Conditions ............................................. 51
### 2.6 Parallel Implementation ................................................................................ 51

### III. SLIDING INTERFACE METHODOLOGY ..................................................... 53
### 3.1 Sliding Interface ............................................................................................ 53
### 3.2 Construction of the Sliding Interface ............................................................ 54
### 3.3 Search Algorithm and Interpolation .............................................................. 64
### 3.4 Flux Computation ......................................................................................... 69
### 3.5 Subdomain Coupling .................................................................................... 71

### IV. MODEL VALIDATION PROBLEMS .............................................................. 72
### 4.1 Steady, Inviscid, Incompressible Shear Flow ............................................... 73
### 4.2 Steady, Inviscid, Supersonic Flow in a Diverging Nozzle ........................... 76
### 4.3 Steady, Inviscid, Supersonic Ramp .............................................................. 79
### 4.4 Unsteady, Inviscid Shock Tube .................................................................... 83
### 4.5 Incompressible, Viscous Cylinder in Crossflow ........................................... 89
### 4.6 Unsteady, Incompressible, Viscous Marine Propeller ................................. 95

### V. LARGE-SCALE APPLICATION VALIDATION PROBLEMS ...................... 104
### 5.1 Case 1: Spinning Missile with Dithering Canards ........................................ 104
#### 5.1.1 Grid Construction .............................................................................. 108
#### 5.1.2 Simulation Conditions and Parameters ............................................. 116
#### 5.1.3 Simulation Results ............................................................................ 117
#### 5.1.4 Computation Expense ....................................................................... 134
### 5.2 Case 2: Missile with Free-Spinning Tail ...................................................... 137
#### 5.2.1 Grid Construction .............................................................................. 140
#### 5.2.2 Simulation Conditions and Parameters ............................................. 150
#### 5.2.3 Simulation Results ............................................................................ 151
##### 5.2.3.1 Static Results ...................................................................... 151
##### 5.2.3.2 Spinning Tail Results ......................................................... 158
#### 5.2.4 Computational Expense ..................................................................... 178
### 5.3 Case 3: High-Speed Centrifugal Compressor ............................................... 179
#### 5.3.1 Grid Construction .............................................................................. 183
#### 5.3.2 Simulation Conditions and Parameters ............................................. 185
#### 5.3.3 Simulation Results ............................................................................ 186

### VI. SUMMARY AND CONCLUSIONS ................................................................. 194

REFERENCES ......................................................................................................... 198
<table>
<thead>
<tr>
<th>APPENDIX</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. EIGENSYSTEM FOR THE CONVECTIVE FLUX JACOBIAN MATRIX</td>
<td>207</td>
</tr>
<tr>
<td>B. VOLUME WEIGHTS FOR TETRAHEDRAL AND NON-SIMPLICAL ELEMENTS</td>
<td>211</td>
</tr>
<tr>
<td>C. FORCE AND MOMENT COEFFICIENT PLOTS FOR THE SPINNING MISSILE WITH DITHERING CANARDS</td>
<td>217</td>
</tr>
<tr>
<td>D. FORCE AND MOMENT COEFFICIENT PLOTS FOR THE MISSILE WITH THE FREE-SPINNING TAIL</td>
<td>224</td>
</tr>
</tbody>
</table>
LIST OF TABLES

<table>
<thead>
<tr>
<th>TABLE</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Comparison of the mass flow across the inlet and outlet surfaces for the diverging supersonic duct</td>
<td>77</td>
</tr>
<tr>
<td>4.2</td>
<td>Flow properties across the oblique shock wave for the 20° ramp at Mach 2</td>
<td>79</td>
</tr>
<tr>
<td>4.3</td>
<td>Net mass flow across the interface for the 20° ramp at Mach 2</td>
<td>80</td>
</tr>
<tr>
<td>4.4</td>
<td>Vortex core location for Re = 26 as measured aft of the cylinder from the centerline</td>
<td>90</td>
</tr>
<tr>
<td>4.5</td>
<td>Axial force coefficient for the three-bladed marine propeller</td>
<td>99</td>
</tr>
<tr>
<td>4.6</td>
<td>Net mass flow across the interface for the three-bladed propeller</td>
<td>100</td>
</tr>
<tr>
<td>5.1</td>
<td>Volume grid sizes for the spinning missile</td>
<td>115</td>
</tr>
<tr>
<td>5.2</td>
<td>Roll-averaged force and moment coefficients for the 0% command level</td>
<td>123</td>
</tr>
<tr>
<td>5.3</td>
<td>Roll-averaged force and moment coefficients for the Exp-fit command level</td>
<td>134</td>
</tr>
<tr>
<td>5.4</td>
<td>Computational expense for the spinning missile simulations</td>
<td>135</td>
</tr>
<tr>
<td>5.5</td>
<td>Percentage of the total run time for the individual sliding interface tasks for the spinning missile simulations</td>
<td>135</td>
</tr>
<tr>
<td>5.6</td>
<td>Parallel efficiency measures for the spinning missile simulations</td>
<td>136</td>
</tr>
<tr>
<td>5.7</td>
<td>Volume grid sizes for the free-spinning tail missile</td>
<td>147</td>
</tr>
<tr>
<td>5.8</td>
<td>Roll-averaged force and moment coefficients for the static case (ω = 0 rpm)</td>
<td>155</td>
</tr>
<tr>
<td>5.9</td>
<td>Roll-averaged force and moment coefficients for ω = 2500 rpm</td>
<td>158</td>
</tr>
<tr>
<td>TABLE</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>5.10  Roll-averaged force and moment coefficients for $\omega = 5000$ rpm</td>
<td>159</td>
<td></td>
</tr>
<tr>
<td>5.11  Roll-averaged force and moment coefficients for the estimated zero spin rate condition ($\omega \approx 1980$ rpm)</td>
<td>168</td>
<td></td>
</tr>
<tr>
<td>5.12  Computational expense for the free-spinning tail missile simulations</td>
<td>178</td>
<td></td>
</tr>
<tr>
<td>5.13  Percentage of the total run time for the individual sliding interface tasks for the free-spinning tail missile simulations</td>
<td>178</td>
<td></td>
</tr>
<tr>
<td>5.14  Parallel efficiency measures for the free-spinning tail missile simulations</td>
<td>179</td>
<td></td>
</tr>
<tr>
<td>B.1   Tetrahedron volume weight definition</td>
<td>213</td>
<td></td>
</tr>
<tr>
<td>B.2   Five-node pentahedron element (Pyramid) volume weight definition</td>
<td>214</td>
<td></td>
</tr>
<tr>
<td>B.3   Six-node pentahedron element (Prism) volume weight definition</td>
<td>215</td>
<td></td>
</tr>
<tr>
<td>B.4   Hexahedron element volume weight definition</td>
<td>216</td>
<td></td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 The interface surface (UVI surface) surrounding the propeller delineates the subdomains</td>
<td>10</td>
</tr>
<tr>
<td>1.2 Geometric modification required to the shaft by the UVI Method</td>
<td>12</td>
</tr>
<tr>
<td>1.3 Volume grid inside the shaft due to the gap created by the UVI Method</td>
<td>12</td>
</tr>
<tr>
<td>2.1 Control volume definition for vertex 0</td>
<td>24</td>
</tr>
<tr>
<td>2.2 Stencil for vertex 0</td>
<td>29</td>
</tr>
<tr>
<td>2.3 $U^2$NCLE flow solver iteration hierarchy</td>
<td>52</td>
</tr>
<tr>
<td>3.1 Two-dimensional interface where the interface edges are extruded into quadrilateral elements to form the overlapping elements</td>
<td>55</td>
</tr>
<tr>
<td>3.2 Location of the sliding interface for the marine propeller example</td>
<td>56</td>
</tr>
<tr>
<td>3.3 Extruded sliding interface elements on the (a) primary interface (inner domain) and (b) the overlap into the outer domain</td>
<td>57</td>
</tr>
<tr>
<td>3.4 Extruded sliding interface elements on the (a) secondary interface (outer domain) and (b) the overlap into the inner domain</td>
<td>58</td>
</tr>
<tr>
<td>3.5 Extrusion distance based on the local subdomain can lead to the sliding elements extending too far from the interface</td>
<td>60</td>
</tr>
<tr>
<td>3.6 Boundary-layer extrusion direction based on local node normal does not align with adjacent subdomain boundary-layer grid</td>
<td>63</td>
</tr>
<tr>
<td>3.7 Boundary-layer extrusion direction that aligns with adjacent subdomain boundary-layer grid and the resulting sliding elements</td>
<td>63</td>
</tr>
<tr>
<td>3.8 Area coordinates for a triangular element and signs</td>
<td>67</td>
</tr>
<tr>
<td>FIGURE</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>4.1</td>
<td>Shear flow surface grid</td>
</tr>
<tr>
<td>4.2</td>
<td>Shear flow velocity contours near the subdomain boundary for the (a) UVI solution and the (b) sliding interface solution</td>
</tr>
<tr>
<td>4.3</td>
<td>Inviscid, supersonic, diverging nozzle configuration</td>
</tr>
<tr>
<td>4.4</td>
<td>Mass conservation for the diffuser rotating sliding interface case</td>
</tr>
<tr>
<td>4.5</td>
<td>Mass conservation over one period for the rotating sliding interface case</td>
</tr>
<tr>
<td>4.6</td>
<td>Surface grids near the interface of the of the 20° ramp for the (a) aligned interface and the (b) non-matching interface</td>
</tr>
<tr>
<td>4.7</td>
<td>Density contours across the oblique shock wave for the (a) baseline, (b) aligned interface, and (c) non-matching interface solutions</td>
</tr>
<tr>
<td>4.8</td>
<td>Shock tube grids for the (a) planar interface and the (b) curved interface</td>
</tr>
<tr>
<td>4.9</td>
<td>Comparison of pressure, density, and velocity for the theoretical and computed shock tube solutions</td>
</tr>
<tr>
<td>4.10</td>
<td>Density Contours for the (a) planar baseline, (b) planar interface, (c) curved baseline, and (d) curved interface solutions</td>
</tr>
<tr>
<td>4.11</td>
<td>Velocity Contours for the (a) planar baseline, (b) planar interface, (c) curved baseline, and (d) curved interface solutions</td>
</tr>
<tr>
<td>4.12</td>
<td>Grid near the cylinder for the cylinder in crossflow case</td>
</tr>
<tr>
<td>4.13</td>
<td>Velocity vectors for Re=26 for the (a) baseline solution and the (b) sliding interface solution</td>
</tr>
<tr>
<td>4.14</td>
<td>Pressure contours for Re=26 for the (a) baseline solution and the (b) sliding interface solution</td>
</tr>
<tr>
<td>4.15</td>
<td>Comparison time history of the lift coefficient for the cylinder at Re=100</td>
</tr>
<tr>
<td>4.16</td>
<td>Comparison of the instantaneous pressure contours for Re=100 for the (a) baseline solution and the (b) sliding interface solution</td>
</tr>
</tbody>
</table>
FIGURE

4.17 Interface locations for the three-bladed marine propeller simulations for the (a) baseline and Sliding1 grids and (b) UVI and Sliding2 grids .. 96

4.18 (a) Cutting plane taken through the Sliding2 marine propeller grid and (b) detailed view illustrating the sliding interface surface intersecting the shaft .................................................................................................... 97

4.19 Modification to the shaft and the resulting UVI volume grid for the marine propeller ................................................................. 98

4.20 Convergence history comparison for the three-bladed marine propeller ...... 99

4.21 Centerline axial velocity contours for the (a) baseline and (b) the UVI solutions ................................................................. 101

4.22 Centerline axial velocity contours for the (a) Sliding1 and (b) Sliding2 solutions ............................................................................... 102

4.23 Percent difference in pressure on the (a) primary sliding interface and (b) secondary sliding interface surfaces for the Sliding1 grid ................. 103

5.1 Spinning missile configuration with dithering canards ......................... 106

5.2 Canard range of motion ........................................................................ 107

5.3 Canard command schedule ................................................................... 107

5.4 Fuselage surface grid for the spinning missile ........................................ 110

5.5 Canard surface grid for the spinning missile .......................................... 110

5.6 Empennage surface grid for the spinning missile .................................. 111

5.7 Typical volume grid resolution for the spinning missile grids ................. 112

5.8 Geometric modification to the canard stock for the UVI method .......... 113

5.9 UVI surface for the port side canard ..................................................... 114

5.10 Modification to the canard stock and the resulting UVI volume grid ....... 114
<table>
<thead>
<tr>
<th>FIGURE</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.11</td>
<td>Sliding interface surface and the resulting volume grid near the canard stock</td>
</tr>
<tr>
<td>5.12</td>
<td>Axial force coefficient for the 0% canard command level</td>
</tr>
<tr>
<td>5.13</td>
<td>Side force coefficient for the 0% canard command level</td>
</tr>
<tr>
<td>5.14</td>
<td>Normal force coefficient for the 0% canard command level</td>
</tr>
<tr>
<td>5.15</td>
<td>Rolling moment coefficient for the 0% canard command level</td>
</tr>
<tr>
<td>5.16</td>
<td>Pitching moment coefficient for the 0% canard command level</td>
</tr>
<tr>
<td>5.17</td>
<td>Yawing moment coefficient for the 0% canard command level</td>
</tr>
<tr>
<td>5.18</td>
<td>Detailed view of the axial force coefficient for the 0% canard command level</td>
</tr>
<tr>
<td>5.19</td>
<td>Convergence history comparison for the 0% canard command level</td>
</tr>
<tr>
<td>5.20</td>
<td>Canard axial force coefficient for the 0% canard command level</td>
</tr>
<tr>
<td>5.21</td>
<td>Canard side force coefficient for the 0% canard command level</td>
</tr>
<tr>
<td>5.22</td>
<td>Canard normal force coefficient for the 0% canard command level</td>
</tr>
<tr>
<td>5.23</td>
<td>Canard rolling moment coefficient for the 0% canard command level</td>
</tr>
<tr>
<td>5.24</td>
<td>Canard pitching moment coefficient for the 0% canard command level</td>
</tr>
<tr>
<td>5.25</td>
<td>Canard yawing moment coefficient for the 0% canard command level</td>
</tr>
<tr>
<td>5.26</td>
<td>Helicity contours for the 0% command level at a series of roll angles: (a) 0°, (b) 45°, (c) 90°, (d) 135°, (e) 180°, (f) 225°, (h) 270°, and (i) 315°</td>
</tr>
<tr>
<td>5.27</td>
<td>Comparison of the helicity contours along the length of the missile</td>
</tr>
<tr>
<td>5.28</td>
<td>Comparison of the helicity contours immediately behind the canards</td>
</tr>
<tr>
<td>5.29</td>
<td>Experimental canard command level</td>
</tr>
<tr>
<td>5.30</td>
<td>Missile configuration with free-spinning tail</td>
</tr>
<tr>
<td>FIGURE</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>5.31</td>
<td>Fuselage surface grid for the missile with a free-spinning tail</td>
</tr>
<tr>
<td>5.32</td>
<td>Canard surface grid for the missile with a free-spinning tail</td>
</tr>
<tr>
<td>5.33</td>
<td>Tail surface grid for the missile with a free-spinning tail</td>
</tr>
<tr>
<td>5.34</td>
<td>Typical volume grid resolution for the free-spinning tail missile grids</td>
</tr>
<tr>
<td>5.35</td>
<td>Missile geometry near the tail illustrating (a) the original geometry and</td>
</tr>
<tr>
<td></td>
<td>the (b) geometric modification to the railings and fuselage for the UVI</td>
</tr>
<tr>
<td></td>
<td>method</td>
</tr>
<tr>
<td>5.36</td>
<td>UVI surface grid for the free-spinning tail missile</td>
</tr>
<tr>
<td>5.37</td>
<td>Modification to the fuselage and railing and the resulting UVI volume grid</td>
</tr>
<tr>
<td>5.38</td>
<td>Sliding interface surface for the free-spinning tail missile</td>
</tr>
<tr>
<td>5.39</td>
<td>Detailed view of the volume grid between the railing and the tail fin</td>
</tr>
<tr>
<td>5.40</td>
<td>Detailed view of the sliding interface surface between the railing and the</td>
</tr>
<tr>
<td></td>
<td>tail fin</td>
</tr>
<tr>
<td>5.41</td>
<td>Static surface pressure contours</td>
</tr>
<tr>
<td>5.42</td>
<td>Static helicity contours</td>
</tr>
<tr>
<td>5.43</td>
<td>Static pressure contours on the surface of the tail</td>
</tr>
<tr>
<td>5.44</td>
<td>Axial force coefficient for the $\omega = 2500$ rpm tail spin rate</td>
</tr>
<tr>
<td>5.45</td>
<td>Side force coefficient for the $\omega = 2500$ rpm tail spin rate</td>
</tr>
<tr>
<td>5.46</td>
<td>Normal force coefficient for the $\omega = 2500$ rpm tail spin rate</td>
</tr>
<tr>
<td>5.47</td>
<td>Rolling moment coefficient for the $\omega = 2500$ rpm tail spin rate</td>
</tr>
<tr>
<td>5.48</td>
<td>Pitching moment coefficient for the $\omega = 2500$ rpm tail spin rate</td>
</tr>
<tr>
<td>5.49</td>
<td>Yawing moment coefficient for the $\omega = 2500$ rpm tail spin rate</td>
</tr>
<tr>
<td>FIGURE</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>5.50</td>
<td>Crossflow velocity vectors near the tail for the UVI solution at the $\omega = 2500$ rpm tail spin rate</td>
</tr>
<tr>
<td>5.51</td>
<td>Crossflow velocity vectors near the tail for the Sliding1 solution at the $\omega = 2500$ rpm tail spin rate</td>
</tr>
<tr>
<td>5.52</td>
<td>Crossflow velocity vectors near the tail for the Sliding2 solution at the $\omega = 2500$ rpm tail spin rate</td>
</tr>
<tr>
<td>5.53</td>
<td>Cutting plane location for the crossflow velocity vector plots</td>
</tr>
<tr>
<td>5.54</td>
<td>Convergence history comparison for $\omega = 2500$ rpm tail spin rate case</td>
</tr>
<tr>
<td>5.55</td>
<td>Rolling moment coefficient versus spin rate for each methodology</td>
</tr>
<tr>
<td>5.56</td>
<td>Tail axial force coefficient for the $\omega = 2500$ rpm tail spin rate</td>
</tr>
<tr>
<td>5.57</td>
<td>Tail side force coefficient for the $\omega = 2500$ rpm tail spin rate</td>
</tr>
<tr>
<td>5.58</td>
<td>Tail normal force coefficient for the $\omega = 2500$ rpm tail spin rate</td>
</tr>
<tr>
<td>5.59</td>
<td>Tail rolling moment coefficient for the $\omega = 2500$ rpm tail spin rate</td>
</tr>
<tr>
<td>5.60</td>
<td>Tail pitching moment coefficient for the $\omega = 2500$ rpm tail spin rate</td>
</tr>
<tr>
<td>5.61</td>
<td>Tail yawing moment coefficient for the $\omega = 2500$ rpm tail spin rate</td>
</tr>
<tr>
<td>5.62</td>
<td>Helicity contours near the tail for the $\omega = 2500$ rpm tail spin rate</td>
</tr>
<tr>
<td>5.63</td>
<td>Comparison of the helicity contours near the tail</td>
</tr>
<tr>
<td>5.64</td>
<td>Full annulus grid of impeller with vane-island diffuser</td>
</tr>
<tr>
<td>5.65</td>
<td>Leading edge of the impeller blades</td>
</tr>
<tr>
<td>5.66</td>
<td>Trailing edge of the impeller blades</td>
</tr>
<tr>
<td>5.67</td>
<td>Sliding interface on cutting plane of the compressor</td>
</tr>
<tr>
<td>5.68</td>
<td>Computed velocity contours across the sliding interface</td>
</tr>
<tr>
<td>FIGURE</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>5.69</td>
<td>188</td>
</tr>
<tr>
<td>5.70</td>
<td>189</td>
</tr>
<tr>
<td>5.71</td>
<td>190</td>
</tr>
<tr>
<td>5.72</td>
<td>191</td>
</tr>
<tr>
<td>5.73</td>
<td>192</td>
</tr>
<tr>
<td>5.74</td>
<td>193</td>
</tr>
<tr>
<td>B.1</td>
<td>213</td>
</tr>
<tr>
<td>B.2</td>
<td>214</td>
</tr>
<tr>
<td>B.3</td>
<td>215</td>
</tr>
<tr>
<td>B.4</td>
<td>216</td>
</tr>
<tr>
<td>C.1</td>
<td>218</td>
</tr>
<tr>
<td>C.2</td>
<td>218</td>
</tr>
<tr>
<td>C.3</td>
<td>219</td>
</tr>
<tr>
<td>C.4</td>
<td>219</td>
</tr>
<tr>
<td>C.5</td>
<td>220</td>
</tr>
<tr>
<td>C.6</td>
<td>220</td>
</tr>
<tr>
<td>C.7</td>
<td>221</td>
</tr>
<tr>
<td>C.8</td>
<td>221</td>
</tr>
<tr>
<td>FIGURE</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>C.9</td>
<td>Normal force coefficient for the Exp-fit canard command level</td>
</tr>
<tr>
<td>C.10</td>
<td>Rolling moment coefficient for the Exp-fit canard command level</td>
</tr>
<tr>
<td>C.11</td>
<td>Pitching moment coefficient for the Exp-fit canard command level</td>
</tr>
<tr>
<td>C.12</td>
<td>Yawing moment coefficient for the Exp-fit canard command level</td>
</tr>
<tr>
<td>D.1</td>
<td>Component contributions: axial force coefficient $\omega = 2500$ rpm</td>
</tr>
<tr>
<td>D.2</td>
<td>Component contributions: side force coefficient $\omega = 2500$ rpm</td>
</tr>
<tr>
<td>D.3</td>
<td>Component contributions: normal force coefficient $\omega = 2500$ rpm</td>
</tr>
<tr>
<td>D.4</td>
<td>Component contributions: rolling moment coefficient $\omega = 2500$ rpm</td>
</tr>
<tr>
<td>D.5</td>
<td>Component contributions: pitching moment coefficient $\omega = 2500$ rpm</td>
</tr>
<tr>
<td>D.6</td>
<td>Component contributions: yawing moment coefficient $\omega = 2500$ rpm</td>
</tr>
<tr>
<td>D.7</td>
<td>Axial force coefficient for the $\omega = 5000$ rpm tail spin rate</td>
</tr>
<tr>
<td>D.8</td>
<td>Side force coefficient for the $\omega = 5000$ rpm tail spin rate</td>
</tr>
<tr>
<td>D.9</td>
<td>Normal force coefficient for the $\omega = 5000$ rpm tail spin rate</td>
</tr>
<tr>
<td>D.10</td>
<td>Rolling moment coefficient for the $\omega = 5000$ rpm tail spin rate</td>
</tr>
<tr>
<td>D.11</td>
<td>Pitching moment coefficient for the $\omega = 5000$ rpm tail spin rate</td>
</tr>
<tr>
<td>D.12</td>
<td>Yawing moment coefficient for the $\omega = 5000$ rpm tail spin rate</td>
</tr>
<tr>
<td>D.13</td>
<td>Axial force coefficient for the $\omega = 1980$ rpm tail spin rate</td>
</tr>
<tr>
<td>D.14</td>
<td>Side force coefficient for the $\omega = 1980$ rpm tail spin rate</td>
</tr>
<tr>
<td>D.15</td>
<td>Normal force coefficient for the $\omega = 1980$ rpm tail spin rate</td>
</tr>
<tr>
<td>D.16</td>
<td>Rolling moment coefficient for the $\omega = 1980$ rpm tail spin rate</td>
</tr>
<tr>
<td>FIGURE</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>D.17</td>
<td>Pitching moment coefficient for the $\omega = 1980$ rpm tail spin rate</td>
</tr>
<tr>
<td>D.18</td>
<td>Yawing moment coefficient for the $\omega = 1980$ rpm tail spin rate</td>
</tr>
</tbody>
</table>
CHAPTER I

INTRODUCTION

Advances in computer technology and state-of-the-art numerical procedures have enabled time-accurate, high-resolution solutions for very large, complex stationary geometries in a relatively modest amount of time. The ability to perform dynamic simulations for complex geometries that involve relative motion remains a computationally intensive problem and a significant challenge. The goal of this work is to develop and demonstrate an unstructured grid approach for high-fidelity aerodynamic simulations of geometries with relative motion.

Time accurate prediction of flow fields about geometries in relative motion is of great interest. Practical applications include turbomachinery, helicopters, tiltrotors, and ship propellers. Liu and Hill [1] conducted an investigation comparing three approaches for simulating the unsteady flow of a centrifugal compressor, including the Frozen Rotor model, Circumferential Average model, and a sliding mesh model, and concluded that although the sliding mesh model is computationally more intensive, it is the necessary approach to predict the inherently unsteady flow field. Only the sliding mesh model is capable of simulating the aerodynamic interaction due to the impeller rotation relative to either the upstream guide inlet vanes or downstream discharge vanes. Advances in solution algorithms have increased the efficiency of time accurate Euler and Navier-
Stokes calculations using an unstructured grid approach and made the solution of these problems feasible. One obstacle that remains for unsteady simulations is how to handle relative grid motion.

Three main approaches have been devised to treat moving bodies or grids with relative motion. One approach is to remesh the domain, either locally or the entire domain. Another is to deform the grid in response to the relative motion of the body. Both allow fully general motion of the components. However, for a third approach, a large class of problems, like those considered herein, the direction of relative motion of the components is known *a priori*. Most rotating machinery problems fall into this class of problems, including axial and centrifugal turbomachinery, mixing tanks, ship and aircraft propellers, etc. Thus for the third approach, the remeshing can be avoided and the grid motion can be accomplished by decomposing the domain into subdomains which move relative to one another along judiciously chosen boundaries or interfaces. In this last approach, the problem then becomes how to couple the two subdomains across the chosen boundary or interface. In the past, there has been much attention devoted to this coupling of subdomains at an interface, even for problems that do not involve relative grid motion.

### 1.1 Structured Topology Grid Motion Strategies

All three approaches have been devised and implemented for structured grid topologies to solve problems involving relative motion. McDonald [2] used a distorted moving grid approach involving grid generation at every time step for the relative
motion. Janus and Whitfield [3] [4] utilized localized grid distortion also requiring local regridding every time step to pass information between zones that move relative to one another in a prop-fan simulation. In their work, the local grid movement near the interface is determined using relative angular velocity and the solution time step. The movement results in localized grid distortion on one side of the interface until a multiple of the azimuth spacing is reached, at which time the grid lines change partners or “click” across the interface.

There are two types of subdomain methods (or zonal methods in structured topology nomenclature), depending upon whether or not the subdomain (or zonal) boundaries align or arbitrarily intersect. Patched grids align at the boundaries and overlapped or overset grids arbitrarily intersect. Rai [5] [6] performed a systematic study on the conservative treatment of patched zonal finite difference or structured grids. Walters et al. [7] and Klopfer [8] extended this approach to cell-centered finite volume schemes. Wei and Chen [9] extended this approach to allow curved surfaces at the zonal boundaries. Note that in references 5 - 9, there is no relative grid motion and the focus is on the treatment of the interface.

In the Chimera or structured composite overset approach [10], grid components are not required to align with neighboring components in any special way, can overlap one another, and there is generally a multiple cell overlap. Valid Chimera holes must be cut in each grid within regions that overlap with other grid components, solid bodies, or any other non-flow regions. The Chimera holes serve to identify active and non-active parts of the overall grid. Interpolation stencils have to be created for all intergrid
boundary points. Identification of the interpolation stencil involves a search for donor cells for all points that lie along the Chimera hole boundary. At the intergrid boundaries, a non-conservative interpolation is typically used to transfer information across the interface. Meakin [11] showed that the formal order of accuracy is maintained with a non-conservative Chimera approach using simple interpolation. Overset grid topologies have been successfully applied to geometrically complex configurations involving moving bodies and relative grid motion [12] [13]. For these types of problems, the hole cutting and donor cell identification must be done each time the grid changes and can be computationally expensive. Research has recently begun to develop a conservative interface for Chimera grids [14] [15]. Wang et al. [14] show that on a sufficiently fine mesh, a conservative Chimera and a non-conservative Chimera scheme converge to the same solution.

Rai et al. [16] [17] demonstrated the feasibility of a sliding grid zonal approach using a system of patched and overlaid grids for time-accurate two-dimensional thin-layer Navier-Stokes (TLNS) simulations of rotor-stator interactions. Combinations of overlapped and patched grids were rigidly rotated to represent the motion of the rotor relative to the stator, and non-conservative bilinear interpolation was used to pass information across the interface. This sliding zonal technique was extended to three-dimensional TLNS simulations, again using non-conservative zonal boundary conditions [18, 19]. Chen and Chakravarthy [20] utilized a similar patched sliding-zone interface using a simple volume-weighted piecewise-constant projection of flow variables from one zone to another. Rumsey [21] also used a similar patched sliding-zone interface and
examined the effects of the non-conservative interface, time step, and speed of the moving zone on the transmission of acoustic waves and used a locally one-dimensional non-conservative interpolation [22]. Rumsey determined that at least 40 – 80 time steps are required for the sliding zone to pass one period of the spatial variation to avoid unreasonable distortions of the acoustic waves. Eliasson et al. [23] present a completely general patched sliding-zone interface again using bilinear interpolation that removes restrictions due to grid topology.

**1.2 Unstructured Topology Grid Motion Strategies**

Unstructured grids have become increasingly popular for computational fluid dynamic (CFD) simulations due to their flexibility, robustness, and ease in rapidly discretizing complex geometrical configurations. There are three main approaches that have been devised when dealing with moving bodies or relative grid motion for unstructured grids. One approach is to completely remesh all or part of the domain as the bodies move [24] [25]. Another is to deform the grid in response to the relative motion of the body and remesh when the deformation skews the grid lines and deteriorates the grid quality. It has been observed that frequent remeshing may lead to poor numerical results due to the interpolation of data from one grid to another [26]. Even if confined to a local portion of the domain, the remeshing procedure can become computationally expensive. Batina [27] [28] proposed a linear spring analogy to move or deform the grid in response to the motion of the body. However the linear spring analogy may lead to elements having negative volumes if grid lines cross. Farhat and Degand [29] [30]
proposed a torsional spring analogy to overcome this difficulty. While both of these techniques allow fully general motion of the components, depending upon the motion, the grid movement may eventually deteriorate the grid quality and remeshing all or part of the domain becomes necessary. Recognizing the need for eventual remeshing, Jayaraman and Marcum [31] developed an efficient, general algorithm for arbitrary motion that utilizes grid deformation, local reconnection, and local remeshing. A window region surrounding the body is identified and the grid deformation takes place within the region, except for within the boundary layer, which is allowed to move rigidly with the body. Local regeneration within the window is performed when the deformed grid violates specific quality criterion. For all of these procedures, the additional expense of remeshing is of particular concern in periodic problems where the solution must be extended several cycles in order to establish the periodicity of the solution.

To avoid the effort and expense of a deforming mesh or remeshing procedure, Nakahasi et al. [32] and Lohner et al. [33] have extended the structured overset or Chimera approach to unstructured topologies to treat problems with moving bodies and relative grid motion. The same hole cutting and donor cell identification procedures as previously discussed are required, but implemented on an unstructured topology. Initially the approach was implemented for inviscid simulations and later extended to viscous simulations [34] [35]. Similar to the structured overset approach, the unstructured approach is not conservative across the intergrid boundaries. Zhang et al [36] have proposed a combination of the previous two approaches: an unstructured overset method that combines grid movement and remeshing. Once the holes have been
cut in the overlapping regions, instead of the intergrid interpolation, the grid is locally remeshed in the hole regions to maintain conservation. However, due to changes in the grid from one time step to the next, the method is not strictly conservative as will be discussed in Section 1.3. In addition to the expense of identifying the holes, the method also incurs the additional expense of remeshing.

Compared to the sliding grid techniques previously discussed for structured grid topologies, the interfaces for unstructured grids are more complex because of the explicit data structure required for connectivity. Even though the grids do not align along the zonal boundaries for the structured grids, the interpolation is relatively easy to implement due to the structured nature of the grids with its implicit connectivity data structure. Unstructured grids, however, result in interfaces that are also unstructured and therefore the interface communication procedures are more complex. Pan et al. [37] developed a sliding grid approach suitable for inviscid computations with global flux conservation across subdomain interfaces. The approach is similar to the structured patch grid, but for use in an unstructured solver. The drawback to this method is that certain symmetry restrictions are placed on the grid interface in order to maintain global conservation. This defeats the one of the main advantages of an unstructured approach; namely flexibility to handle complex geometries. Yu et al. [38] used a sliding interface on a hybrid unstructured grid for a rotor-stator analysis that imposes even further restrictions; requiring that the subdomain grids exactly align at the interface at all time steps. Since the grids at the interface match identically, no interpolation is required. However the spacing at the interface is dictated by the time step and a change in the time step requires
remeshing of the interface region and again one of the main advantages of an unstructured approach is defeated.

Mathur [39] has developed a more general sliding interface for unstructured grids that imposes no restriction on the node placement of the subdomain boundary interfaces and is also conservative. The method takes advantage of the cell-centered scheme for which it is implemented and requires no interpolation across the interface. The overlapping faces at the interface surfaces are replaced by a new set of faces formed by their intersections such that each new face has a unique cell neighbor on the opposite side. However, the method is restricted to sliding boundaries that are cylindrical or conical surfaces of revolution.

While not a method to treat relative grid motion, Blazek [40] [41] developed a conservative interface treatment specifically for coupling a cell-centered structured scheme with a node-centered unstructured scheme. The approach is to compute a flux at the interface using the unstructured solver and distribute the flux to the structured solver. Information is needed for both the left (unstructured) and right (structured) states to compute the flux at the interface for the unstructured solver. The information needed to compute the right state is found by interpolation and the gradient information from the structured solver is used to evaluate the left state.

1.3 UVI Methodology

The state-of-the-art for unstructured grid topologies was developed by Sreenivas et al. [42] for tilt-rotor simulations. This ingenious method, referred to as the UVI
method, employs local grid reconnection to enable relative grid motion. For example, to simulate a rotating propeller, a surface or interface is created to divide the domain into two subdomains: an inner domain representing the propeller and an outer domain representing the rest of the domain of interest. This is illustrated in Figure 1.1. To simulate the motion of the rotating propeller, the cells attached to either the inside or the outside of the interface are deleted, thus leaving a void in the domain. Next, the inner subdomain representing the propeller is rotated as a rigid unit into the desired orientation. Then a local reconnection process [43] [44] is performed to re-generate the deleted cells in order to merge the two subdomains, resulting in one continuous domain. The domain is continuous in the sense that the two subdomains are merged together sharing a common set of nodes at the interface boundary. The UVI method, in principle, is an unstructured implementation of the localized grid distortion and clicking method introduced by Janus [3]. The local-reconnection process reconnects the distorted grid lines at the interface, and the inner grid is essentially clicked into place.

Despite the fact the domain is continuous at the interface boundary, the approach is not strictly conservative. Venkatakrishnan and Mavriplis [45] point out that after the local-reconnection process, the solution, which is stored at the node locations, needs to be modified to satisfy the conservation in time requirement since the control volumes for the nodes or points involved in the reconnection may have changed discontinuously from the previous time level. They proposed a conservative, linearity-preserving interpolation procedure to modify the solution to account for the change in connectivity as well as other possible approaches to update the solution. However, for the UVI method as
Currently implemented, no adjustments are made to the solution after the reconnection process to account for the change in connectivity.

Figure 1.1 The interface surface (UVI surface) surrounding the propeller delineates the subdomains

A limitation of the UVI method is that the grid reconnection can only be done in isotropic regions of the grid, and cannot include any physical boundaries. For the propeller example, this requires that the shaft be cut and a gap created to allow the UVI surface to pass through the shaft as shown in Figure 1.2. This gap creates a void in the physical boundary of the shaft surface that is now part of the flow field. Thus, during the volume grid generation process, a volume grid will be generated within the gap as shown
in Figure 1.3. If the gap required by the UVI method is aligned with the free stream, the flow through this gap can pose numerical problems, particularly in supersonic flows. This reconnection limitation is an implementation issue of the reconnection process that can be resolved. However, an issue that cannot be resolved is the requirement that the grid spacing on the UVI surface must be nearly uniform in the circumferential direction in order for the grid reconnection to work successfully. This is inherent in the reconnection process to maintain adequate grid quality. This issue can pose difficulties for complex geometries where the distance between the UVI surface and neighboring surfaces is relatively small.
Figure 1.2 Geometric modification required to the shaft by the UVI Method

Figure 1.3 Volume grid inside the shaft due to the gap created by the UVI Method
Another limitation for the UVI method is the local reconnection grid generation process is a serial process and therefore utilizes only a single CPU. The overall unstructured grid generation process could be implemented to work in parallel. However, the local-reconnection of a thin layer about a UVI domain is not highly parallelizable. In a parallel computing environment, this creates a bottleneck by forcing all the remaining processors to remain idle and wait on a single processor to perform the local reconnection. This can significantly increase the run-time for simulations involving relative grid motion. In addition, since it is a serial process running on a single CPU, there are memory restrictions that limit the resolution of the UVI surface. To minimize the modification to the geometry, it is obviously desired to make the gap as small as possible. For the reconnection process to work best, the elements inside the gap should be nearly isotropic, which requires the grid spacing in the region of the gap to be approximately half the width of the gap. Thus, additional resolution is wasted on a part of the numerical domain that is not even present in the physical problem domain.

1.4 Objectives and Approach

The objective of this work is to develop a sliding interface method [46] that is fast and efficient and addresses some of the previously described limitations for simulations involving relative grid motion. To achieve these objectives, an approach that involves no grid deformation, remeshing, or hole cutting will be used. This approach has been implemented into a fully parallel, node-centered finite volume, unstructured viscous flow solver. The rotational motion was accomplished by rigidly rotating the subdomain
representing the moving component, which is similar to the UVI and structured patch and overlaid methods. The sliding interface method will not impose any restrictions on the subdomain interface. However for rotational motion, the surface must be axisymmetric as no hole cutting is done. At the subdomain interface boundary, the faces along the interface will be extruded into the adjacent subdomain to create new volume elements and forming a one-cell overlap. These new volume elements are to be used to compute a flux across the subdomain interface. The values of the solution variables and other quantities for the nodes created by the extrusion process will be determined by linear interpolation. The extrusion is done so that the interpolation will maintain information as localized as possible. Another notable feature is that the grid on the interface is arbitrary. The boundary between the two subdomains can have completely independent grids from one another; meaning that they do not have to connect in a one-to-one manner and no symmetry or pattern restrictions are placed on the grid. In addition, the interface surface may intersect boundaries and the method will be implemented in a fully parallel way to take advantage of high-performance computing clusters.

The sliding interface approach developed here is not limited to applications involving rotating components. It can be used when other types of grid movement is required. It could also be used when there is no grid movement, such as a case where the geometry is so complex that the grids must be constructed in separate pieces and assembled. The individual grids do not normally align or match at the subdomain
boundaries and this allows for fast grid generation and assembly for very complex geometries and domains.

1.5 Outline

The remainder of this work is divided into five chapters. Chapter II presents the equations of fluid motion as well as the numerical discretization and solution algorithms of the unstructured viscous flow solver into which this sliding interface method is implemented. Chapter III describes in detail the sliding interface method and algorithms used in its implementation. Six validation cases are presented in Chapter IV to examine the effect of the sliding interface on various types of flow conditions: steady, unsteady, viscous, inviscid, compressible and incompressible flows. Results and discussion for validation of the sliding interface method using three geometrically complex and large-scale applications are presented in Chapter V. A summary and conclusions are presented in Chapter VI.

Four appendices are also included that contain supplemental but necessary information. Appendix A documents the eigensystem of the inviscid flux Jacobian matrix and Appendix B contains additional information used in the volume search algorithm discussed in Chapter III. Appendices C and D contain additional force and moment coefficient time histories for the large-scale missile applications described in Chapter V.
CHAPTER II

NUMERICAL DISCRETIZATION AND SOLUTION ALGORITHM

The sliding interface methodology is incorporated into the U^2NCLE (Unstructured Unsteady Computation of Field Equations) flow solver [47] [48]. U^2NCLE is a parallel, finite-volume, flow simulation code that solves the unsteady Reynolds averaged Navier-Stokes equations for complex geometries represented by multi-element unstructured grids. This chapter is intended to provide an overview of the numerical techniques used by the U^2NCLE flow solver. Note there are multiple options for many of the techniques discussed in this chapter implemented in the flow solver and only those techniques used to generate the results described in Chapters IV and V will be covered here.

2.1 Governing Equations

For this work, the governing equations for fluid flow are assumed to be governed by the unsteady Reynolds averaged Navier-Stokes equations. The majority of the model problems and large-scale applications in Chapters IV and V pertain to compressible flows and therefore the derivation that follows will be concerned with compressible fluids.
Compressible fluids are invariably gases for which the principal body force, gravity, is negligible and therefore the body force will be omitted in the discussion.

The time-dependent three-dimensional compressible Reynolds averaged Navier-Stokes equations are presented here in conservative form in Cartesian coordinates and cast in an Arbitrary Lagrangian-Eulerian (ALE) frame of reference. The governing equations are given as a system of conservation laws which relate the time rate of change of mass, momentum and energy within a control volume to the net rate of flux of these quantities through the boundaries of the control volume. The derivation of the governing equations used in this study is based on the continuum assumption of a uniform, homogeneous fluid without mass diffusion, external heat addition, or finite-rate chemical reactions. The non-dimensional equations can be written in integral form as

\[
\frac{\partial}{\partial t} \int_{\Omega} Q dV + \int_{\partial \Omega} \vec{F}(Q) \hat{n} dA = \frac{M}{\text{Re}_L} \int_{\partial \Omega} \vec{G}(Q) \hat{n} dA \tag{2.1}
\]

where \( \hat{n} \) is the outward pointing unit normal vector to the control volume \( \Omega \). The dependent variable vector \( Q \) for the conservative variables of mass, momentum, and energy is

\[
Q = [\rho \quad \rho u \quad \rho v \quad \rho w \quad E]^T \tag{2.2}
\]

The convective or inviscid flux vector, \( \vec{F}(Q) \), is

\[
\vec{F} = \begin{bmatrix}
\rho (u - V_x) \\
\rho u (u - V_x) + P \\
\rho v (u - V_x) \\
\rho w (u - V_x) \\
\rho H (u - V_x) + V_x P
\end{bmatrix} \hat{i} + \begin{bmatrix}
\rho (v - V_y) \\
\rho u (v - V_y) + P \\
\rho v (v - V_y) + P \\
\rho w (v - V_y) \\
\rho H (v - V_y) + V_y P
\end{bmatrix} \hat{j} + \begin{bmatrix}
\rho (w - V_z) \\
\rho u (w - V_z) \\
\rho v (w - V_z) \\
\rho w (w - V_z) + P \\
\rho H (w - V_z) + V_z P
\end{bmatrix} \hat{k} \tag{2.3}
\]
and the viscous flux vector, $\vec{G}(\vec{Q})$, is

$$
\vec{G} = \begin{bmatrix}
0 & \tau_{xx} & \tau_{xz} \\
\tau_{yx} & 0 & \tau_{yz} \\
\tau_{zx} & \tau_{zy} & 0 \\
\end{bmatrix} \hat{i} + \begin{bmatrix}
0 & \tau_{xy} & \tau_{xz} \\
\tau_{yy} & 0 & \tau_{yz} \\
\tau_{zy} & \tau_{zz} & 0 \\
\end{bmatrix} \hat{j} + \begin{bmatrix}
0 & \tau_{xy} & \tau_{xz} \\
\tau_{yx} & 0 & \tau_{yz} \\
\tau_{zx} & \tau_{zy} & 0 \\
\end{bmatrix} \hat{k}
$$

(2.4)

where $\rho$ is fluid density, $\rho u$, $\rho v$, and $\rho w$ are the Cartesian momentum components in the $x$, $y$, and $z$ directions, respectively, $E$ is the total energy per unit volume, $H$ is the total enthalpy per unit mass given by $H = \frac{E + P}{\rho}$, $P$ is the static pressure, and $\hat{i}$, $\hat{j}$, $\hat{k}$ are the unit vectors in the $x$, $y$, and $z$ directions, respectively.

The variables in the preceding equations are non-dimensionalized using a characteristic length ($L$), and freestream values of velocity ($U_\infty$), speed of sound ($c_\infty$), density ($\rho_\infty$), and viscosity ($\mu_\infty$). Pressure is normalized with $\frac{P^*}{\rho_\infty c_\infty^2}$, where $P^*$ is the local dimensional static pressure. The non-dimensional parameters used in the preceding equations are the freestream Mach number, defined as $M_\infty = \frac{U_\infty}{c_\infty}$, and the Reynolds number, which is a ratio of momentum to viscous forces, and is defined as $Re_L = \frac{\rho_\infty U_\infty L}{\mu_\infty}$.

For the convective flux vector $F(\vec{Q})$, the velocity vector representing the flux crossing a control volume face is written relative to the speed of the mesh where $u, v, w$ are the absolute fluid velocity components and $V_x, V_y, V_z$ are the control volume face velocity
components in the \(x\), \(y\), and \(z\) directions, respectively. The velocity normal to a control volume face, or contravariant velocity, is thus defined as

\[
\Theta = u\hat{n}_x + v\hat{n}_y + w\hat{n}_z + V_{gs} = (u - V_x)\hat{n}_x + (v - V_y)\hat{n}_y + (w - V_z)\hat{n}_z \quad (2.5)
\]

where the grid speed is

\[
V_{gs} = -(V_x\hat{n}_x + V_y\hat{n}_y + V_z\hat{n}_z) \quad (2.6)
\]

and \(\hat{n}_x\), \(\hat{n}_y\), and \(\hat{n}_z\) are the components of the outward-pointing unit vector for the control volume face.

For the viscous flux vector \(G(\Theta)\), the components of the viscous shear stress tensor \(\tau_{ij}\) represent the stress acting on the \(i\) plane in the \(j\) direction. The fluid is assumed to be a Newtonian fluid and only the range of fluid behavior within local thermodynamic equilibrium is considered for which the Stokes hypothesis is valid. The traditional turbulence modeling assumption based on the Boussinesq analogy [49] is used, which is the turbulent shear is analogous to molecular shear. Using these assumptions, the normal stress components are

\[
\tau_{xx} = (\mu + \mu_t) \left[ 2 \frac{\partial u}{\partial x} - 2 \left( \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \right) \right] \quad (2.7)
\]

\[
\tau_{yy} = (\mu + \mu_t) \left[ 2 \frac{\partial v}{\partial y} - 2 \left( \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \right) \right] \quad (2.8)
\]

\[
\tau_{zz} = (\mu + \mu_t) \left[ 2 \frac{\partial w}{\partial z} - 2 \left( \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \right) \right] \quad (2.9)
\]
and the shear stress components are

\[ \tau_{xy} = \tau_{yx} = (\mu + \mu_t) \left( \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) \]  

(2.10)

\[ \tau_{xz} = \tau_{zx} = (\mu + \mu_t) \left( \frac{\partial u}{\partial z} + \frac{\partial w}{\partial x} \right) \]  

(2.11)

\[ \tau_{yz} = \tau_{zy} = (\mu + \mu_t) \left( \frac{\partial v}{\partial z} + \frac{\partial w}{\partial y} \right) \]  

(2.12)

For the energy equation, Fourier’s law for heat transfer by conduction is assumed. The components of the conductive heat flux vector \( \hat{q} \) are

\[ q_x = \frac{\partial \hat{q}}{\partial x} = \frac{1}{\gamma - 1} \left( \frac{\mu}{Pr} + \frac{\mu_t}{Prt} \right) \frac{\partial T}{\partial x} \]  

(2.13)

\[ q_y = \frac{\partial \hat{q}}{\partial y} = \frac{1}{\gamma - 1} \left( \frac{\mu}{Pr} + \frac{\mu_t}{Prt} \right) \frac{\partial T}{\partial y} \]  

(2.14)

\[ q_z = \frac{\partial \hat{q}}{\partial z} = \frac{1}{\gamma - 1} \left( \frac{\mu}{Pr} + \frac{\mu_t}{Prt} \right) \frac{\partial T}{\partial z} \]  

(2.15)

where \( \gamma \) is the specific heat ratio, \( \mu \) and \( \mu_t \) are the fluid (laminar) molecular and turbulent eddy viscosities and \( Pr \) and \( Pr_t \) are the fluid (laminar) and turbulent Prandtl numbers. The Prandtl number is a measure of the energy dissipated by friction to the energy transported by thermal conduction and is defined as \( Pr = \frac{\mu c_p}{k} \), where \( c_p \) and \( k \) are the specific heat at constant pressure and the coefficient of thermal conductivity, respectively. For this study, the working fluid is assumed to be air and for air at standard conditions, \( \gamma = 1.4, Pr \)
\[ \frac{\mu}{\mu_c} = \frac{(1 + C^*)^{\frac{\rho^*}{\gamma}}}{T^* + C^*} \] (2.16)

where \( C^* = \frac{198.6^\circ R}{T_{ref}} \), \( T^* = \frac{T}{T_{ref}} \), and \( T_{ref} = 460^\circ R \). The turbulent eddy viscosity is not a fluid property, but is a function of the flow conditions and geometry and is calculated using the one-equation Spalart-Allmaras turbulence model [51] described in Section 2.4.

The Navier-Stokes equations by themselves are an open model; they do not form a complete or closed set of equations. To close the system of fluid dynamic equations, it is necessary to relate the thermodynamic variables and to relate the fluid transport properties to the thermodynamic variables. The fluid is assumed to be a calorically perfect gas with constant specific heats. The system of equations is closed by using the equation of state for a perfect gas

\[ P = \rho RT \] (2.17)

where \( R \) is the ideal gas constant and \( T \) is temperature.

Writing the velocity in relative form by including the grid speed terms casts the governing equations in the Arbitrary Langrangian-Eulerian form to allow the use of moving and deforming grids. The ALE formulation [52] is a combination of Lagrangian and Eulerian methods that accounts for the relative motion of the grid with respect to the fluid. This method allows not only prescribed motions, but also arbitrary grid motion where the motion is not predetermined [53]. In the Eulerian frame of reference, the
computational grid is stationary and the fluid’s behavior at a fixed location (a control volume) is obtained and information about the flow in terms of what happens as the fluid flows through the control volume. In the Lagrangian frame of reference, the control volume is associated with individual fluid particles with local fluid velocity and the fluid particle is followed to obtain information as it moves about. With the contravariant velocity defined as in Equation 2.5, when the mesh is fixed and the mesh speeds are zero, the Eulerian form of the convective fluxes is recovered. If the mesh speeds equal the fluid velocity, then the contravariant velocity is zero and the Lagrangian form of the convective fluxes are recovered whereby the advective terms vanish.

The inclusion of the grid speed terms in the conservation of mass gives rise to an additional divergence term that must be satisfied. From the continuity equation with uniform conditions zero fluid velocity,

\[
\frac{\partial}{\partial t} \int_\Omega dV = \int_{\partial \Omega} V \cdot \hat{n} dA \tag{2.18}
\]

Equation 2.18 represents the Geometric Conservation Law (GCL) [54] in integral form and relates the time variation of the control volume to the area, orientations and velocities of the control volume faces. The GCL must be satisfied exactly to avoid creation of spurious mass source and sink terms caused by volume changes. Note that the GCL is automatically satisfied for non-deforming grids that simply rotate or translate as the shape of the control volume does not change with time.
2.2 Spatial Discretization

The governing equations are discretized using a finite volume technique. In the finite volume formulation, the integral expressions for the governing equations are solved directly rather than first being transformed to differential form. These integral expressions are valid for any control volume and so there is considerable flexibility in the definition of the control volume shape. The control volume definition used here is based on a vertex-centered approach where the control volume surrounds each vertex, or node, and the boundaries of the control volume are defined using a median dual. The median dual is constructed by connecting the centroid of each element to the midpoint of each edge as shown in Figure 2.1 for two dimensions. In three dimensions, the median dual for a given vertex is constructed by connecting the element centroids, face centroids, and edge midpoints of all elements sharing the vertex. The median-dual control volume results in a polyhedral hull around the vertex. The resulting dual mesh forms unique nonoverlapping control volumes that completely cover the domain and there is a one-to-one mapping between the control volume (dual) faces and edges in the original grid.

Equation 2.1 is valid for any control volume and so it can be applied to each control volume individually and the flux balance will be satisfied for the overall control volume. The surface integrals are approximated by a quadrature over each face of the control volume. The numerical evaluation of the spatial residual (the surface integral terms in Equation 2.1) is performed separately for the inviscid and viscous contributions.
2.2.1 Inviscid Contributions

The inviscid flux contribution is approximated using a midpoint quadrature of the flux over each surface of the control volume boundary. The surface integral in Equation 2.1 representing the inviscid flux contribution can therefore be approximated as

\[
\int_{\partial \Omega} \vec{F}(\vec{Q}) \cdot \hat{n} dA = \int_{\partial \Omega} \tilde{F}(\vec{Q}) d\vec{n} \approx \sum_{i=1}^{N} \tilde{\Phi}(Q_L, Q_R; \hat{n}) \times A_i
\]

where \(N\) is the number of edges incident to the vertex of interest, \(A_i\) is the area of the control volume face, and \(\tilde{\Phi}\) is the numerical flux vector on the control volume face. The numerical flux is evaluated from data on the left \((Q_L)\) and right \((Q_R)\) of the control volume face.

The manner in which the flux through the control volume face is computed comprises the approximation for the surface integration. The simplest approach is just an average of the left and right states, which is equivalent to an integration using the
trapezoidal rule. However, depending upon the Mach number, the governing fluid equations may be elliptic, hyperbolic, or parabolic in nature or a mixture of the three and the unsteady equations are hyperbolic in time. Equations that are hyperbolic and parabolic in nature have a limited domain of dependence, which means that the solution at a point does not depend on every other point in the field and implies that information travels only in certain characteristic directions. Any flux formulation should honor the direction of information propagation determined by the eigenvalues of the corresponding flux Jacobian matrix. Upwind differencing represents an attempt to include the mathematical and physical character of the equations in the difference expressions in order to more properly simulate the information within a flow field in characteristic directions.

In this study, the numerical flux on the control volume faces is calculated using Roe’s approximate Riemann solver [55]. The changes in the flux quantities at the interface have been interpreted as a series of one-dimensional Riemann problems that carries information in an upwind manner along the direction normal to the control volume faces [56]. Roe’s solver is termed an approximate Riemann solver because it solves an approximate Riemann problem exactly given the two solution states on each side of the control volume face.

The numerical fluxes on the control volume faces are constructed from data on either side of the face as

$$\Phi = \frac{1}{2} \left( \bar{F}(Q_L; \hat{n}) + \bar{F}(Q_R; \hat{n}) \right) - \frac{1}{2} |\hat{a}(Q_L, Q_R; \hat{n})| (Q_R - Q_L)$$  \hspace{1cm} (2.20)
where \( \tilde{F}(\rho, \hat{n}) \) and \( \tilde{F}(\rho, \hat{n}) \) are the inviscid flux vectors given by Equation 2.3 and are formed from the data on the left and right side of the face, respectively. The flux formulation in Equation 2.20 is only a first order method if the data on the left and right side of the control volume face are the values at the vertices on either side of the cell face. For a higher-order flux evaluation, a higher-order reconstruction of data to the left and right side of the control volume is required and this reconstruction is discussed in the next section. The Roe matrix, \( \tilde{A} = \tilde{T} \tilde{\Lambda} \tilde{T}^{-1} \), is a locally constant matrix representative of local interface conditions. The matrix \( \tilde{T} \) is a matrix constructed from the right eigenvectors of the flux Jacobian and \( \tilde{\Lambda} \) is a diagonal matrix whose entries contain the eigenvalues of the flux Jacobian. In Equation 2.20, the absolute value of the Roe matrix is constructed using the absolute values of the eigenvalues. The definition of the matrix of right eigenvectors, corresponding inverse, and the construction of the Roe matrix is included in Appendix A. The quantities denoted with the \( \sim \) character are constructed with variables using the Roe averaging procedure. The Roe averaged variables are

\[
\rho = \sqrt{\rho_L \rho_R} \quad \text{(2.21)}
\]
\[
\tilde{u} = \frac{\sqrt{\rho_L u_L + \rho_R u_R}}{\sqrt{\rho_L + \rho_R}} \quad \text{(2.22)}
\]
\[
\tilde{v} = \frac{\sqrt{\rho_L v_L + \rho_R v_R}}{\sqrt{\rho_L + \rho_R}} \quad \text{(2.23)}
\]
\[
\tilde{w} = \frac{\sqrt{\rho_L w_L + \rho_R w_R}}{\sqrt{\rho_L + \rho_R}} \quad \text{(2.24)}
\]
\[
\tilde{H} = \sqrt{\rho_L H_L} + \sqrt{\rho_R H_R} \over \sqrt{\rho_L} + \sqrt{\rho_R}
\]  \hspace{1cm} (2.25)

The eigenvalues of the Roe matrix are
\[
\tilde{\lambda}_{1,2,3} = \Theta, \quad \tilde{\lambda}_{4,5} = \Theta \pm \tilde{c}
\]  \hspace{1cm} (2.26)

where the Roe averaged speed of sound is given by
\[
\tilde{c}^2 = (\gamma - 1) \left[ \tilde{H} - \frac{1}{2} (\tilde{u}^2 + \tilde{v}^2 + \tilde{w}^2) \right]
\]  \hspace{1cm} (2.27)

and \( \Theta \) is defined as
\[
\Theta = \tilde{u} n_x + \tilde{v} n_y + \tilde{w} n_z - \left( \tilde{V}_x n_x + \tilde{V}_y n_y + \tilde{V}_z n_z \right)
\]  \hspace{1cm} (2.28)

where \( \tilde{V}_x, \tilde{V}_y, \tilde{V}_z \) are the arithmetically averaged grid speed terms between the left and right states. The solution of the linearized problem allows for discontinuous jumps and non-physical entropy violating expansion fans. An entropy fix proposed by Harten and Hyman [57] to avoid this non-physical expansion shock is used. The correction, which adds sufficient diffusion to remove the expansion discontinuity, is given by
\[
|\tilde{\lambda}| = \begin{cases} 
\frac{|\lambda|}{1 + \left( \frac{\lambda^2 + \varepsilon^2}{\varepsilon} \right)} & |\lambda| \geq \varepsilon \\
\frac{1}{2} \left( \frac{\lambda^2 + \varepsilon^2}{\varepsilon} \right) & |\lambda| < \varepsilon 
\end{cases}
\]  \hspace{1cm} (2.29)

where
\[
\varepsilon = \max[0, (\lambda - \lambda_L)(\lambda_R - \lambda)]
\]  \hspace{1cm} (2.30)
2.2.2 Higher Order Methods

To evaluate the numerical flux vector at the control volume face, it is necessary to extrapolate the solution at the vertices to the faces of the surrounding control volume. For first order spatial accuracy, the data on the left and right side of the control volume face are simply set equal to the values at the vertices on either side of the cell face. For second-order spatial accuracy, the assumption of piecewise constant distribution for the control volume is replaced by a piecewise linear distribution. The primitive variables, \( \tilde{q}^T = [\rho, u, v, w, p] \), are extrapolated to the faces of the control volume using a second-order Taylor series expansion about the central vertex. The linear reconstruction at the control volume face is given by

\[
\tilde{q}_f = \tilde{q}_0 + \nabla \tilde{q}_0 \cdot \tilde{r}
\]  

(2.31)

where \( \tilde{q}_f \) is the reconstructed function, \( \nabla \tilde{q}_0 \) is the gradient of primitive variables at the vertex, and \( \tilde{r} \) is the vector from the vertex to the midpoint of the edge (the quadrature point for the control volume face). The primitive variables are extrapolated rather than the conserved variables to avoid extrapolating nonphysical values, that is, to avoid negative values for pressure.
2.2.3 Gradient Estimation

The gradient of the solution at the vertices needed in Equation 2.31 is based on a least-squares approach that uses data from neighboring vertices to reconstruct a more accurate approximation to the solution at the control volume face. Figure 2.2 shows the two-dimensional unstructured stencil for vertex 0, however this discussion is equally applicable in three dimensions. The assumption is that the solution behaves linearly over the control volume and to that end, a truncated second order Taylor series for the solution at neighboring vertex \( i \) can be written as

\[
q_i = q_0 + (\vec{r}_i - \vec{r}_0) \cdot \nabla q_0 = q_0 + q_{x_0}(x_i - x_0) + q_{y_0}(y_i - y_0) + q_{z_0}(z_i - z_0)
\] (2.32)

Equation 2.32 can be written for each neighboring vertex and the resulting \( N \times 3 \) system of equations is
where \( N \) is the number of neighboring vertices. A weighted least squares estimate may be obtained by multiplying both sides of Equation 2.33 with a weighting constant specific to each equation. For inverse distance weighted least squares, the weighting constant \( \alpha \) is

\[
\alpha_i = \frac{1}{\| \hat{x}_i - \hat{x}_0 \|} \tag{2.34}
\]

For an unweighted least squares estimate, the constants are all unity.

Equation 2.33 typically represents an over-determined system of linear equations, \( Ax = b \), that can be solved in a least squares sense to obtain the gradients at the vertices. This system of equations may be solved in a least squares sense by premultiplying both sides by \( A^T \):

\[
A^T A x = A^T b \tag{2.35}
\]

Equation 2.35 represents a square system \((3 \times 3)\) with a unique least square minimizing solution. This equation could be solved directly (the normal equations approach), however Anderson and Bonhaus [58] point out that this approach is susceptible to numerical ill-conditioning, particularly for high aspect ratio grids suitable for viscous simulations. Instead, a QR decomposition [59] is used to decompose the coefficient matrix \( A \) into a product of an orthonormal matrix \( Q \) and an upper triangular matrix \( R \), \( A = QR \). The basis vectors needed to perform the QR factorization of \( A \) are found using the Gram-Schmidt process [60]. Substituting into Equation 2.35 yields
The solution to Equation 2.39 can be obtained by back substitution since $R$ is upper triangular.

Anderson and Bonhaus [58] have shown that the unweighted least squares formulation of the gradient more accurately estimates the solution at the control volume faces for highly anisotropic grids than an estimate computed using either inverse distance weights or Green’s theorem. It was also shown for computing the actual values of gradients, inverse distance weighting and Green’s theorem give very similar results, both of which are more accurate than unweighted least squares. Therefore, when reconstructing data on the control volume faces, the unweighted least squares formulation is used and when actual gradients are needed, for example when discretizing the viscous flux contributions, the weighted least squares formulation is used.

### 2.2.4 Flux Limiting

The linear reconstruction given in Equation 2.31 is not appropriate for problems that contain large discontinuities and steep gradients. Additional steps must be taken to prevent oscillations and overshoots from developing in the numerical solution. This is
accomplished by enforcing strict monotonicity in the reconstruction [61]. In this context, monotonicity means the reconstructed value does not exceed the minimum or maximum of the values at the neighboring vertices. Therefore, the slope of the reconstruction is reduced until monotonicity is restored and no new extrema are created.

Monotonicity is enforced during the reconstruction by

\[ \tilde{q}_f = \tilde{q}_0 + \phi \nabla \tilde{q}_0 \cdot \tilde{r} \]  

(2.40)

where variable \( \phi \) is a slope-limiting function that ideally ranges from one when reconstructions are smooth to zero in locations near steep gradients and discontinuities.

For this study, the limiter developed by Barth and Jespersen [62] is used to control oscillations and overshoots in the numerical solution. The limiter function \( \phi \) is defined as

\[
\phi_i = \begin{cases} 
\min \left( 1, \frac{q_i^{\text{max}} - q_0}{q_i^{\text{r}} - q_0} \right) & : q_i^{\text{r}} - q_0 > 0 \\
\min \left( 1, \frac{q_i^{\text{min}} - q_0}{q_i^{\text{r}} - q_0} \right) & : q_i^{\text{r}} - q_0 < 0 \\
1 & : q_i^{\text{r}} - q_0 = 0 
\end{cases}
\]

(2.41)

where \( q_0^{\text{min}} \) and \( q_0^{\text{max}} \) are the minimum and maximum values of \( q \), respectively, among neighboring vertices adjacent to vertex 0, including vertex 0 itself, \( q_i^{\text{r}} \) is the unlimited reconstructed value at the neighboring vertex on the opposite side of the control volume face, and \( \phi_i \) is the limiter value for the face in that direction.
2.2.5 Viscous Contributions

The viscous flux vector, $\vec{G}(Q)$ in Equation 2.4, which represents shear stress and heat transfer effects, contains terms involving velocity and temperature gradients. A finite volume technique with a direct approximation for the gradients at the quadrature points is used to discretize the viscous contributions. The viscous fluxes are evaluated directly at each edge midpoint using separate approximations for the normal and tangential components of the gradient vector to construct the gradients required in the viscous fluxes:

$$\nabla Q_{ij} \approx \nabla Q_{ij,norm} + \nabla Q_{ij,tan}$$

(2.42)

where the $ij$ subscripts refers to vertices $i$-$j$ that define the edge (or control volume face). This approach, termed the directional derivative technique [63], uses a directional derivative along the edge to approximate the normal component of the gradient and the average of the nodal gradients to approximate the tangential component of the gradient. The normal and tangential components of the gradient are computed as

$$\nabla Q_{ij,norm} = \left(\nabla Q_j \cdot \hat{s}\right)\hat{s} \approx \frac{Q_j - Q_i}{\Delta s} \hat{s}$$

(2.43)

$$\nabla Q_{ij,tan} = \left(\nabla Q_j \cdot \hat{t}\right)\hat{t} \approx \overrightarrow{\nabla Q} - \left(\overrightarrow{\nabla Q} \cdot \hat{s}\right)\hat{s}$$

(2.44)

where $\hat{s}$ is a unit vector in the direction of edge $i$-$j$, $\hat{t}$ is a unit vector in a direction normal to the edge, $\overrightarrow{\nabla Q}$ is the average of $\nabla Q_i$ and $\nabla Q_j$, and $\Delta s = \bar{x}_j - \bar{x}_i$. Substituting Equations 2.43 and 2.44 into Equation 2.42 gives the final form of the directional derivative approximation to the gradient.
\[
\n\nabla Q_y \approx \nabla \bar{Q} + [Q_y - Q_y - \nabla \bar{Q} \cdot \frac{\Delta s}{|\Delta s|^2}] \frac{\Delta s}{|\Delta s|^2} \tag{2.45}
\]

As previously discussed, the weighted least squares method is used to evaluate the nodal gradients in Equation 2.45.

### 2.3 Temporal Discretization

A semi-discretized form of the governing equations described by Equation 2.1 can be written as

\[
\int dV + \sum_{i=1}^{N} \tilde{F}(Q) \cdot \hat{n}A_i = \sum_{i=1}^{N} \tilde{G}(Q) \cdot \hat{n}A_i \tag{2.46}
\]

where \( Q \) is assumed to be constant over the control volume of interest. Evaluating the integral in Equation 2.46 and discretizing the temporal derivative yields a fully discretized form of the governing equations

\[
V_i \frac{Q^{n+1} - Q^n}{\Delta t} + \sum_{i=1}^{N} \tilde{F}(Q) \cdot \hat{n}A_i = \sum_{i=1}^{N} \tilde{G}(Q) \cdot \hat{n}A_i \tag{2.47}
\]

or

\[
V \frac{\Delta Q^n}{\Delta t} = R(Q) \tag{2.48}
\]

where \( \Delta Q^n = Q^{n+1} - Q^n \). \( V \) is the volume of the control volume surrounding the vertex of interest, and \( R \) is the spatial residual containing all of the discrete approximations to the inviscid and viscous contributions. The time level at which the right hand side of Equation 2.48 is evaluated results in either an explicit or implicit scheme. The evaluation of the spatial residual at time level \( n \) results in an explicit scheme and evaluation at time
level $n + 1$ results in an implicit scheme. For advancing the solution in time, a general differencing scheme is used [64] that employs a two-parameter family of algorithms and encompasses both explicit and implicit schemes and is given by

$$
\Delta Q^n = \frac{1}{1 + \varphi} \frac{\Delta t}{V} \left[ \theta R^{n+1}(Q^{n+1}) + (1 - \theta) R^n(Q^n) \right] + \frac{\varphi}{1 + \varphi} \Delta Q^{n-1} + O\left[ \left( \theta - \frac{1}{2} - \varphi \right) \Delta t^2 + \Delta t^3 \right]
$$

(2.49)

where $n$ is the current time level, $\Delta t$ is the time step, and $\Delta Q^n = Q^{n+1} - Q^n$. The two parameters, $\theta$ and $\varphi$, determine the accuracy of the scheme. Some of the implicit time differencing schemes represented are first order backward Euler ($\theta = 1, \varphi = 0$), second order three-point backward Euler ($\theta = 1, \varphi = \frac{1}{2}$), and Crank-Nicholson ($\theta = \frac{1}{2}, \varphi = 0$). Equation 2.49 also includes explicit time differencing schemes such as forward Euler ($\theta = 0, \varphi = 0$). In this work, the second order backward Euler implicit differencing scheme is used for the unsteady problems and the first order backward Euler scheme for the steady-state cases.

For a moving or deforming grid, there is an explicit dependence on time for the volume of the control volume. To account for grid deformation, the following two identities are used [65]:

$$
\Delta Q^n = \left( \overline{QV} \right)^{n+1} - \left( \overline{QV} \right)^n = V^{n+1} \Delta \overline{Q^n} + \overline{Q^n} \Delta V^n
$$

(2.50)

$$
\Delta Q^{n-1} = \left( \overline{QV} \right)^n - \left( \overline{QV} \right)^{n-1} = V^{n-1} \Delta \overline{Q^{n-1}} + \overline{Q^{n-1}} \Delta V^{n-1}
$$

(2.51)

where $\overline{Q}$ is the integral average over the control volume, i.e. $\overline{Q} = \frac{Q}{V}$. Inserting these two identities into Equation 2.49 and rearranging yields
\[
\frac{V^{n+1} \Delta \overline{Q} - \phi \frac{V^{n-1} \Delta \overline{Q}}{1 + \phi}}{\Delta t} + \overline{Q} \left[ \Delta V^n - \phi \frac{\Delta V^{n-1}}{1 + \phi} \right] + \frac{1}{1 + \phi} \left[ \theta R^{n+1} + (1 - \theta) R^n \right] = 0
\]

(2.52)

The first term in brackets in the preceding equation is a temporal discretization of the change in the volume of the control volume and represents the Geometric Conservation Law previously discussed in Section 2.1. The GCL arises from the governing equations to account for a deforming grid and relates the rate of change of the control volume to the motion of the faces. The GCL stems from the requirement that the computation of the control volume, or of the grid velocities, must be done in such a way that the resulting numerical scheme is able to maintain a state of uniform flow, independently of the deformation of the grid [66]. The solution of the volume conservation equation (GCL) must be performed in exactly the same manner as the flow equations to ensure that the GCL is satisfied, otherwise spurious mass source and sink terms caused by volume changes may be created [54] [65].

2.3.1 Time Advancement Scheme

Equation 2.48 represents a system of nonlinear equations to be solved for \( Q^{n+1} \). This requires that the spatial residual must be evaluated at time level \( n+1 \), however as previously stated, the solution state at time level \( n+1 \) is unknown. Newton’s method is used to solve this system of nonlinear equations where the linearization is expanded about the known solution \( Q^n \). A nonlinear homogeneous system of equations
represented by \( L(Q^{n+1}) = 0 \), where \( L \) is a vector function, can be solved using Newton’s method by

\[
\frac{\partial L(Q^{n+1,m})}{\partial Q^{n+1,m}} \Delta Q^{n+1,m} = -L(Q^{n+1,m}) \quad (2.53)
\]

where \( m \) is the Newton iteration index and

\[
\Delta Q^{n+1,m} = Q^{n+1,m+1} - Q^{n+1,m} \quad (2.54)
\]

The Newton iterations are initialized using the previous converged time level solution, \( Q^{n+1,0} = Q^n \).

Using Equation 2.52, define the vector function \( L \) as

\[
L(Q^{n+1}) = \frac{(1 + \varphi)N^{n+1} - \varphi N^n}{\Delta t} \Delta Q^n + \frac{1}{1 + \varphi} \left[ \theta R^{n+1} + (1 - \theta) R^n \right]
\]

(2.55)

Next, linearize this function about \( Q^{n+1,m} \) by expanding \( L(Q^{n+1}) \) in a first order Taylor series about a known level \( n+1,m \) to obtain

\[
L(Q^{n+1,m+1}) \approx L(Q^{n+1,m}) + \frac{\partial L(Q^{n+1,m})}{\partial t} \Delta t \quad (2.56)
\]

Since \( Q \) is an implicit function of time, apply the chain rule to the partial derivative term and replace the resulting \( \frac{\partial Q^{n+1}}{\partial t} \) term with a first order temporal difference results in

\[
L(Q^{n+1,m+1}) = L(Q^{n+1,m}) + \frac{\partial L(Q^{n+1,m})}{\partial Q} \Delta Q^{n+1,m} \quad (2.57)
\]
where $\Delta Q^{n+1,m} = Q^{n+1,m+1} - Q^{n+1,m}$. The quantity $L(Q^{n+1,m+1})$ is the function that is to be driven to zero by the Newton iteration and thus Equation 2.57 becomes

$$\frac{\partial L(Q^{n+1,m})}{\partial Q} \Delta Q^{n+1,m} = -L(Q^{n+1,m})$$

(2.58)

Substituting Equation 2.55 into the above expression and differentiating results in

$$\begin{bmatrix}
(1 + \varphi) V^{n+1} I + \theta \frac{\partial R^{n+1,m}}{\partial Q^{n+1,m}} \\
\frac{\partial R^{n+1,m}}{\partial Q^{n+1,m}}
\end{bmatrix} \Delta Q^{n+1,m} = $$

$$- \begin{bmatrix}
(1 + \varphi) V^{n+1} \Delta Q^{n} - \varphi V^{n+1} \Delta Q^{n-1} \\
\frac{\partial R^{n+1} + (1 - \theta) R^{n}}{\partial Q^{n+1,m}}
\end{bmatrix} + \frac{1}{1 + \varphi} \begin{bmatrix}
(1 + \varphi) \Delta V^{n} - \varphi \Delta V^{n-1} \\
\theta R^{n+1} + (1 - \theta) R^{n}
\end{bmatrix}$$

(2.59)

The partial derivative of the residual with respect to the dependent variable vector, the $\frac{\partial R}{\partial Q}$ term, is the flux Jacobian matrix. A discussion of the flux Jacobian matrix is included in Section 2.3.3.

The right hand side of Equation 2.59 includes a discretization of the time derivative, as well as the spatial derivative. Thus if the right hand side of the above expression is driven to zero for a given time level using multiple Newton iterations, the time dependency of the problem is naturally taken into account and the resulting solution satisfies the unsteady governing equations and a time accurate solution is achieved. Multiple Newton iterations are used to rid the solution of time linearization error at a given time level and typically three to five Newton iterations are sufficient. For a steady solution, the time derivative goes to zero and Equation 2.59 is solved using the first order
(in time) backward Euler scheme with only a single Newton iteration. This is equivalent to a temporal linearization of the spatial terms only.

2.3.2 Linear System Solution Technique

Through the use of Newton’s method, the original nonlinear system of equations, Equation 2.48, results in a linear system of equations, Equation 2.59, which must be solved every Newton iteration. Equation 2.59 can be written as $Ax = b$, where the bracketed term on the left hand side represents the coefficient matrix $A$, $\Delta \overline{Q}$ represents the vector of unknowns $x$, and the right hand side represents $b$. The coefficient matrix $A$ is generally a large, sparse, non-symmetric matrix, although the matrix is block symmetric. This implies that the algorithms developed for positive-definite matrices cannot be used to solve Equation 2.59 and therefore iterative methods offer a viable solution strategy.

The resulting linear system is solved using a symmetric Gauss-Seidel algorithm. The coefficient matrix $A$ is split into a diagonal, upper triangular, and lower triangular parts, $A = [D + U + L]$, and the symmetric Gauss-Seidel sweeps may be written as the following two-step process per iteration:

\begin{align*}
[L + D] \Delta \overline{Q}^{k+\frac{1}{2}} + [U] \Delta \overline{Q}^{k} &= R^{n+1,m} \\
[U + D] \Delta \overline{Q}^{k+1} + [L] \Delta \overline{Q}^{k+\frac{1}{2}} &= R^{n+1,m}
\end{align*}

(2.60) (2.61)

where $k$ is the subiteration index. The diagonal part represents the contribution from the vertex of interest and the upper and lower triangular parts represent contributions from
the neighboring vertices. The subiterations are needed to reduce the error in the solution caused by the splitting of the linear system and typically eight subiterations are used. Since a Gauss-Seidel method, which is a relaxation solution technique, is used to solve the linear system of equations resulting from each Newton iteration, the overall scheme for solving the system of unsteady nonlinear equations is referred to as Newton-Relaxation [67].

2.3.3 Jacobian Calculations

The Jacobian matrices arise from the differentiation of the residual vector with respect to the dependent variable vector. The viscous terms are linear and make direct contributions to the Jacobian matrix. The discretization of the inviscid terms is nonlinear, and in order to be consistent with the flux formulation, the same formulation should be used with the Jacobian matrix. However, using a higher order formulation for the Jacobian matrix results in a larger stencil and larger memory requirements, and thus the Jacobian matrices are approximated using only first order accurate fluxes in the Jacobian computations.

The approximate Roe Jacobian is constructed by taking the derivative of the numerical Roe flux given in Equation 2.20 and is re-written here in a slightly different form:

\[
F = \frac{1}{2} (\bar{F}(Q_L) + \bar{F}(Q_R)) - \frac{1}{2} \left[ \tilde{A}^{+}(\bar{Q}_L, \bar{Q}_R) - \tilde{A}^{-}(\bar{Q}_L, \bar{Q}_R) \right] (Q_R - Q_L) \quad (2.62)
\]
where $\tilde{A}^+$ and $\tilde{A}^-$ are formed using contributions from only the positive and negative eigenvalues, respectively. The $\tilde{\cdot}$ quantities are again evaluated using Roe averaged variables. Assuming that the Roe matrix is constant, then the partial derivative of the numerical flux with respect to $Q_L$ and $Q_R$ is given by

$$\frac{\partial F}{\partial Q_L} \approx \frac{1}{2} \left( A(Q_L) + \tilde{A}^+ - \tilde{A}^- \right) = \frac{1}{2} \left( A(Q_L) + \tilde{A} \right)$$ (2.63) 

$$\frac{\partial F}{\partial Q_R} \approx \frac{1}{2} \left( A(Q_R) - \tilde{A}^+ + \tilde{A}^- \right) = \frac{1}{2} \left( A(Q_R) - \tilde{A} \right)$$ (2.64)

where $A$ is simply the convective flux Jacobian matrix, $\frac{\partial F}{\partial Q}$.

Recall that the objective of Newton’s method is to drive the right hand side of Equation 2.59 to zero and it does not matter how that is done. Setting the Jacobians to identity matrices results in an explicit solution process. This implies that the choice of Jacobians does not affect the final solution itself, but rather affects parameters used to obtain the solution such as rate of convergence, allowable time step, etc. Therefore, not only may different orders of accuracy be used when computing the Jacobians, but different flux formulations may be used as well. As an alternative to the approximate Roe Jacobian, the Jacobians resulting from Steger-Warming flux vector split (FVS) theory [68] are also implemented. The derivation and elements of the Jacobian matrix are given in reference 69. If the Steger-Warming Jacobians are used, then the right-hand side of Equation 2.59 operates with Roe’s scheme (Roe’s scheme is flux difference splitting or FDS) while the left-hand side operates with FVS theory. Despite the inconsistency, good results are still obtained. This is due to the way Equation 2.59 is
formulated. The right-hand side of Equation 2.59 contains the spatial and temporal
discretization terms and thus contains all of the “physics” of the problem and is
discretized using the more accurate FDS theory as opposed to FVS. The left-hand side of
the equation contains the “numerics” and thus any suitable approximation to the flux
Jacobian can be used and typically an approximation that offers more numerical
dissipation, such as the Steger-Warming approximation, will be more robust. The large-
scale missile applications in Chapter V used the Steger-Warming Jacobians due to the
extreme difficulty encountered with linearized Roe Jacobian matrices.

In a true Newton’s method implementation, the Jacobian matrices should be
recomputed or updated every Newton iteration. However, this is quite expensive. It has
been observed that updating the Jacobian matrices has a significant effect only when the
unsteadiness in the problem is quite significant. In practice, the Jacobians are usually
computed only between converged time levels and held fixed for the remaining Newton
iterations. However, for the missile applications in Chapter V, the unsteady effects
within one given time step are quite significant and the flux Jacobian matrices were
updated every Newton iteration.

2.4 Turbulence Modeling

As mentioned previously, the governing equations of fluid motion are assumed to
be the Reynolds averaged Navier-Stokes equations. This turbulent averaging process is
used to obtain the governing equations for the mean or time-averaged turbulent
quantities. The time-averaging removes the influence of the turbulent fluctuations while
preserving unsteadiness associated with other time-dependent phenomena that have time scales larger than those of turbulence. The time-averaging process also introduces new terms to the governing equations and these new quantities must be related to the mean flow variables through the use of turbulence models and additional assumptions.

In this work, the turbulence model is incorporated in a loosely-coupled manner where the mean flow equations are solved first and then the turbulence model is solved separately. Coupling between the mean flow and the turbulence model is accomplished since the turbulence model uses the most recently computed solution \(Q^{n+1}\), and the solution of the governing equations uses the most recently computed eddy viscosity \(\mu^n\).

In this study, the one-equation Spalart-Allmaras turbulence model is used to estimate the turbulent viscosity unless stated otherwise. The \(k-\omega\) [70], \(k-\varepsilon\) [71], and \(q-\omega\) [72] two-equation turbulence models are also implemented within the flow solver. However, it has been this author’s experience that for compressible external flows, the Spalart-Allmaras turbulence model has proven to be more robust than the other available turbulence models.

The Spalart-Allmaras turbulence model consists of a model equation for the transport of \(\nu\), which is then related to the turbulent kinematic eddy viscosity. The transport equation for \(\nu\) is given by

\[
\frac{\partial \tilde{\nu}}{\partial t} + \tilde{\nu} \cdot \nabla \tilde{\nu} = \mu_{b1} \left[ f_{r1} - f_{r2} \right] \tilde{S} \tilde{\nu} + \frac{M_x}{\sigma \text{Re}_L} \left\{ \nabla \cdot \left[ \left( \nu + (1 + c_{b2}) \tilde{\nu} \right) \tilde{\nu} - c_{b2} \tilde{\nu} \nabla^2 \tilde{\nu} \right] \right\}
\]

\[
- \frac{M_x}{\text{Re}_L} \left( c_{w1} f_{w} - \frac{c_{b1}}{\kappa^2} f_{r2} \right) \left( \frac{\tilde{\nu}}{d} \right)^2 + \frac{\text{Re}_L}{M} f_{n1} \Delta U^2
\]  \hspace{1cm} (2.65)
The left-hand side of Equation 2.65 represents the material derivative of \( \tilde{\nu} \) with the first term representing the time rate of change of \( \tilde{\nu} \) and the second term represents the convection. The first term on the right-hand side of the transport equation represents the production of turbulence, which is the rate at which kinetic energy is transferred from the mean flow to the turbulence. The second term represents turbulence diffusion and the third term represents turbulence destruction. The fourth term is the trip term, which is the specified location of transition from laminar to turbulent flow. The turbulent eddy viscosity is given by

\[
\nu_t = \tilde{\nu} f_{\nu t}
\] (2.66)

The variables in Equation 2.65 are defined as

\[
f_{\nu t} = \frac{\chi^3}{\chi^3 + c_{\nu t}^3}
\] (2.67)

\[
\chi = \frac{\tilde{\nu}}{\nu}
\] (2.68)

\[
\tilde{S} = S + \frac{M_c}{\text{Re}_L \kappa^2 d^2} f_{\nu 2}
\] (2.69)

\[
S = |\vec{\omega}| = |\tilde{\nabla} \times \vec{\nu}| = \sqrt{(w_z - v_z)^2 + (u_z - w_x)^2 + (v_z - u_y)^2}
\] (2.70)

\[
f_{\nu 2} = 1 - \frac{\chi}{1 + 2 f_{\nu t}}
\] (2.71)

\[
f_{w} = g \left( \frac{1 + c_{w 3}^6}{g^6 + c_{w 3}^6} \right)^{\frac{1}{6}}
\] (2.72)

\[
g = r + c_{w 3} (r^6 - r)
\] (2.73)
\[ r = \frac{M_w}{\text{Re}_T} \frac{\tilde{v}}{S \kappa^2 d^2} \]  \hspace{1cm} (2.74)

\[ f_{12} = c_{i3} \exp(-c_{i4} \kappa^2) \]  \hspace{1cm} (2.75)

where \( d \) is the distance to the nearest wall location and \( \kappa \) is the von Karman constant.

The trip function \( f_{1l} \) is

\[ f_{1l} = c_{i1} g_{i1} \exp \left[ -c_{i2} \left( \frac{\omega_t}{\Delta U} \right)^2 (d^2 + g_{i1} d_{i1}^2) \right] \]  \hspace{1cm} (2.76)

where \( d_{i1} \) is the distance from the field point to the trip, \( \Delta U \) is the difference between the velocity at the field point and that at the trip, \( \omega_t \) is the wall vorticity at the trip,

\[ g_{i1} = \min \left( 0.1, \frac{\Delta U}{\omega_t \Delta x} \right), \]  \hspace{1cm} and \( \Delta x \) is the grid spacing along the wall at the trip. For this work, the trip term is neglected which assumes the flow is fully turbulent from the leading edge of the viscous surface. The constants used in the above expressions are given as:

\[ \kappa = 0.41 \quad \sigma = \frac{\gamma}{5} \quad c_{v1} = 7.1 \]

\[ c_{b1} = 0.1355 \quad c_{b2} = 0.622 \]

\[ c_{w1} = \frac{c_{b1}}{\kappa^2} + \left( 1 + c_{b2} \right) / \kappa \quad c_{w2} = 0.3 \quad c_{w3} = 2.0 \]

\[ c_{i1} = 1.0 \quad c_{i2} = 2.0 \quad c_{i3} = 1.1 \quad c_{i4} = 2.0 \]

In order to solve the transport equation, Equation 2.65 must be discretized. A directional derivative method is used to discretize the diffusive terms that are inside of
the divergence operator and a pure upwind method is used to discretize the convective terms. The \( \tilde{v} \) term outside of the divergence operator is assumed to be constant within the control volume and the turbulence production and destruction terms are also evaluated assuming \( \tilde{v} \) is constant. At no-slip boundaries, \( \tilde{v} \) is specified to be zero. At farfield boundaries, \( \tilde{v} \) is set to 1/10 the freestream value [51] for the boundary face flux evaluation. The dependent variable is the freestream value for inflow and is extrapolated from the interior domain for outflow boundaries. After the transport equation is solved for \( \tilde{v} \), the turbulent eddy viscosity is evaluated as

\[
\mu_t = \rho v_t = \rho \tilde{v} f_v
\]

(2.77)

2.5 Boundary Conditions

The boundary conditions are enforced by applying an appropriate flux through the control volume faces which lie on the boundary and then following the same procedure as the interior control volumes to solve for the solution in the control volume. All boundary conditions are handled in an implicit manner. The boundary conditions are evaluated at time level \( n+1 \) and are linearized if necessary to form the contribution to the flux Jacobian matrix.

2.5.1 Inviscid Surface

An inviscid surface (or slip condition) is one where the velocity is tangent to the surface and the normal component of velocity is zero. This condition is enforced by
setting $\Theta$ to zero in the flux evaluation for the face, where $\Theta$ is the velocity normal to the control volume face (the contravariant velocity of Equation 2.5). By doing so, each of the momentum equations receives a contribution of $P_{\text{eff}} \hat{n}$. The effective pressure, $P_{\text{eff}}$, of the boundary triangle or quadrilateral is computed by taking a linear combination of the pressures of the neighboring vertices. The energy equation receives only a contribution from the surface wall velocity, $-P_{\text{eff}} V_{gs}$. The contributions to each equation are made to the flux vector directly.

2.5.2 Viscous Surface

Viscous surface conditions are enforced by directly specifying the desired wall velocity. If the mesh is non-stationary, then the grid speed terms, and thus the wall velocity, are non-zero. To maintain the desired wall velocity during the solution procedure, the linear system is modified such that no change is allowed in the velocity. An adiabatic condition is imposed by enforcing that temperature gradient normal to the viscous surface is zero, that is by enforcing $\nabla T \cdot \hat{n} = 0$.

2.5.3 Rotating Viscous Surface

A rotating viscous condition is also implemented to support rotating components in the domain. This condition is imposed by specifying the rotational vector for the desired rotating surface and the surface velocity is computed for every point on the specified surface. This computed surface velocity is then specified as the desired wall
velocity for the viscous boundary condition and the condition is enforced by the same procedure as described in the previous section for viscous surfaces.

2.5.4 Farfield Conditions

Farfield conditions are handled via a characteristic variable reconstruction. For a farfield surface, the flow field is assumed to be inviscid so that the characteristic reconstruction of the quantities needed for the flux computation is obtained from two locally one-dimensional Riemann invariants [73]. Although this assumption is not strictly valid, it is acceptable since the perturbations from the inviscid condition are generally very small provided the farfield surface is a sufficient distance away from the body of interest. The equations can also be derived from the characteristic equations in three-dimensional partial differential equation form [74].

The governing equations can be written in the direction normal to the boundary, \( \eta \), as

\[
\frac{\partial Q}{\partial t} + A \frac{\partial Q}{\partial \eta} = 0
\]  \( (2.78) \)

where \( A \) is the flux Jacobian matrix as defined previously. The flux Jacobian can be diagonalized using a similarity transformation as

\[
A = T \Lambda T^{-1}
\]  \( (2.79) \)

where \( \Lambda \) is a diagonal matrix containing the eigenvalues of \( A \), and \( T \) is a matrix whose columns are the right eigenvectors of \( A \). Refer to Appendix A for the construction of \( T \).
Pre-multiplying both sides of Equation 2.78 by $T^{-1}$ results in a decoupled hyperbolic partial differential equation:

$$\frac{\partial W_0}{\partial t} + \Lambda \frac{\partial W_0}{\partial \eta} = 0$$

(2.80)

where

$$W_0 = T_0^{-1}Q$$

(2.81)

and $W_0$ are the characteristic variables and $T_0^{-1}$ is evaluated at some constant conditions (Roe averaged conditions). The Riemann invariants are constant along characteristics normal to the surface (direction of information propagation) and the slopes of these characteristics in $x - \eta$ space are given by the eigenvalues. Recall, the eigenvalues are given by

$$\lambda_{1,2,3} = \Theta$$

(2.82)

$$\lambda_4 = \Theta + c$$

(2.83)

$$\lambda_5 = \Theta - c$$

(2.84)

where $c$ is the local speed of sound.

The objective is to determine the values of the characteristic variables on the boundary, $W_{0,b}$, in order to obtain the values of the solution vector on the boundary, $Q_b$.

For a subsonic outflow condition, the information associated with eigenvalues $\lambda_{1,2,3,4}$ originates from the interior of the computational domain and information associated with $\lambda_5$ originates from outside the computational domain. For a subsonic inflow condition, information associated with $\lambda_{1,2,3,5}$ propagates from outside the domain and information
associated with $\lambda_4$ originates from inside the domain. For supersonic outflow, all information originates from the interior of the domain. For supersonic inflow, all information originates from outside the computational domain.

Using Equation 2.81, the values of the characteristic variables on the boundary are

$$W_{0,b} = \begin{bmatrix} I_0^1 \cdot Q_1 \\ I_0^2 \cdot Q_2 \\ I_0^3 \cdot Q_3 \\ I_0^4 \cdot Q_4 \\ I_0^5 \cdot Q_5 \end{bmatrix}$$ \hfill (2.85)

where $I_0^i$ are the left eigenvectors, which are rows of $T_0^{-1}$, and $Q_i$ is the solution vector evaluated using either interior or exterior values based on the sign of the corresponding eigenvalue $\lambda_i$. The values of solution on the boundary are

$$Q_{0,b} = T_0 W_{0,b} = T_0 \begin{bmatrix} I_0^1 \cdot Q_1 \\ I_0^2 \cdot Q_2 \\ I_0^3 \cdot Q_3 \\ I_0^4 \cdot Q_4 \\ I_0^5 \cdot Q_5 \end{bmatrix}$$ \hfill (2.86)

With the value of the solution vector known at the boundary, the numerical flux through the boundary can be computed and it’s contribution added to the appropriate control volumes.
2.5.5 Symmetry Plane Boundary Conditions

A symmetry plane boundary condition is also available. It imposes that $\nabla \phi \cdot \vec{b} = 0$ for any arbitrary variable $\phi$ and $\vec{b}$ is the inward pointing normal to the plane of symmetry. Similar to the solid wall condition, no flow is allowed through a symmetry surface, so $\Theta$ is set to zero. The symmetry condition is imposed by enforcing the $\nabla \phi \cdot \vec{b} = 0$ constraint in any evaluation of gradient data. Thus, evaluation of viscous fluxes on a symmetry plane first has $\nabla \vec{V} \cdot \vec{b}$ removed. All gradient calculations have $\nabla \phi \cdot \vec{b} = 0$ for each component of the gradient calculated. For the inviscid terms, the solid wall condition described in Section 2.5.1 is used. Surface boundary normals for nodes on a symmetry surface are ensured to lie in the symmetry plane, since symmetry implies that an equal and opposite geometry segment exists.

2.6 Parallel Implementation

For quick turnaround time in a design environment, it is essential to parallelize the flow solution algorithm. The U$^2$NCLE flow solver is based on coarse-grained domain decomposition and each block in the domain is uniquely mapped to a given processor. The grid is partitioned using the METIS software package [75] [76] in a pre-processing step before the solution procedure begins. The solver employs MPI message passing [77] for interprocessor communication.

The iteration hierarchy [47] for the iterative implicit solution algorithm is shown in Figure 2.3. For relaxation solution techniques such as Gauss-Seidel, updating during
the linear subiterations allows recovery of a modified form of the original serial algorithm. The frequency of updating of the parallel interface $\Delta \tilde{Q}$'s determines the degree to which the algorithm is recovered. There is a choice regarding the frequency of the updates of $\Delta \tilde{Q}$ in order to approximately maintain the convergence rate of the serial implicit algorithm. One has the choice to update $\Delta \tilde{Q}$ after each linear subiteration or after each subiteration and directional sweep. There is also the choice to update after each color change, but for the symmetric Gauss-Seidel algorithm there is only one node color involved.

![Figure 2.3: U²NCLE flow solver iteration hierarchy](image)
CHAPTER III

SLIDING INTERFACE METHODOLOGY

This chapter outlines the methodology of the sliding interface. The idea behind the sliding interface is quite simple: extrude each interface so it overlaps the neighboring subdomain, and then use the extruded elements to close the control volume for the nodes on the interface. Information for the nodes of the extruded elements is obtained via interpolation from the neighboring subdomain, and with the control volume for the nodes on the interface now closed, a flux across the interface can be computed, and the flow field in that area resolved.

3.1 Sliding Interface

In order to simulate the flow field about a component moving, or rotating, relative to another component, the grid is divided into two subdomains, one rotating relative to the other. Note that both could be rotating, but for the sake of discussion one is assumed to be stationary. The idea used herein is to physically rotate one piece of the domain relative to the other, and then extrude the elements along the interface to couple the two subdomains.

Since the subdomains are discontinuous, the grid at the interface is arbitrary and thus the grids on either side of the interface do not normally align. There are two main
issues that must be addressed in order to compute the flow across the disjoint subdomain interface: (1) flux computation across the subdomain interface, and (2) subdomain coupling in the solution process. Before addressing these issues, the construction of the sliding interface will be discussed.

3.2 Construction of the Sliding Interface

Each subdomain boundary interface is extruded to form part of the sliding interface. Both subdomain interfaces are extruded and overlap in the region near the interface. In three-dimensions, triangular faces are extruded into prisms and quadrilateral faces into hexahedral elements. This is shown conceptually for a 2-D interface in Figure 3.1 where the edges on the interface are extruded into quadrilateral elements. For purely notational reasons, one subdomain interface is denoted the primary interface and the other the secondary interface. The one with the finer grid resolution is typically denoted the primary interface. Where the interface intersects boundary surfaces, such as a farfield boundary or a solid wall, the interface edges along the boundary are extruded into quadrilateral faces. These extruded faces will then have the appropriate boundary condition applied in order to make the appropriate boundary contribution to the flux evaluation for the node on the interface. The elements and nodes that are created by the extrusion are denoted as “sliding” elements and “sliding” nodes, respectively. These sliding elements close the control volumes for the nodes on the interface and behave just as interior control volumes. The extruded elements are only a mathematical abstraction
as they do not have to represent a physically possible situation as a whole. With the control volume closed, a flux can be computed.

![Two-dimensional interface](image)

**Figure 3.1** Two-dimensional interface where the interface edges are extruded into quadrilateral elements to form the overlapping elements

The sliding interface will be illustrated in three-dimensions using the propeller example from Section 1.3. The propeller and the location of the sliding interface are shown in Figure 3.2. The domain consists of two subdomains: an inner subdomain representing the propeller (primary interface) and an outer subdomain representing the rest of the domain of interest (secondary interface). An example of the extruded elements
on the primary sliding interface is shown in Figure 3.3 and the extent to which they overlap in the outer subdomain. Figure 3.4 shows the extruded elements of the secondary sliding interface and the extent to which they overlap the inner subdomain. As illustrated by these figures, the sliding elements on the outer domain interface surface (primary interface) are extruded to overlap with the inner rotating domain and the sliding elements on the inner rotating domain interface surface (secondary interface) are extruded to overlap with the outer domain.

Figure 3.2 Location of the sliding interface for the marine propeller example
Figure 3.3 Extruded sliding interface elements on the (a) primary interface (inner domain) and (b) the overlap into the outer domain
Figure 3.4 Extruded sliding interface elements on the (a) secondary interface (outer domain) and (b) the overlap into the inner domain
The extrusion distance is based on a local measure of the element sizes of the domain being extruding into. A distance metric for each node on the interface is computed as one half the average distance between face centroid and the element centroid sitting directly atop that face for every face attached to the node. This metric provides a local measure of the cell size for the domain being extruded from. In order to keep the information as localized as possible, information about the cell size of the neighboring subdomain is needed. This is accomplished by an information exchange. For example, for a node on the primary interface, the distance metric information is exchanged with the closest node on the secondary interface, and vice versa. After the information exchange, the actual extrusion distance for a node on the primary interface is one half the local distance metric of the closest node on the secondary surface and vice versa.

Since a flux across the interface needs to be computed, it is important to obtain the information to compute that flux from nodes as close as possible to the interface, and thus the need for the information exchange to get the extrusion distance based on the domain being extruded into. Note that in general when computing a flux for a face, the information used in the evaluation is local and obtained from surrounding nodes and thus the reasoning for keeping the extrusion close to the interface. For example, if the element size in one subdomain were significantly larger than the neighboring subdomain, then simply using the distance metric computed for the node would lead to an extrusion distance that might span multiple elements in the neighboring subdomain, as illustrated in Figure 3.5. In addition to obtaining information farther away from the interface, this
would also allow for the possibility of an extruded element lying inside a solid boundary if one were near the interface. In the vast majority of cases, the procedure described above leads to the extruded elements lying within the elements immediately adjacent to the interface in the neighboring subdomain.

![Figure 3.5](image)

**Figure 3.5** Extrusion distance based on the local subdomain can lead to the sliding elements extending too far from the interface

The closest node information serves as the basis for coupling the interfaces together and provides the locality information to couple the disjoint subdomains. The closest node information is found using a recursive-box algorithm [78] and it has been implemented to work in a parallel computing environment since the closest node may reside in another block. For every node on the primary interface surface, the nearest node
on the secondary surface is found and vice versa. The closest node information is also used to speed up the volume and surface searches when the element or face containing a point is needed. The starting location for these searches is taken to be an element or face attached to the closest node, and this ensures the search will complete in just a few iterations.

For nodes in the isotropic regions of the grid, the extrusion direction is based on the local node normal. The local node normal is simply the average of all the interface surface normals attached to the node. Special consideration is given when extruding into boundary layers. If the local node normal was used and the interface surface did not align with high aspect ratio boundary-layer grid as shown in Figure 3.6, then the sliding element might span multiple elements in the neighboring subdomain since the boundary-layer elements are extremely high aspect ratio. This would again lead to using non-local information when closing off the control volume. Instead, the extrusion direction is computed so it will align with the boundary-layer grid of the neighboring subdomain, as shown in Figure 3.7. This will ensure the information used to compute the flux will be as local as possible.

To compute the boundary-layer extrusion direction, two options were implemented. For the first option, the extrusion direction is taken to be tangential to the boundary layer elements of the domain being extruded into. For a point on the primary surface, a surface search is performed to find the containing face on the secondary surface and vice versa. The extrusion direction is the direction from the centroid of the containing face to the centroid of the element sitting directly atop the containing face. For the second option, the extrusion direction is taken to be parallel to the extrusion
direction of the closest boundary node. For a given interface surface, the nearest boundary node is found using the recursive-box algorithm for nodes in the boundary layer. The extrusion direction of the boundary layer nodes is taken to be the same as the extrusion direction of the nearest boundary node.
Figure 3.6 Boundary-layer extrusion direction based on local node normal does not align with adjacent subdomain boundary-layer grid.

Figure 3.7 Boundary-layer extrusion direction that aligns with adjacent subdomain boundary-layer grid and the resulting sliding elements.
Along boundary edges, the boundary nodes, which are nodes along the intersection of the sliding interface surface and any other boundary surface, are projected onto a boundary face of the neighboring subdomain to ensure they lie within the domain. A fast ray-triangle intersection algorithm [79] is used to determine which boundary face the node should be projected in order to ensure the extrusion direction of the boundary edges are tangent to the neighboring subdomain boundary.

To close the control volumes for the boundary nodes, the boundary edges are extruded into quadrilateral faces. The boundary condition applied to these extruded faces is the same as the boundary condition of the face that it overlaps in the neighboring subdomain.

The sliding interface can be constructed every time step using the previously described procedure or, to minimize computational costs, created only once during a preprocessing step. In which case, the extruded interface is rotated into place each time the grid is moved and there is no updating of the local extrusion distance or extrusion direction vector. This is appropriate if the grid spacing on the interface is nearly uniform in the circumferential direction.

### 3.3 Search Algorithm and Interpolation

The key for this approach to work efficiently in a parallel computing environment is the search algorithm. A search is performed to find the containing or host element of the sliding nodes in order to interpolate the needed quantities for the sliding node. For grids in relative motion to one another, the grids at the interface are changing every time
step and thus the host element of the sliding nodes changes and a search is required. Note that a search is required for every extruded point, or sliding node, on the interface.

The parallel search algorithm uses multiple passes of a volume coordinate search algorithm (or area coordinate search if a point lies on a surface) until all points are found. During the course of the search for a point, should the search algorithm attempt to move across a (parallel) block boundary, the point is transmitted to the appropriate block and the search is continued. As mentioned previously, to speed the volume search, a good starting location for each point is chosen so that the host element is found in one or two steps. The starting location is taken to be one of the elements attached to the closest node. Using this procedure, the majority of the points are found on the first pass, and only a few (the ones that are near or on block boundaries) require a second pass.

The basic strategy of the volume search algorithm (and corresponding surface search in 2-D) is based on the neighboring element algorithm described in reference 80. This approach exploits the simple properties of linear basis functions that at a given node, the function has a value of unity and decreases linearly in the direction opposite the face at which it is equal to zero. By construction, the linear basis functions have the property that they sum to unity, and thus a point lies within an element if and only if all of the basis functions have values between zero and one; otherwise the point does not lie within the element. The sign of the basis functions are used to determine which direction the search should continue. The element containing the point is in the direction of the most negative basis function, and so the element adjacent to the face opposite the node with the minimum value should be searched.
The basis functions for tetrahedral elements can be conveniently expressed using volume coordinates; and note that for a tetrahedral element, the number of nodes and faces are equal. Volume coordinates are the volume ratios of the sub-tetrahedra formed by connecting the nodes of each face to the search point, denoted point P. Area coordinates are the equivalent in two-dimensions and Figure 3.8 depicts the area coordinates for a triangular element as well as their signs depending upon the location of point P to indicate which direction the search should proceed. For a simplical tetrahedral element, the number of nodes and faces are equal. However, for a viscous mesh that contains five- and six-node pentahedral and/or hexahedral elements, this is not the case. To extend this concept to non-simplical elements, the concept of volume weights is used where the element is subdivided into tetrahedra to compute the face–based basis functions needed for the search. The volume weights for the each of the element types are provided in Appendix B.
A pseudocode example for the volume search algorithm using volume weights for the basis functions is given below:

```
initialize host_elem
found = false
BeginSearch:
  compute volume and volume weights, $\phi$, of host_elem
  flag host_elem as visited
  sort the volume weights and find minimum value, $\phi_{\text{min}}$
  if $\phi_{\text{min}} > -\text{tolerance} \times \text{volume}$ then
    found = true
  else
    do i = 1,nphi (loop over volume weights)
      ielem = neighboring element adjacent to face associated with $\phi_i$
      if ielem not flagged as visited then
        host_elem = ielem
        goto BeginSearch:
      end if
    end do
  end if
EndSearch:
Check value of found to determine if search successful
```
A relative tolerance is used for the containment test to avoid round-off problems associated with finite precision arithmetic when a point lies on a face or an edge. At boundary locations where the point lies outside the domain, the tolerance value is iteratively increased and the search repeated until the point is found.

In rare circumstances, a point may fail to be found due to boundary surfaces. If the point is not found after twenty passes of the parallel search, the point is broadcast to all blocks and the search proceeds by simply looping over each element until the point is found.

Except for the coordinates of the sliding nodes, all quantities \( \{\tilde{Q}, \nabla \tilde{Q}, \mu_r, \text{etc}\} \) are interpolated. The interpolation is done using the nodal values of the host element and weighting functions. For tetrahedral elements, finite element isoparametric shape functions are used. Given the containing element and coordinates of the sliding grid point, the shape functions can be easily computed. However for the non-simplical elements, computing the shape functions requires the solution of a cubic polynomial and can be computationally expensive. Instead for the pyramid, prism, and hexahedral elements, inverse distance weighting functions are used. An approach where these elements are subdivided into tetrahedral elements and then compute the finite element shape functions for the containing sub-tetrahedral has also been used. With the weighting functions computed, the value of the desired quantity for the sliding interface node is interpolated via

\[
\hat{u} = \sum_{j=1}^{n} \phi_j u_j
\]

(3.1)
where the $\hat{u}$ is the interpolated quantity, $\phi_j$ are the finite element shape functions or weighting functions for the containing element, $u_j$ is the value of the quantity at the nodes of the host element, and the summation is over the nodes of the host element. The interpolation involves all the nodes of the host element and as such there is no regard to upwinding or the direction of information travel or wave propagation. This will be taken into account during the flux calculation for the node on the interface surface.

The updating and interpolation for the values at the sliding nodes involves parallel communication since the host element may reside on another block. The block number of the host element is stored and communication lists are built and stored in order to transfer information between the appropriate blocks. The communication lists are just for the sliding interface nodes in order to minimize the communications costs and these lists are rebuilt every time the grid moves.

### 3.4 Flux Computation

With the values of the vector of conserved variables, $\bar{Q}$, known at the sliding node, the control volume for the node on the interface can now be closed and a flux computed. Regarding the flux computation, there are several approaches that could be taken when devising the sliding interface. One approach is to conserve the local flux at every point across the interface. Another approach is to conserve the total or global flux across the entire interface. Yet another approach is to compute a flux across the interface without regard to conservation. For this work, the flux is not explicitly conserved across
the interface. The interpolation is done for both subdomain interfaces and a flux is computed across each interface independently of the other.

Note that local flux conservation guarantees global conservation, but global conservation does not guarantee local conservation. If there is a discontinuity in the flow field, then only satisfying global conservation may not be of much use since the discontinuity may not be preserved, and so the globally conservative and the non-conservative approaches could lead to similar results. There is no way to impose local conservation across an interface without a fully connected interface and the only choices are global conservation or none at all. An exhaustive set of numerical simulations, including steady and unsteady, viscous and inviscid cases, have been performed to examine this issue of the non-conservativeness of the interface flux and for all the cases examined thus far, the present approach has performed well. Using this approach for the flux computation, excellent agreement has been obtained with both theoretical and experimental results. The results of these simulations are presented in the following chapters.

Regardless of whether the flux across the interface is conserved or not, a flux at the interface must be computed, and the problem remains how to compute this flux. Different options exist for computing this flux. One is that the flux itself could be directly interpolated. Another is to interpolate for the values of \( \mathbf{Q} \), the solution or conserved variable vector, and use these interpolated values to compute the flux. The latter approach is used here. With the flux computed, the residual can now be computed in order to compute a new value of \( \mathbf{Q} \) for the next time step.
3.5 Subdomain Coupling

The next issue to address is how to couple the disjoint subdomains during the solution process. During the solution process, \( \tilde{Q} \) and \( \Delta \tilde{Q} \) for the sliding nodes are not computed explicitly. Instead, these values are interpolated. As discussed previously, the approach proposed here is to extrude both subdomain interfaces and form overlapping elements. The overlapping interface allows \( \tilde{Q} \) for the physical nodes on one side of the interface to use information from the other side of the interface and couples the two subdomains. The degree of coupling depends on the interpolation frequency of \( \tilde{Q} \) and \( \Delta \tilde{Q} \).

Recall Section 2.6 describing the frequency of the updates of \( \Delta \tilde{Q} \) in order to approximately maintain the convergence rate of the serial implicit algorithm. In that context, the update refers to all nodes on parallel block boundaries, not just sliding interface nodes. Regardless of the update frequency choice, the updating or interpolation for the values of \( \tilde{Q} \) and \( \Delta \tilde{Q} \) of the sliding nodes is always done to ensure the tightest coupling possible despite the additional communication costs. The interpolation for \( \Delta \tilde{Q} \) is done at the end of each linear subiteration, color change, and directional sweep and the interpolation for \( \tilde{Q} \) is done at the end of each Newton iteration.
CHAPTER IV

MODEL VALIDATION PROBLEMS

Six different test cases have been devised to examine the effect of the sliding interface on the flow solution. They are: (1) a shear flow (steady, inviscid, incompressible), (2) flow through a diverging duct (steady, inviscid), (3) supersonic flow over a ramp (steady, inviscid), (4) a shock tube (unsteady, inviscid), (5) a cylinder in crossflow (steady and unsteady, viscous, incompressible), and (6) a marine propeller (unsteady, viscous, incompressible). Three cases have no relative motion: the supersonic flow over a ramp, the shock tube, and the cylinder in crossflow. These cases are used to determine if discontinuities can be maintained across the extruded interface. These three cases are inherently two-dimensional problems, but were simulated as three-dimensional problems. Unless otherwise stated, the grids were constructed so that there was a one-to-one correspondence of grid points at the interface. Note that this need not be the case, but was done in order to facilitate comparisons and clearly see the effect of the interface. Two grids were constructed; one for use as a baseline grid and one for use as a sliding interface. The volume grids are identical between the two. The only difference between the grids is that for the baseline grid, the nodes on the interface have been merged so that those nodes are common to both subdomains. For the sliding interface grid, the nodes on
the two overlapping surfaces at the subdomain boundary are not merged, which results in two discontinuous subdomains.

4.1 Steady, Inviscid, Incompressible Shear Flow

The first case is an incompressible, inviscid, steady shear flow designed to see if discontinuities can be maintained across the sliding interface. The domain consists of two subdomains as illustrated in Figure 4.1: an inner sphere and an outer cube. The spherical subdomain rotates relative to the stationary cube and the sliding interface boundary is the spherical surface. The upper half of the domain was given a uniform velocity of \( i\hat{2} \) and the lower half of the domain was given a uniform velocity of \( i\hat{1} \).

Grids were constructed for use with the UVI method and the sliding interface method. For both solutions, the spherical subdomain was rotated 1 degree per time step until the residual reached a nearly constant value. A contour plot of the velocity magnitudes of the shear flow near the subdomain boundary is shown in Figure 4.2. The contours are displayed on a cutting plane taken in the middle of the domain. Both solutions yield very similar results. As expected, a slight discontinuity is introduced in the sliding interface solution. However, the overall shear layer discontinuity is still preserved across the sliding interface boundary.
Figure 4.1 Shear flow surface grid
Figure 4.2 Shear flow velocity contours near the subdomain boundary for the (a) UVI solution and the (b) sliding interface solution
4.2 Steady, Inviscid, Supersonic Flow in a Diverging Nozzle

The second case is inviscid, supersonic flow through a diverging nozzle. This case was designed specifically to check the mass conservation. The inlet Mach number is 1.0 and the inlet radius is 1.0 and geometry of the nozzle is such that the area ratio \[ \frac{A_{exit}}{A_{inlet}} = 1.687 \] of the inlet and exit result in an exit Mach number of approximately 2.0 [81]. The configuration is shown in Figure 4.3. At the mid-span location, there is a spherical interface surface of radius 0.75. The radius of the nozzle at the mid-span location is 1.15 and thus approximately 40-50% of the flow passes through the interface.

![Figure 4.3 Inviscid, supersonic, diverging nozzle configuration](image)

Three cases were run; the baseline case and two cases employing the sliding interface. The baseline case was run assuming steady flow for 1000 iterations beginning from uniform conditions until the forces and residual remained constant. The first sliding interface case (Sliding) was run using the same conditions and spherical subdomain was held stationary. Even though the spherical subdomain is not rotating, the flow must still pass through the non-matching interface. During construction of the first sliding grid, the spherical subdomain was rotated 45 degrees about the x-axis to ensure that the primary
and secondary sliding interfaces would not align. For the second sliding interface case (Sliding_Spin), the spherical subdomain was rotated 1 degree per time step about the x-axis (the axial direction) and due to the grid movement, the simulation was run unsteady for nearly 25,000 iterations. Steady state conditions were achieved after approximately 9,000 iterations, however the simulation was continued to examine if the rotating sliding interface had any effect on the mass conservation.

The mass flow rate on the inlet and the outlet surfaces for each case are shown in Table 4.1. The mass imbalance is simply the difference in mass flow between the inlet and outlet surfaces. The percent difference listed in Table 4.1 for the sliding interface solutions are with respect to the baseline solution, \( \text{%Difference} = \left| \frac{\text{Baseline} - \text{Sliding}}{\text{Baseline}} \right| \). A plot of the mass conservation time history is shown in Figure 4.4 for the Sliding_Spin case. A detailed view over one revolution is shown in Figure 4.5. Note there is very little variation over one period and there are no large variations that are simply averaged out. A comparison of the components of the moment flux and energy flux were also made and revealed percentage differences on the order of 0.01 and smaller.

Table 4.1 Comparison of the mass flow across the inlet and outlet surfaces for the diverging supersonic duct

<table>
<thead>
<tr>
<th>Solution</th>
<th>Inlet</th>
<th>Outlet</th>
<th>Mass Imbalance</th>
<th>% Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>3.13064363e-2</td>
<td>3.13643160e-2</td>
<td>5.78797e-5</td>
<td>--</td>
</tr>
<tr>
<td>Sliding</td>
<td>3.13064363e-2</td>
<td>3.13642363e-2</td>
<td>5.78000e-5</td>
<td>0.14</td>
</tr>
<tr>
<td>Sliding_Spin</td>
<td>3.13064363e-2</td>
<td>3.13628801e-2</td>
<td>5.64444e-5</td>
<td>2.48</td>
</tr>
</tbody>
</table>
Figure 4.4 Mass conservation for the rotating sliding interface case

Figure 4.5 Mass conservation over one period for the rotating sliding interface case
4.3 Steady, Inviscid, Supersonic Ramp

The third case is inviscid, steady, supersonic flow over a 20° ramp at Mach 2. This case was designed to see if the resulting oblique shock wave would be affected by the interface. The interface is positioned approximately half-way up the ramp, as shown in Figure 4.6a. A third grid was constructed so that points along the interface did not align in order to evaluate the effect of the non-matching interface. The point spacing on the right of interface was set to 1.5 times that of the left, as shown in Figure 4.6b. Each case was run 1000 iterations until the solution converged using 2nd order spatial variable extrapolation and the ramp was assumed to be an inviscid surface. Convergence was achieved by decreasing the residual until it remained constant. The sides of the domain were assumed to be symmetry planes and the upper surface was an inviscid surface.

The properties behind the shock wave are listed in Table 4.2. The subscripts 1 and 2 denote properties upstream and downstream of the shock, respectively. Points 1 and 2 were selected to be on opposite side of the interface. The maximum error with respect to the exact solution [81] is less than 1%, where the error is defined as

\[ \text{Error} = \left| \frac{\text{Exact} - \text{Numerical}}{\text{Exact}} \right|. \]

<table>
<thead>
<tr>
<th>Solution</th>
<th>( M_2 )</th>
<th>( \rho_2/\rho_1 )</th>
<th>( P_2/P_1 )</th>
<th>( T_2/T_1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exact</td>
<td>1.211</td>
<td>2.041</td>
<td>2.841</td>
<td>1.392</td>
</tr>
<tr>
<td>Baseline</td>
<td>1.213</td>
<td>2.024</td>
<td>2.815</td>
<td>1.391</td>
</tr>
<tr>
<td>Aligned</td>
<td>1.210</td>
<td>2.026</td>
<td>2.820</td>
<td>1.392</td>
</tr>
<tr>
<td>Non-matching</td>
<td>1.202</td>
<td>2.027</td>
<td>2.836</td>
<td>1.399</td>
</tr>
</tbody>
</table>
A comparison of the density contours taken on cutting planes through the center of the grids is shown in Figure 4.7. The interface location is highlighted in Figure 4.7b for the grid using the aligned interface. As expected, a slight discontinuity is introduced by the interface. The discontinuity is more pronounced for the non-matching interface (Figure 4.7c), and this is due in part to the coarser resolution causing the contour lines to spread. The properties behind the shock are again obtained with relatively good accuracy.

In an effort to quantify how non-conservative the sliding interface is, the total mass flow was computed across the interface for the merged grid and also for both sliding interface grids. The mass flow and percent error relative to the baseline grid is listed in Table 4.3. As expected, the mass flow is not conserved exactly between the primary and secondary surfaces; however the error compared to the baseline grid is relatively small.

Table 4.3 Net mass flow across the interface for the 20° ramp at Mach 2

<table>
<thead>
<tr>
<th>Grid</th>
<th>Mass Flow</th>
<th>% Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>24.00750</td>
<td>---</td>
</tr>
<tr>
<td>Aligned Primary</td>
<td>24.00257</td>
<td>-0.02</td>
</tr>
<tr>
<td>Aligned Secondary</td>
<td>24.00926</td>
<td>0.01</td>
</tr>
<tr>
<td>Non-Matching Primary</td>
<td>24.01799</td>
<td>0.04</td>
</tr>
<tr>
<td>Non-Matching Secondary</td>
<td>24.00388</td>
<td>-0.02</td>
</tr>
</tbody>
</table>
Figure 4.6  Surface grids near the interface of the 20° ramp for the (a) aligned interface and the (b) non-matching interface
Figure 4.7 Density contours across the oblique shock wave for the (a) baseline, (b) aligned interface, and (c) non-matching interface solutions
4.4 Unsteady, Inviscid Shock Tube

The third case is an inviscid shock tube. There is no relative motion in this case, but it was designed to see how unsteady waves pass through the interface. The pressure ratio of the high-pressure gas (on the left of the diaphragm located at $\frac{x}{L} = 0.5$) to the low-pressure gas (on the right of the diaphragm) was 8:1 and the density ratio is 10:1. The extent of the domain is $L \times L \times 0.5L$ and all surfaces of the shock tube are assumed to be inviscid surfaces.

Four grids were constructed, two having planar interfaces and two having curved interfaces. For the first pair, there are two planar interfaces surfaces, one at $x/L = 0.4$ and the other at $x/L = 0.6$, as illustrated in Figure 4.8a. A second pair of grids was constructed having curved interfaces as shown in Figure 4.8b. For each pair, a baseline grid and a sliding interface grid were constructed. In the baseline grid, the points on the interface surfaces were merged to connect the subdomains and in the sliding interface grid, the points on the interface surfaces were not merged.

The solution for each case was run for 160 time steps using a time step of $\Delta t = 0.001$. At $t^* = 0.16$, the shock wave and contact surface have passed through the interface located at $x/L = 0.6$ and the interface at $x/L = 0.4$ is in the middle of the expansion wave. For the first pair of grids having planar interfaces, the entire wave passes though the interface at the same time. To determine if the shape of the interface affects the waves as they passes through the interface, the second pair of grids with curved interfaces were used. For the curved interface grids, different sections of the waves will be passing through the interface at different times.
A comparison of the computed three-dimensional solutions to the exact one-dimensional solution [81] is shown in Figure 4.9. The computed solution was taken along \((x,0.5L,0.25L)\). Note that even the baseline solution tends to smear the discontinuities. This can be resolved by increasing the number of points in the flow direction. The key idea to note is that all five computed solutions are nearly identical. The various waves were able to pass through all the different interfaces relatively undisturbed.
Figure 4.8  Shock tube grids for the (a) planar interface and the (b) curved interface
Figure 4.9 Comparison of pressure, density, and velocity for the theoretical and computed shock tube solutions

Shaded contours of density and velocity for each grid are shown in Figures 4.10 and 4.11, respectively. Overall, the solutions for the sliding interfaces match the baseline results very well. The curved sliding interface exhibits the same characteristics as the curved interface having the grid points merged on the interface. Both slightly shift the location of the contact wave as evident in Figure 4.10. They also shift the shock wave location and smear the velocity in the region between the shock and expansion.

The mass flow across the interface at x/L = 0.4 and at x/L = 0.6 was also computed for the shock tube grids. The maximum error compared to the merged baseline grids was less than 0.5%.
Figure 4.10 Density Contours for the (a) planar baseline, (b) planar interface, (c) curved baseline, and (d) curved interface solutions
Figure 4.11  Velocity Contours for the (a) planar baseline, (b) planar interface, (c) curved baseline, and (d) curved interface solutions
4.5 Incompressible Viscous Cylinder in Crossflow

The fourth case is low Reynolds number flow around a cylinder. The incompressible solver was used for two Reynolds numbers, Re=26 and Re=100. At these low Reynolds numbers, the flow is laminar. The three-dimensional grid is shown in Figure 4.12, and the interface bounds the trapezoidal region of tight grid spacing behind the cylinder. The interface lies just outside the region of viscous packing on the rear of the cylinder. Note there is no relative motion in this case, but it is used to examine the impact of the interface on the flowfield.

Figure 4.12 Grid near the cylinder for the cylinder in crossflow case
For the Re=26 case, the flow is steady and two stable vortices develop behind the cylinder as shown by the plot of the velocity vectors in Figure 13. The vectors in Figure 4.13 are shaded by velocity magnitude. The location of the vortex cores are shown in Table 4.4 and it can be seen that the vortices are located at virtually identical locations. The computed core location was based upon the minimum value of velocity in the recirculating region behind the cylinder. The experimental results were taken from reference 82. Note that the core location is located across the extruded interface and it is evident that the interface has minimal effect on the location of the vortices. The pressure contours are shown in Figure 4.14. A slight discontinuity can be seen in Figure 4.14b.

Table 4.4 Vortex core location for Re = 26 as measured aft of the cylinder from the centerline

<table>
<thead>
<tr>
<th>Grid</th>
<th>Core Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Experimental</td>
<td>(0.44, 0.50)</td>
</tr>
<tr>
<td>Baseline</td>
<td>(0.46, 0.49)</td>
</tr>
<tr>
<td>Interface</td>
<td>(0.46, 0.49)</td>
</tr>
</tbody>
</table>
Figure 4.13  Velocity vectors for Re=26 for the (a) baseline solution and the (b) sliding interface solution
Figure 4.14  Pressure contours for Re=26 for the (a) baseline solution and the (b) sliding interface solution
At the higher Reynolds number, the two vortices are no longer stable and they begin to alternately shed from the upper and lower part of the rear surface of the cylinder; hence, the flow is no longer steady. This case was selected to see if the vortices are convected downstream across the interface. The time history of the lift coefficient for nearly a period and a half is shown in Figure 4.15. The solution using the sliding interface is nearly identical to the baseline solution and consequently both result in the same estimation for the Strouhal number, \( St = 0.163 \), which compares well to the experimental value of \( St = 0.167 \) [49]. The instantaneous pressure contours are shown in Figure 4.16. Again, the solution using the extruded interface compares very well to the baseline solution. A slight discontinuity can again be seen in the pressure contours due to the interface.

![Figure 4.15](image.png)

**Figure 4.15** Comparison time history of the lift coefficient for the cylinder at \( Re=100 \)
Figure 4.16  Comparison of the instantaneous pressure contours for Re=100 for the (a) baseline solution and the (b) sliding interface solution
4.6 Unsteady, Incompressible Viscous Marine Propeller

The fifth case was a 3-bladed propeller used in hydrodynamic applications. Four grids were constructed: a baseline grid, a grid suitable for use with the UVI methodology, and two different sliding interface grids. The interfaces for the grids are shown in Figures 4.17a and 4.17b. To facilitate comparison, the domain for the baseline grid and the first sliding interface grid (Sliding1) was divided into an inner portion, which contains the shaft and the propeller, and an outer portion that contains the farfield domain. The only difference between these two grids is the coincident nodes at the sliding interface are not merged. The UVI and second sliding interface (Sliding2) grids were constructed in a manner more representative and indicative of its actual use. As shown in Figure 4.18, the interface surface of the Sliding2 grid intersects the shaft and passes through the boundary layer. The UVI grid is the same one used in the discussion of the UVI method in Section 1.3 and a cutting plane taken along the centerline is shown in Figure 4.19. For the baseline solution, the entire grid was rotated about the longitudinal axis to simulate the propeller. The rotational speed was 7.5 radians/sec and a time step corresponding to approximately 1.5 degrees of rotation per time step was used. A model scale Reynolds number of \( \text{Re}_D = 7.65 \times 10^5 \) based on the propeller diameter was used. The initial normal spacing was set at \( 5 \times 10^{-6} \) and lead to \( y^+ \) values of approximately 1.0 for the first point from the wall at the blade tips. For the UVI and sliding interface grids, only the inner domain containing the propeller was rotated and the outer domain was stationary. Note that for the UVI and Sliding2 case, only the part of the shaft within the interface surface is rotating. In order to be consistent with the previous two cases, a boundary condition was applied to the shaft in the outer stationary domain to simulate a spinning shaft.
Figure 4.17 Interface locations for the three-bladed marine propeller simulations for the (a) baseline and Sliding1 grids and (b) UVI and Sliding2 grids
Figure 4.18  (a) Cutting plane taken through the Sliding2 marine propeller grid and (b) detailed view illustrating the sliding interface surface intersecting the shaft
The time-accurate solutions were run for 2500 time steps, with the first 20 time steps taken as first-order (in space), until the axial force achieved a relatively constant value. The convergence history for each grid is shown Figure 4.20. Both the UVI and sliding interface methods converge the residual to approximately the same value and the residual for the UVI and both Sliding1 and Sliding2 solutions are slightly higher than the Baseline case. This was typical for all the cases presented here.
A comparison of the axial force coefficient is listed in Table 4.5. All three cases agree within 1% of the baseline case. The total mass flow across the interface was computed and the results are shown in Table 4.6. The mass flow difference for the Sliding1 grid is with respect to the baseline grid. For the Sliding2 grid, the mass flow difference is the difference across the primary and secondary interface for the UVI grid.

Table 4.5 Axial force coefficient for the three-bladed marine propeller

<table>
<thead>
<tr>
<th>Grid</th>
<th>CFx</th>
<th>% Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>-0.41971</td>
<td>---</td>
</tr>
<tr>
<td>UVI</td>
<td>-0.42237</td>
<td>0.63</td>
</tr>
<tr>
<td>Sliding1</td>
<td>-0.42082</td>
<td>0.26</td>
</tr>
<tr>
<td>Sliding2</td>
<td>-0.42198</td>
<td>0.30</td>
</tr>
</tbody>
</table>
Table 4.6 Net mass flow across the interface for the three-bladed propeller

<table>
<thead>
<tr>
<th>Grid</th>
<th>Mass Flow</th>
<th>% Diff</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>0.01172165</td>
<td>---</td>
</tr>
<tr>
<td>Sliding1 Primary</td>
<td>0.01142494</td>
<td>-2.53</td>
</tr>
<tr>
<td>Sliding1 Secondary</td>
<td>0.01172668</td>
<td>0.04</td>
</tr>
<tr>
<td>Sliding2 Primary</td>
<td>5.81994x10^{-3}</td>
<td>---</td>
</tr>
<tr>
<td>Sliding2 Secondary</td>
<td>6.08602x10^{-3}</td>
<td>4.57</td>
</tr>
</tbody>
</table>

Cutting planes through the center of the field were made and the axial velocity distributions across the interface are shown in Figures 4.21 and 4.22. The UVI solution does not appear to be as smooth as the baseline or sliding interface solutions. A discontinuity across the interface can be seen in Figures 4.22a and 4.22b, but overall, the velocity field for the sliding interface solution compares very well to the baseline solution. An additional comparison was made to quantify the error resulting due to the flux not being strictly conserved. The pressure on the interface surface was computed and compared to the baseline case for the Sliding1 grid. The percent difference results are shown in Figure 4.23 where the difference is computed as

\[
\% Difference = \left\lfloor \frac{\text{Baseline} - \text{Sliding}}{\text{Baseline}} \right\rfloor. 
\]

The maximum difference is 17% and occurs near the end of the shaft, where the pressure values are the smallest. The remaining values are less than 7%.
Figure 4.21 Centerline axial velocity contours for the (a) baseline and (b) the UVI solutions
Figure 4.22  Centerline axial velocity contours for the (a) Sliding1 and (b) Sliding2 solutions
Figure 4.23 Percent difference in pressure on the (a) primary sliding interface and (b) secondary sliding interface surfaces for the Sliding1 grid
CHAPTER V

LARGE-SCALE APPLICATION VALIDATION PROBLEMS

For the validation and evaluation of the sliding interface methodology, three configurations are presented to demonstrate the new method for geometrically complex problems with relative grid motion. The first case is a spinning missile with dithering canards, the second is a missile with a free-spinning tail, and the third is a high-speed centrifugal compressor. For each of these cases, the UVI method introduced geometry modifications or could not be used to solve the problem at all. Comparisons to the UVI solution will be made when possible.

5.1 Case 1: Spinning Missile with Dithering Canards

The spinning missile configuration used for the first case is shown in Figure 5.1. The missile has a hemispherical nose, cylindrical fuselage, a boat tail section, two moveable canards, and four stationary tail fins. The fuselage has a fineness ratio (length to diameter) of approximately 20. The four tail fins are located at 90-degree increments circumferentially around the tail section. The tail fins are designed to induce the entire missile to spin about its longitudinal axis; hence they are canted one degree to generate a left-handed roll. Yaw and pitch control is actuated using the two interconnected canards.
From their zero-deflection position, the canards can rotate ±15 degrees as shown in Figure 5.2. As the missile spins, the canards change pitch position to perform vehicle maneuvers. The motion of the canards is synchronous and specified by a canard command schedule. For the present work, two command levels were used. The command level shown in Figure 5.3, referred to as the $0\%$ command level, was used with a missile spin rate of $\Omega = 8.75$ Hz. The time required for the canards to move between the maximum and minimum deflection angles is very short, approximately 6.5 ms. Using this command level and roll rate, the body rolls approximately 20 degrees during the time required for the canards to traverse between the minimum and maximum deflection position.
Figure 5.1 Spinning missile configuration with dithering canards
Figure 5.2  Canard range of motion

Figure 5.3  Canard command schedule
The original motivation behind this simulation was to predict the aerodynamic performance of a spinning missile with moveable control surfaces and to demonstrate an unstructured grid approach for the high-fidelity aerodynamic simulations of complex geometries with relative grid motion. The results describing the required resolution for accurate prediction of the aerodynamic forces acting on the body and the relative importance of the viscous effects and different canard dither patterns are given in references [83] [84]. Those results were generated using the UVI method and were found to be in good agreement with other grid topology approaches that have been successfully applied to this same problem: Murman et al. [85] applied a Cartesian method and both Hall [86] and Nygaard et al. [87] used Chimera overset structured grid methods. For the purpose of this work, the results generated using the UVI method will serve to validate those using the new sliding interface method. A comparison of these results will be made to the other topology approaches when available. Since the canard-generated vortices must pass through the sliding interface and travel downstream, this represents a good case to examine the dissipation introduced by the sliding interface.

5.1.1 Grid Construction

Body-fitted mixed-element type unstructured grids were generated suitable for use with both the UVI and sliding interface methods. The volume grids were constructed using the Advancing-Front/Local-Reconnection (AFLR) grid generation technique [88] and SolidMesh was used to prepare the CAD geometry and generate surface grids [89].
The surface grid for the fuselage, canards, and boat tail are shown in Figures 5.4, 5.5, and 5.6, respectively. Both the UVI and sliding interface volume grids were constructed using essentially the same surface grids. The differences amongst the grids will be discussed shortly. The normal spacing of the first grid point from the body for all the volume grids was $1.0 \times 10^{-5}$, which led to a $y^+$ distribution of less than 1.0 over the missile surface, thus indicating good viscous sublayer resolution. An example of the field grid and boundary-layer resolution is shown in Figure 5.7.

An emphasis was placed on capturing the vortices generated by the canards and resolving them along the missile body in order to ultimately determine if they impinge on the tail fins. To help control the field resolution near the canard tips, an embedded surface was constructed around the missile to better control the point spacing and the growth of the cell sizes in this region. This allowed fine resolution for the field grid between this surface and the missile and a coarse field grid between this surface and the far-field. The embedded surface surrounding the missile was approximately three diameters (of the missile body) in width in order to encapsulate the vortices generated by the canards.
Figure 5.4 Fuselage surface grid for the spinning missile

Figure 5.5 Canard surface grid for the spinning missile
Figure 5.6 Empennage surface grid for the spinning missile
The dithering canards are attached to the missile fuselage via a short, tapered cylindrical stock and rotate about the axis of the stock. As discussed in Section 1.3, for the UVI method to simulate the relative motion of the canards, a modification to the geometry is required. In order for the UVI surface to completely encompass the canard, the canard stock must be cut as shown in Figure 5.8 for the UVI surface to pass through. This results in a small gap in the stock that is parallel to the freestream flow. The UVI surface for the port canard is shown in Figure 5.9 and a similar surface is created for the starboard canard. The subsequent volume grid is shown in Figure 5.10. As supersonic
flow passes through this small opening, strong expansions are created on the leeward side of the stock and the corresponding drop in pressure leads to numerical problems. To remedy the problem, a blockage was defined to turn off portions of the field grid using a tapered cylinder to represent the missing portion of the stock. Edges with both nodes inside the region definition were turned off and edges with only one node inside the region were treated as viscous edges.

Figure 5.8 Geometric modification to the canard stock for the UVI method
Figure 5.9 UVI surface for the port side canard

Figure 5.10 Modification to the canard stock and the resulting UVI volume grid
Two grids were constructed for use with the sliding interface method. In order to make a direct comparison with the UVI method, the first sliding interface grid is the same as the UVI grid except the nodes on the interface surfaces that surround the canards are not merged. The same cylindrical blockages were defined to represent the missing portions of the canard stock. The second sliding interface grid is constructed with no modifications to the canard stock. The sliding interface surface intersects the canard stock as shown in Figure 5.11. The same volume grid generation parameters were used as for the UVI grid so the field grids would be as similar as possible. The number of nodes and elements by type for each grid are listed in Table 5.1. Note the difference in the number of nodes and triangles between the UVI and Sliding1 grids is due to duplication of the interface surfaces.

<table>
<thead>
<tr>
<th>Grid</th>
<th>Nodes</th>
<th>Triangles</th>
<th>Quads</th>
<th>Tetrahedra</th>
<th>Pyramids</th>
<th>Prisms</th>
</tr>
</thead>
<tbody>
<tr>
<td>UVI</td>
<td>9,262,283</td>
<td>909,090</td>
<td>0</td>
<td>24,619,587</td>
<td>32,753</td>
<td>9,882,451</td>
</tr>
<tr>
<td>Sliding1</td>
<td>9,288,317</td>
<td>961,150</td>
<td>0</td>
<td>24,619,587</td>
<td>32,753</td>
<td>9,882,451</td>
</tr>
<tr>
<td>Sliding2</td>
<td>9,577,991</td>
<td>964,448</td>
<td>11,099</td>
<td>27,034,228</td>
<td>37,177</td>
<td>9,634,480</td>
</tr>
</tbody>
</table>
5.1.2 Simulation Conditions and Parameters

The spinning missile simulations were conducted using the following parameters: \( M_\infty = 1.6, \alpha = 3 \text{ deg.}, \Omega = 8.75 \text{ Hz}, \) and \( \text{Re}_L = 41.3 \times 10^6 \) based on the missile length. At these conditions, a particle travels approximately 36.6 body lengths in one revolution. The force coefficients are defined using the standard normalization by dynamic pressure

\[
C_F = \frac{\text{Force}}{\frac{1}{2} \rho_\infty U_\infty^2 S}
\]  

(5.1)
Similarly, the moment coefficients are

\[ C_M = \frac{\text{Moment}}{\frac{1}{2} \rho \omega U^2 \pi SL} \]  

(5.2)

where the reference area \( S \) was taken to be the cross-sectional area of the missile body and the reference length \( L \) was taken to be the length of the missile. The moments were computed about the missile center of gravity.

Preliminary simulations were performed to determine the proper solver parameters to ensure the unsteady solution was converged at each time step. Those convergence study results were presented in reference 84. For the results presented here, eight linear sub-iterations were used. Time step studies were also conducted, and it was found a \( \Delta t \) corresponding to 1.0 degree of roll per time step was sufficient. Convergence studies were also performed to determine the number of Newton iterations needed per time step and it was found that four Newton iterations were needed to converge the axial force.

5.1.3 Simulation Results

The flow field about this missile is characterized by complex vortical flow and shock structures. The cross-section of the canards are thin supersonic airfoils with sharp edges that generate shocks and expansions. The canards also generate strong horseshoe vortices that convect the length of the missile. The horseshoe vortex is generated due to the decrease in lift at the canard tips. Since the missile is spinning, the vortices tend to twist around the missile as they convect downstream. At this low angle of attack, the vortices are also close enough to the missile body to effect the force distribution on the
fuselage and tail. A detached bow shock is formed ahead of the blunt, hemispherical-shaped nose and a complex pattern of intersecting shocks and expansion are formed around the canards and tail fins.

The overall force and moment coefficient histories are shown in Figures 5.12 – 5.17 and the roll-averaged values are listed in Table 5.2, where $C_A$, $C_Y$, $C_N$ are the axial, side, and normal force coefficients, respectively, and $C_l$, $C_m$, $C_n$ are the rolling, pitching, and yawing moment coefficients, respectively. Note that the side force and yawing moment coefficients indicate the largest differences, however, these values are expected to be near zero and would show the largest differences. In coefficient history figures, only a single revolution is shown since the solution is periodic after one revolution. Both of the sliding interface solutions yield nearly the same solution as the UVI method. Recall that the Sliding1 and the UVI solutions are obtained using virtually the same grid and it is difficult to distinguish the difference between the two. The curve for the Sliding1 solution lies almost identically upon the curve for the UVI solution. A detailed view of the axial force coefficient is shown in Figure 5.18. There is a discernible difference between the Sliding2 solution and the UVI solution, which is to be expected, since there is no cut in the canard stock grid for the Sliding2 case. Figure 5.19 shows a comparison of the convergence history for each grid. Both the UVI and sliding interface methods converge the residual to approximately the same order of magnitude. Thus there does not appear to be any adverse effects due to the non-conservativeness of the flux evaluation in the sliding interface method.
Figure 5.12 Axial force coefficient for the 0% canard command level

Figure 5.13 Side force coefficient for the 0% canard command level
Figure 5.14  Normal force coefficient for the 0\% canard command level

Figure 5.15  Rolling moment coefficient for the 0\% canard command level
Figure 5.16 Pitching moment coefficient for the 0% canard command level

Figure 5.17 Yawing moment coefficient for the 0% canard command level
Figure 5.18 Detailed view of the axial force coefficient for the 0% canard command level

Figure 5.19 Convergence history comparison for the 0% canard command level
To provide additional insight, the forces from the Sliding2 solution were decomposed by component as shown in Figures C.1-C.6 in Appendix C. As expected, the fuselage was responsible for the majority of the axial force since it has the largest wetted surface area of any of the components. For the canard component, the plateau regions correspond to when the canard is at maximum deflection and thus has the maximum surface area exposed to the flow. Note that on the aft end of the missile, the axial force acting on the exhaust or base surface is acting in the opposite direction of the other forces. The forces acting on this surface are predominantly pressure forces, which is due to the pressure drag caused by the expansion at the aft end of the missile. Due to the orientation and symmetry of this surface, the forces acting on this surface are only in the axial direction and have no contribution to the side or normal forces or any of the moments.

The canards generate nearly all of the side force and consequently are responsible for nearly all of the yawing moment. Even though the fuselage is axisymmetric, since it is spinning it generates some side force due to the Magnus effect. The variation of the fuselage side force is in phase with that of the canards, indicating the strong vortices shed from the canards are modifying the pressure distribution along the fuselage. The tail generates almost no side force due to its symmetry, but the small amount of side force
generated by the tail is out of phase or opposite to that generated by the canards. This
variation in the tail force indicates the vortices generated by the canards travel the length
of the body and are still strong enough to modify the pressure distribution on the tail.
The tail force is out of phase with the canards since the canards are in their maximum or
minimum deflection for approximately 20 degrees, but it takes nearly 10 degrees of roll
for a particle to travel from the canards to the tail.

All of the components have significant normal or lift force contributions. As in
the case of the side force, the tail component is out of phase with the canard normal force.
Near the maximum canard lift orientations, \( \theta = 0 \) degrees and 180 degrees, the fuselage
and tail are also generating their maximum normal force contributions, indicating the
vortices shed from the canards are modifying the pressure distribution on the fuselage
and tail to increase the normal force. The canards also generate the majority of the
pitching moment with some contribution from the tail.

Comparisons of the canard force and moment coefficients are shown in Figures
5.20 – 5.25 for all three cases. The canard was selected because it lies completely within
the sliding interface surface and the incoming flow must pass through the interface before
acting on the canard surface. Any influence of the sliding interface should be evident in
the forces generated by the canard. As seen in Figures 5.20 – 5.25, there is virtually no
difference between the three solutions, indicating little to no influence by the sliding
interface.
Figure 5.20 Canard axial force coefficient for the 0% canard command level

Figure 5.21 Canard side force coefficient for the 0% canard command level
Figure 5.22 Canard normal force coefficient for the 0% canard command level

Figure 5.23 Canard rolling moment coefficient for the 0% canard command level
Figure 5.24 Canard pitching moment coefficient for the 0% canard command level

Figure 5.25 Canard yawing moment coefficient for the 0% canard command level
Helicity contours are used to detect and track the vortices generated by the canards. Helicity is a measure of the helical motion of the fluid and is defined as

\[ H = \vec{V} \cdot (\vec{V} \times \vec{V}) = \vec{V} \cdot \vec{\omega} \]

Due to the magnitude of \( H \), \( \pm \log_{10}(|H|) \) is actually plotted and \( \pm \) is used depending upon the original sign of \( H \). Note that values of \( H \) in the range \(-1 \leq H \leq 1\) were set to zero. The planes on which helicity is plotted are located at \( x/L = 0.15, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, \) and 0.9. For the helicity contours, the red color indicates a positive rotation of the flow (clockwise if viewed from the nose looking aft) and the blue color indicates a negative (counter-clockwise) rotation. Figure 5.26 shows the path of the vortices and position relative to the missile at several different roll angles and also how they change in rotation direction due to the canard motion. The helicity contours shown in Figure 5.26 are for the UVI solution, as there are no discernable differences in the helicity contours among the three solutions. A comparison of the helicity contours for each solution is shown in Figure 5.27. The contours are nearly identical and thus the vortices pass through the sliding interface without any additional dissipation or distortion. A detailed view of the helicity contours taken just behind the canard is shown in Figure 5.28. The location of the first plane at \( x/L = 0.15 \) passes through the sliding interface and the location of the other planes is \( x/L = 0.175, 0.2, 0.225, \) and 0.25.
Figure 5.26  Helicity contours for the 0% command level at a series of roll angles: (a) 0°, (b) 45°, (c) 90°, (d) 135°, (e) 180°, (f) 225°, (h) 270°, and (i) 315°
Figure 5.27  Comparison of the helicity contours along the length of the missile
Figure 5.28  Comparison of the helicity contours immediately behind the canards
Included in Figures 5.12 – 5.17 and Table 5.2 are the results using the OVERFLOW-D flow solver from reference 87. While both codes are high-resolution Navier-Stokes flow solvers, OVERFLOW-D represents a different approach to solving this type of complex problem involving relative grid motion. Whereas U²NCLE uses an unstructured grid methodology, OVERFLOW-D uses a Chimera overset structured grid methodology. The OVERFLOW-D results were generated using an extremely fine spatial and temporal resolution: 41 million grid points and 12,000 time steps per revolution. The roll-averaged results using the sliding interface methodology agree very well. Qualitatively, the force and moment coefficient roll histories compare well with the most noticeable difference occurring at roll-angles of 120 and 210 degrees. The differences are likely due to the increase in resolution of the OVERFLOW-D grid, 41 million vs. 9 million points and increase in temporal resolution. However, it is difficult to draw any definitive conclusion other than both methodologies predict nearly similar results and the lack of significant differences indicate both codes offer viable approaches to solving this class of problem.

In an attempt to verify the accuracy of the results predicted by the sliding interface method, a comparison was made with experimental data [90]. A simulation was performed representative of one of the experimental tests, which included the ability to simultaneously roll the missile and to dither the canards. The actual motion of the canards during the experiment was averaged over ten body revolutions and is represented in Figure 5.29 by the curve labeled ‘Exp-Avg’ [85]. An analytical approximation to the ensemble average was used to dither the canards in the simulation and is represented by the curve labeled ‘Exp-Fit’ in Figure 5.29. Note that this command level is also periodic.
over one body revolution. The run conditions were $M_\infty = 1.6$, $\alpha = 3.0$ deg., and $\Omega = 18.0$ Hz.

![Figure 5.29 Experimental canard command level](image)

The computed force and moment coefficients for all three cases are shown in Figures C.7 – C.12. Both of the sliding interface solutions again yield nearly the same solution as the UVI method, and the curve for the Sliding1 solution lies directly atop the curve for the UVI solution. As before, the influence of the canard motion is quite evident. This command level produces a starboard yaw movement of the missile whereas the 0% command level resulted in zero or very small side forces. The roll-averaged forces are compared against the experimental values in Table 5.3. Note the experimental axial force coefficient includes the contribution from the base region but not the
correction for buoyancy, which is an experimentally derived quantity accounting for the non-zero pressure gradient in the streamwise direction of the wind tunnel [90]. Except for the side force coefficient, \(C_y\), all of the results are within the bounds of the experimental uncertainty. Over the ten body revolutions, the experimental dither pattern has significant variation (approximately 7 degrees) in the canard deflection angle around \(\theta = 160\) degrees to 180 degrees and deviates significantly from the curve fit. This could potentially affect the side force coefficient, but further investigation is required. Overall, the results compare very well.

### Table 5.3 Roll-averaged force and moment coefficients for the Exp-fit command level

<table>
<thead>
<tr>
<th>Solution</th>
<th>(C_A)</th>
<th>(C_y)</th>
<th>(C_N)</th>
<th>(C_l)</th>
<th>(C_m)</th>
<th>(C_n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Experiment</td>
<td>1.155 – 1.176</td>
<td>0.15 – 0.20</td>
<td>0.45 – 0.61</td>
<td>-0.036 – -0.019</td>
<td>-1.5 – -0.39</td>
<td>0.93 – 1.5</td>
</tr>
<tr>
<td>UVI</td>
<td>1.155</td>
<td>0.22</td>
<td>0.51</td>
<td>-0.023</td>
<td>-0.76</td>
<td>1.5</td>
</tr>
<tr>
<td>Sliding1</td>
<td>1.155</td>
<td>0.22</td>
<td>0.51</td>
<td>-0.024</td>
<td>-0.76</td>
<td>1.5</td>
</tr>
<tr>
<td>Sliding2</td>
<td>1.153</td>
<td>0.21</td>
<td>0.51</td>
<td>-0.025</td>
<td>-0.70</td>
<td>1.5</td>
</tr>
</tbody>
</table>

5.1.4 Computational Expense

All simulations were performed on a Linux supercluster located at the ERC at Mississippi State University. The cluster is comprised of 192 IBM x335 nodes and uses 100Mb/s Ethernet switches for inter-node communication. Individual nodes contain dual 3.06GHz Xeon processors and 2.5GB of RAM. Each solution utilized 64 processors and run time information is listed in Table 5.4. All times are listed in seconds. Even though the Sliding2 grid is slightly larger than the UVI grid, it had a shorter run time than the UVI grid. In each case, a single revolution can be run in less than 10 hours.
Table 5.4 Computational expense for the spinning missile simulations

<table>
<thead>
<tr>
<th>Grid</th>
<th>Pts. Per Processor</th>
<th>CPU time per time step</th>
<th>Wall time per time step</th>
<th>Wall time per revolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>UVI</td>
<td>145,000</td>
<td>63.78</td>
<td>97.46</td>
<td>35,087</td>
</tr>
<tr>
<td>Sliding1</td>
<td>145,000</td>
<td>59.89</td>
<td>92.02</td>
<td>33,126</td>
</tr>
<tr>
<td>Sliding2</td>
<td>150,000</td>
<td>63.26</td>
<td>95.98</td>
<td>34,553</td>
</tr>
</tbody>
</table>

There are three main tasks in the sliding interface method: constructing the interface, searching for the host elements of the extruded sliding nodes, and updating of the various quantities for the sliding nodes. The percentage of total run time for each task is listed in Table 5.5. The total run time is the sum of the run times for the individual processors. The computational expense of the sliding interface is approximately 25-30% of the overall expense, with the majority of the expense due to communication overhead to update the values at the sliding nodes. Recall that when a particular quantity at a sliding nodes is updated, communication takes place between the processor on which the sliding node resides and the one containing the host element.

Table 5.5 Percentage of the total run time for the individual sliding interface tasks for the spinning missile simulations

<table>
<thead>
<tr>
<th>Grid</th>
<th>Build Interface</th>
<th>Search</th>
<th>Update</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sliding1</td>
<td>3.94</td>
<td>1.53</td>
<td>23.27</td>
<td>28.74</td>
</tr>
<tr>
<td>Sliding2</td>
<td>4.45</td>
<td>2.33</td>
<td>20.58</td>
<td>27.36</td>
</tr>
</tbody>
</table>

Even though the sliding interface adds an additional 25-30% to the cost of the simulation, the sliding interface solutions still have shorter run times. Recall that the reconnection process is an inherently serial process and takes place on a single processor. In this case, it takes place on two processors, one for the port canard and one for the
starboard. Thus the UVI method has a longer run time because the majority of the processors, 62 of 64, are idle during the reconnection process. To get an idea of the percentage of time the remaining processors are idle, the following estimate is used [91]:

\[
\% \text{Idle} = \frac{T_{\text{reconn}} (np - nuvi)}{T_{\text{Total}}} \cdot 100
\]

(5.3)

where \( T_{\text{reconn}} \) is the total time spent in the rotation and reconnection process for all UVI processors, \( np \) is the total number of processors, \( nuvi \) is the number of UVI processors, and \( T_{\text{Total}} \) is the sum of the run times for the individual processors. Note the for the sliding interface cases, \( nuvi \) is the number of processors or partitions having sliding interface surfaces. The results are listed in Table 5.6 and all times are listed in seconds. Note that nearly 20\% of the run time is spent idle for all but two of the processors in the UVI solution.

<table>
<thead>
<tr>
<th>Grid</th>
<th>Total Run Time</th>
<th>Total Communication Time</th>
<th>Total Rotation/Reconnection Time</th>
<th>nuvi</th>
<th>POV</th>
<th>% Idle</th>
</tr>
</thead>
<tbody>
<tr>
<td>UVI</td>
<td>2,252,844</td>
<td>512,836</td>
<td>6,569.7</td>
<td>2</td>
<td>22.76</td>
<td>18.08</td>
</tr>
<tr>
<td>Sliding1</td>
<td>2,127,404</td>
<td>69,860</td>
<td>1,028.4</td>
<td>8</td>
<td>3.28</td>
<td>2.71</td>
</tr>
<tr>
<td>Sliding2</td>
<td>2,218,673</td>
<td>72,806</td>
<td>1,078.9</td>
<td>11</td>
<td>3.28</td>
<td>2.58</td>
</tr>
</tbody>
</table>

Also listed in Table 5.6 are the total communication time and the parallel overhead (POV). The total communication time is the sum of the communication times for the individual processors. The POV is the ratio of the total communication time to the total run time,
POV = \frac{Total\ Communication\ Time}{Total\ Run\ Time}  \tag{5.4}

and can also be thought of as an efficiency measure. The UVI solution has nearly an order of magnitude more communication than the sliding interface solutions while having a nearly equal run time and thus has significantly more overhead costs. The additional communication is a result of the less than optimum grid partitioning. All of the vertices and edges attached to the UVI surfaces must be kept on the same processor for the reconnection process and results in less than ideal partitioning for equal load balancing. Thus a second penalty is incurred due to the serial reconnection process.

5.2 Case 2: Missile with Free-Spinning Tail

The missile configuration used for the second case is shown in Figure 5.30. The missile has a tangent ogive nose, cylindrical fuselage, four moveable canards, and four tail fins. The tail fins are attached to a bearing that spins relative to the fuselage section fore and aft of it. The fuselage has a fineness ratio of approximately 15. Yaw and pitch control is achieved using the canards. The canards are equally spaced around the fuselage in an X-configuration relative to the plus (+)-configuration of the tail fins. For the results presented here, the canards are deflected 16 degrees and held fixed in that orientation as shown in Figure 5.30. The four tail fins are located at 90-degree increments circumferentially around the tail section and are shown in Figure 5.30 in a zero degree orientation.

The missile geometry also includes various geometrically complex features along the missile fuselage. Included are a ring of 10 protuberances located at approximately
mid-span of the fuselage, two covers located aft of the canards that extend axially approximately 20% of the fuselage, and two nozzles located aft of the tail fins. In addition, there are two railings (located at 0° and 180°) that run nearly the length of the fuselage.

The original motivation behind this simulation was to predict the aerodynamic performance of a missile with a free-spinning tail. The tail fins are mounted on a bearing that allows the tails fins to rotate as a unit relative to the fuselage about the missile
longitudinal axis. When the canards are deflected or the air flow disturbed, the resulting aerodynamic loading imparts a rolling moment on the tail fins causing them to spin, even under steady-state flight conditions. To accurately compute the aerodynamic performance of the missile requires that the tail dynamics be adequately resolved. The spin-rate of the tail is a function of the canard deflection and the freestream angle of attack. In addition, at low angles of attack, the spin rate is also dependent upon the strength and position of the vortices generated by the canards that convect downstream and impinge on the tail fins. It was of particular interest to determine the tail spin rate that results in a net zero (roll-averaged) rolling moment. Previous results describing the required grid resolution for accurate prediction of the aerodynamic forces acting on the body and the relative importance of the viscous effects and tail spin rate are given in references [92]. Those results were generated using the UVI method and were found to be in good agreement with other grid topology approaches that have been successfully applied to this same problem: Murman et al. [93] applied a Cartesian method and both Hall [94] and Nygaard et al. [95] used Chimera overset structured grid methods. For the purpose of this work, the results generated using the UVI methods will serve to validate those using the new sliding interface method and comparison of these results will be made to the other topology approaches when available. The effect of the sliding interface will be evident since the canard vortices, which have a significant impact on the aerodynamic performance of the missile, must pass through the sliding interface in order to impinge on the tail.
5.2.1 Grid Construction

Body-fitted mixed-element type unstructured grids suitable for viscous simulations were generated using AFLR. The surface grid for the fuselage, canards, and tail are shown in Figures 5.31, 5.32, and 5.33, respectively. Both the UVI and sliding interface volume grids were constructed using essentially the same surface grids. An initial normal spacing of $5.0 \times 10^{-5}$ was used for all the volume grids, which led to $y^+$ values of less than 1.0 for the first point from the body indicating good viscous sublayer resolution. An example of the field grid and boundary-layer resolution is shown in Figure 5.34.

![Figure 5.31 Fuselage surface grid for the missile with a free-spinning tail](image)
Figure 5.32  Canard surface grid for the missile with a free-spinning tail
Figure 5.33  Tail surface grid for the missile with a free-spinning tail
In order to accurately resolve the tail aerodynamics, an emphasis was placed on capturing the vortices generated by the canards and resolving them along the missile body. If the vortices remain close to the missile body, they can modify the pressure distribution along the length of the missile. This in turn changes the forces and moments acting on the missile and affects the aerodynamic performance. Thus it is very important to adequately resolve the vortices. Similar to the spinning missile grid, an embedded surface was constructed around the missile to better control the point spacing and the growth of the cell sizes near the canard tips and along the body. This allowed fine
resolution for the field grid between this surface and the missile, and allowed a coarse field grid between this surface and the farfield surface. The embedded surface surrounding the missile was approximately three diameters (of the missile body) in width in order to encapsulate the vortices generated by the canards.

Significant modifications to the geometry were required for the UVI method. Figure 5.35 shows a detailed section near the tail. However for the UVI surface to encompass the tail, the fuselage requires two cuts for the UVI surface to pass through: one immediately before the bearing and one immediately after. As shown in Figure 5.35(a), both the railings extend underneath the tail fins and there is a very small gap between the fins and the railings to allow the fins to pass over the railings as they spin. To adequately resolve this small gap requires fine point spacings. However due to the memory constraints of a single processor, the required resolution exceeds that allowed for the number of points on the UVI surface. Recall that the UVI method is based on an inherently sequential grid generation reconnection process, and as such, the entire UVI surface and all volume elements connecting to the surface must reside on a single processor. Therefore both railings were truncated to avoid resolving the gaps and the resulting UVI surface and modified railings are shown in Figure 5.36. The subsequent volume grid is shown in Figure 5.37. Note that part of the stationary fuselage is now within the UVI surface and will be rotated. To counteract the effect of this spinning surface, a rotating viscous boundary condition was applied to this surface with a spin rate equal and opposite to that of the tail. As was done for the spinning missile case, regions were defined to turn off the portions of the field grid inside the UVI-created gaps of the fuselage shown in Figure 5.37.
Figure 5.35. Missile geometry near the tail illustrating (a) the original geometry and the (b) geometric modification to the railings and fuselage for the UVI method.
Figure 5.36  UVI surface grid for the free-spinning tail missile

Figure 5.37  Modification to the fuselage and railing and the resulting UVI volume grid
Again, two grids for use with the sliding interface method were constructed. In order to make a direct comparison with the UVI method, the first sliding interface grid is the same as the UVI grid except that the nodes on the interface surface surrounding the tail are not merged. The same cylindrical blockages were defined to represent the missing portions of the fuselage. The second sliding interface grid is constructed with no modifications to the fuselage or railing and is shown in Figure 5.38. A detailed view, as indicated in Figure 5.38, of the volume grid in the region between the railing and the tail is shown in Figure 5.39. The sliding interface surface and location where it intersects the fuselage in this region is shown in Figure 5.40. The same volume grid generation parameters were used as for the UVI grid so that the field grids would be as similar as possible. However, extending the railings and resolving the gap between the fins and the railings adds approximately two million nodes to the Sliding2 grid. Most of the additional grid points are in the isotropic region, as evident by the additional number of tetrahedral and similar number of boundary elements listed in Table 5.7, which lists the number of nodes and elements for each grid. Again, note the difference in the number of nodes and triangles between the UVI and Sliding1 grids is due to duplication on the interface surfaces.

Table 5.7 Volume grid sizes for the free-spinning tail missile

<table>
<thead>
<tr>
<th>Grid</th>
<th>Nodes</th>
<th>Triangles</th>
<th>Quads</th>
<th>Tetrahedra</th>
<th>Pyramids</th>
<th>Prisms</th>
</tr>
</thead>
<tbody>
<tr>
<td>UVI</td>
<td>9,471,032</td>
<td>970,200</td>
<td>0</td>
<td>25,333,748</td>
<td>47,174</td>
<td>10,004,190</td>
</tr>
<tr>
<td>Sliding1</td>
<td>9,504,695</td>
<td>1,037,522</td>
<td>0</td>
<td>25,333,748</td>
<td>47,174</td>
<td>10,004,190</td>
</tr>
<tr>
<td>Sliding2</td>
<td>12,036,746</td>
<td>1,820,388</td>
<td>19,658</td>
<td>36,618,343</td>
<td>45,714</td>
<td>10,858,416</td>
</tr>
</tbody>
</table>
Figure 5.38  Sliding interface surface for the free-spinning tail missile
Figure 5.39 Detailed view of the volume grid between the railing and the tail fin
5.2.2 Simulation Conditions and Parameters

The simulations were performed using the following parameters: $M_\infty = 1.6$, $\alpha = 4$ degrees, $\omega = 2500$ rpm, and $Re_L = 7.01 \times 10^6$ based on the missile length. This angle of attack was selected since it was expected to have the strongest interaction between the canard vortices and the tail. The force and moment coefficients are defined using the standard normalization by dynamic pressure. The reference area $S$ was taken to be the
cross-sectional area of the fuselage and the reference length \( c \) was taken as the fuselage diameter. The moments were taken about a point at \( x/L = 0.66 \) located on the centerline.

Preliminary simulations were performed to determine the proper solver parameters to ensure the unsteady solution was converged at each time step. The convergence study results are presented in reference 96. For the results presented here, eight linear sub-iterations were used. Time step studies were conducted and a \( \Delta t \) corresponding to 0.5 degrees of rotation per time step was deemed necessary. Convergence studies were also performed to determine the number of Newton iterations needed per time step, and it was found that three Newton iterations were sufficient.

5.2.3 Simulation Results

The flow field about this missile is also characterized by complex vortical flow and shock structures. The cross section of the canards are thin supersonic airfoils with sharp edges that generate shocks and expansions. The canards also generate strong horseshoe vortices that convect the length of the missile. At this low angle of attack, the vortices are also close enough to the missile body to effect the force distribution on the fuselage and tail. A detached bow shock is formed just ahead of the tangent-ogive nose and intersecting shocks and expansion are formed around the canards and tail fins.

5.2.3.1 Static Results

Before the unsteady cases were run, a static case (\( \omega = 0 \) rpm) was run to determine the direction the tail spins. For the static case, the tail is in an X-configuration
where the tail fins have been rotated 45 degrees (clockwise if viewed looking aft from the nose) from the zero degree orientation shown in Figure 5.30. This orientation was selected because the X-configuration is the statically stable configuration of the tail. The static case also provides a qualitative indication of the flowfield. Since the flow is supersonic, disturbances outside the boundary-layer cannot propagate upstream and thus the majority of the flowfield will be identical, regardless of the tail spin-rate. The surface pressure contours shown in Figure 5.41 illustrate the shocks and expansions due to the canards and the various protuberances along the length of the body. All of the static results in Figures 5.41 - 5.43 are shown using the Sliding2 solution. Again, upstream of the tail region, the flow fields are nearly identical for all three solutions.
Figure 5.41  Static surface pressure contours

(a) Windward side

(b) Leeward side
With the asymmetrical canard deflection used here, the two canards on the upper surface generate tip vortices rotating in a negative sense about the longitudinal axis (or counter-clockwise if viewed from the nose looking aft). The corresponding canard root vortices are rotating in a positive sense (clockwise) about the longitudinal axis. The lower two canards generate clockwise tip vortices and counterclockwise root vortices.

Helicity is again used to track the vortices generated by the canards and $\pm \log_{10} |H|$ is plotted on planes located at $x/L = 0.25, 0.3, 0.4, 0.49, 0.6, 0.7, 0.8, 0.89, 0.96$, and $1.05$. The strongest vortex is generated by the upper port canard since it is deflected leading edge upwards and sees the largest relative angle of attack. Due to the angle of attack and deflection of the canards, as the vortices travel downstream they convect upwards and to the port side as indicated in Figure 5.42(a). By the tail section of the missile, 3 of the 4 pairs of vortices are located in the upper port quadrant. The fourth pair, the one generated by the lower starboard canard, impinges on the fuselage as shown in Figure 5.42(b). Based on Figure 5.42, the root vortices from the upper and lower port canards appear to be wrapped around the stronger tip vortices as they convect downstream.

This vortex system leads to an asymmetrical inflow condition for the tail fins. The asymmetry of the flow results in an asymmetrical pressure distribution on the tail fins. The pressure differential on the upper port tail fin is evident in Figure 5.43. The canards, which cause a starboard motion of the missile, combined with this vortex induced pressure difference on the fins produce a positive rolling moment about the longitudinal axis and causes the tail to spin. The roll-averaged static results for the UVI and sliding interface solutions are shown in Table 5.8. Again, there is very little
difference between the UVI and Sliding1 results. However, there is an appreciable
difference between the Sliding1 and Sliding2 results. The difference is due to the
difference in geometry between the two configurations, namely the additional segment of
railing in the Sliding2 grid.

Also included in Table 5.8 are the results generated using the OVERFLOW-D flow solver [95]. The axial force coefficient for the OVERFLOW-D results is omitted because it did not include the base drag. Except for the pitching moment coefficient, there is excellent agreement with the results predicted using OVERFLOW-D.

<table>
<thead>
<tr>
<th>Solution</th>
<th>(C_A)</th>
<th>(C_Y)</th>
<th>(C_N)</th>
<th>(C_l)</th>
<th>(C_m)</th>
<th>(C_n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>UVI</td>
<td>0.787</td>
<td>0.607</td>
<td>1.083</td>
<td>0.453</td>
<td>0.297</td>
<td>-6.317</td>
</tr>
<tr>
<td>Sliding1</td>
<td>0.788</td>
<td>0.607</td>
<td>1.085</td>
<td>0.454</td>
<td>0.289</td>
<td>-6.319</td>
</tr>
<tr>
<td>Sliding2</td>
<td>0.808</td>
<td>0.602</td>
<td>1.097</td>
<td>0.455</td>
<td>0.246</td>
<td>-6.344</td>
</tr>
<tr>
<td>OVERFLOW</td>
<td>0.607</td>
<td>1.06</td>
<td>0.483</td>
<td>0.439</td>
<td>0.439</td>
<td>-6.44</td>
</tr>
</tbody>
</table>
Figure 5.42  Static helicity contours
Figure 5.43  Static pressure contours on the surface of the tail
5.2.3.2 Spinning Tail

In the present work, tail spin rates of $\omega = 2500$, and 5000 rpm were examined. The direction to spin the tail was determined from the zero spin-rate case, i.e. the tail is held fixed. The previously described vortex system leads to an asymmetrical inflow condition for the tail fins. The asymmetry of the flow results in an asymmetrical pressure distribution on the tail fins. The pressure differential on the upper port tail fin is evident in Figure 5.43. The canards, which cause a starboard motion of the missile, combined with this vortex induced pressure difference on the fins produce a positive rolling moment about the longitudinal axis and cause the tail to spin.

It was of interest to find the natural tail spin rate, or the spin rate that generates zero rolling moment. An iterative procedure was used to estimate the natural spin rate. Simulations were performed using constant tail spin rates of $\omega = 2500$ rpm and 5000 rpm. The force and moment coefficient histories for the $\omega = 2500$ rpm case are shown in Figures 5.44 – 5.49 and the roll-averaged results are shown in Table 5.9. The force and moment coefficient histories for the $\omega = 5000$ rpm case are included in Appendix D and the roll-averaged results are listed in Table 5.10. Note the solution is periodic after a quarter revolution.

| Table 5.9 Roll-averaged force and moment coefficients for $\omega = 2500$ rpm |
|-----------------|--------|--------|--------|--------|--------|--------|
| Solution        | $C_A$  | $C_Y$  | $C_N$  | $C_I$  | $C_m$  | $C_n$  |
| UVI             | 0.794  | 0.585  | 1.152  | -0.119 | -2.00e-2 | -6.415 |
| Sliding1        | 0.795  | 0.585  | 1.153  | -0.119 | -2.52e-2 | -6.418 |
| Sliding2        | 0.811  | 0.580  | 1.169  | -0.119 | -8.40e-2 | -6.446 |
Table 5.10 Roll-averaged force and moment coefficients for $\omega = 5000$ rpm

<table>
<thead>
<tr>
<th>Solution</th>
<th>$C_A$</th>
<th>$C_Y$</th>
<th>$C_N$</th>
<th>$C_I$</th>
<th>$C_m$</th>
<th>$C_n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>UVI</td>
<td>0.802</td>
<td>0.568</td>
<td>1.153</td>
<td>-0.640</td>
<td>-2.72e-2</td>
<td>-6.497</td>
</tr>
<tr>
<td>Sliding1</td>
<td>0.804</td>
<td>0.568</td>
<td>1.155</td>
<td>-0.641</td>
<td>-3.33e-2</td>
<td>-6.499</td>
</tr>
<tr>
<td>Sliding2</td>
<td>0.820</td>
<td>0.562</td>
<td>1.169</td>
<td>-0.646</td>
<td>-8.57e-2</td>
<td>-6.531</td>
</tr>
</tbody>
</table>

Upon inspection of the force and moment plots and the roll-averaged values, the Sliding1 and UVI results are again very similar. The largest variation occurs in the axial force coefficient and the other values are virtually identical, most notable the rolling moment coefficient. The differences in the Sliding2 results are due to the presence of the railing as the tail fin passes overhead. A comparison of the crossflow velocity vectors taken on a cutting plane in the vicinity of where the tail passes over the top railing is shown in Figures 5.50 – 5.52 for the $\omega = 2500$ rpm tail spin rate solution. The location of the cutting plane is shown in Figure 5.53 and the view in Figures 5.50 – 5.52 is standing at the nose and looking aft. Note the railing has been truncated and is no longer present in the UVI and Sliding1 solutions. The presence of the railing causes the flow in the gap between the railing and the fin to accelerate to nearly 45% of the freestream value. The flow in this region is nearly twice as fast as compared to the flow in the same region (without the railing) in the UVI and Sliding1 solutions. Also note the UVI and Sliding1 solutions are nearly identical.

Figure 5.54 shows a comparison of the convergence history for each grid. Both the UVI and sliding interface methods converge the residual to approximately the same order of magnitude. Thus there does not appear to be any adverse effect due to the non-conservativeness of the flux evaluation in the sliding interface method.
Figure 5.44  Axial force coefficient for the $\omega = 2500$ rpm tail spin rate

Figure 5.45  Side force coefficient for the $\omega = 2500$ rpm tail spin rate
Figure 5.46  Normal force coefficient for the $\omega = 2500$ rpm tail spin rate

Figure 5.47  Rolling moment coefficient for the $\omega = 2500$ rpm tail spin rate
Figure 5.48 Pitching moment coefficient for the $\omega = 2500$ rpm tail spin rate

Figure 5.49 Yawing moment coefficient for the $\omega = 2500$ rpm tail spin rate
Figure 5.50 Crossflow velocity vectors near the tail for the UVI solution at the $\omega = 2500$ rpm tail spin rate
Figure 5.51  Crossflow velocity vectors near the tail for the Sliding1 solution at the $\omega = 2500$ rpm tail spin rate
Figure 5.52 Crossflow velocity vectors near the tail for the Sliding2 solution at the $\omega = 2500$ rpm tail spin rate
Figure 5.53 Cutting plane location for the crossflow velocity vector plots
The main effect of the tail spin rate is on the rolling moment coefficient. As shown in Tables 5.8 and 5.9, the value of the roll-averaged rolling moment changes sign from the $\omega = 0$ rpm to $\omega = 2500$ rpm case and becomes larger (i.e. more negative) as the spin rate is increased to 5000 rpm. A plot of the roll-averaged rolling moment coefficients vs. tail spin rate for each solution is shown in Figure 5.55. For this range of spin rates, the relationship between spin rate and the rolling moment coefficient is linear. Using linear interpolation, the spin rate to generate a zero net rolling moment was estimated to be 1980 rpm using the UVI and Sliding1 results and 1982 rpm based on the Sliding2 data. Both of these values compare to the zero net rolling moment estimated by the OVERFLOW-D code, 1970 rpm [95]. Simulations were run at these estimated zero
moment spin rates and the roll-averaged results are included in Table 5.11. The rolling moment coefficient was indeed nearly zero for each case.

![Graph of rolling moment coefficient versus spin rate](image)

**Figure 5.55** Rolling moment coefficient versus spin rate for each methodology

<table>
<thead>
<tr>
<th>Solution</th>
<th>$C_A$</th>
<th>$C_Y$</th>
<th>$C_N$</th>
<th>$C_1$</th>
<th>$C_m$</th>
<th>$C_n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>UVI</td>
<td>0.793</td>
<td>0.589</td>
<td>1.152</td>
<td>-1.16e-2</td>
<td>-2.17e-2</td>
<td>-6.399</td>
</tr>
<tr>
<td>Sliding1</td>
<td>0.795</td>
<td>0.588</td>
<td>1.154</td>
<td>-1.14e-2</td>
<td>-2.71e-2</td>
<td>-6.401</td>
</tr>
<tr>
<td>Sliding2</td>
<td>0.810</td>
<td>0.583</td>
<td>1.170</td>
<td>-1.08e-2</td>
<td>-8.81e-2</td>
<td>-6.429</td>
</tr>
</tbody>
</table>

Table 5.11 Roll-averaged force and moment coefficients for the estimated zero spin rate condition ($\omega \approx 1980$ rpm)

The force and moment coefficient plots for the $\omega = 5000$ rpm and $\omega = 1980$ rpm cases are included in Appendix D. Not surprisingly, the tail spin rate had very little effect on the other force and moment coefficients. The magnitude of the side force and yawing
moment coefficients decreased slightly as the spin rate increased as shown in Figures D.8 and D.12. The normal force and pitching moment coefficients are virtually unaffected by the tail spin rate.

As discussed previously, the vortex interaction has a significant effect on the missile performance. The fact that the sliding interface results are similar to the UVI results is important since the vortices must pass through the sliding interface to interact with the tail. Thus, it appears the vortices are passing through the sliding interface without any change in strength or position and the interface is not having any noticeable impact on the results. Comparisons of the tail force and moment coefficients are shown in Figures 5.56 – 5.61 to check for further evidence of any influence of the sliding interface on the forces generated by the tail. The tail was selected because it lies completely within the sliding interface surface, and the vortices must pass through the interface before interacting with the tail surfaces. As discussed before, there is very little difference in the Sliding1 and UVI solutions indicating little to no influence by the sliding interface. The differences in the Sliding2 results are due to geometric differences in the grids.

The force and moment histories for the Sliding2 solution decomposed by component are included in Appendix D. As expected, the fuselage was responsible for the majority of the axial force. Note that the axial force acting on the base region is acting in the opposite direction of the other forces. The forces acting on this surface are primarily pressure forces, which is due to the pressure drag caused by the expansion at the end of the missile. Due to the orientation and symmetry of this surface, the forces
acting on this surface are only in the axial direction and have no contribution to the side or normal forces or any of the moments.

As indicated in Figure D.2, the canards generate a side force to induce a starboard motion of the vehicle while the tail generates an opposing side force. All of the components have significant normal force contributions and the variation in the normal force over a quarter revolution is due strictly to the tail. As previously mentioned, the tail is responsible for nearly all of the rolling moment as illustrated in Figure D.4. All of the components have significant contributions to the pitching and yawing moments.
Figure 5.56  Tail axial force coefficient for the $\omega = 2500$ rpm tail spin rate

Figure 5.57  Tail side force coefficient for the $\omega = 2500$ rpm tail spin rate
Figure 5.58  Tail normal force coefficient for the $\omega = 2500$ rpm tail spin rate

Figure 5.59  Tail rolling moment coefficient for the $\omega = 2500$ rpm tail spin rate
Figure 5.60  Tail pitching moment coefficient for the $\omega = 2500$ rpm tail spin rate

Figure 5.61  Tail yawing moment coefficient for the $\omega = 2500$ rpm tail spin rate
The variation of rolling moment is caused by the vortices impinging on the tail fins as they rotate. The decrease in the rolling moment coefficient near $\theta = 35^\circ$ occurs as the fin approaches the strongest counter-clockwise rotating vortex, the one generated by the upper port canard. Helicity contours for the UVI solution are shown at multiple rotation angles in Figure 5.62 as the vortices impinge on the tail. The increase in rolling moment coefficient occurs as the tail fin encounters the clockwise rotating vortex generated by the lower port canard. A similar variation is also evident in the side force coefficient as indicated in Figure 5.57.

A comparison of the helicity contours near the tail for each solution is shown in Figures 5.63 and 5.64. The planes on which helicity is plotted are $x/L = 0.96$ and 1.05. The contours are very similar and thus the vortices pass through the sliding interface without any additional dissipation or distortion.
Figure 5.62 Helicity contours near the tail for the $\omega = 2500$ rpm tail spin rate
Figure 5.63 Comparison of the helicity contours near the tail
Figure 5.63  Comparison of the helicity contours near the tail (cont.)
5.2.4 Computational Expense

Each solution utilized 64 processors on the same cluster as the spinning missile case. The run time information is listed in Table 5.12 and all times are listed in seconds. Note the Sliding2 grid is significantly larger than the other two grids yet had a shorter run time than did the UVI case. In each case, a quarter revolution can be run in less than 9 hours and the Sliding1 case in under 5 hours. The percentage of total run time for each of the main sliding interface tasks is listed in Table 5.13. Similar to the spinning missile case, the computational expense of the sliding interface is approximately 30% of the overall expense. The parallel efficiency measures are listed in Table 5.14. Compared to the spinning missile case, there are more points on the UVI surface (67,000 points vs 13,000) and thus the reconnection takes significantly longer. With just a single UVI processor, the UVI solution incurs significantly more idle time and a correspondingly longer run time.

Table 5.12 Computational expense for the free-spinning tail missile simulations

<table>
<thead>
<tr>
<th>Grid</th>
<th>Pts. Per Processor</th>
<th>CPU time per time step</th>
<th>Wall time per time step</th>
<th>Wall time per ¼ revolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>UVI</td>
<td>148,000</td>
<td>59.32</td>
<td>175.16</td>
<td>31,528</td>
</tr>
<tr>
<td>Sliding1</td>
<td>148,500</td>
<td>56.31</td>
<td>96.13</td>
<td>17,304</td>
</tr>
<tr>
<td>Sliding2</td>
<td>188,074</td>
<td>82.78</td>
<td>164.30</td>
<td>29,574</td>
</tr>
</tbody>
</table>

Table 5.13 Percentage of the total run time for the individual sliding interface tasks for the free-spinning tail missile simulations

<table>
<thead>
<tr>
<th>Grid</th>
<th>Build Interface</th>
<th>Search</th>
<th>Update</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sliding1</td>
<td>4.45</td>
<td>1.61</td>
<td>21.77</td>
<td>27.83</td>
</tr>
<tr>
<td>Sliding2</td>
<td>5.19</td>
<td>2.72</td>
<td>24.00</td>
<td>31.91</td>
</tr>
</tbody>
</table>
Table 5.14 Parallel efficiency measures for the free-spinning tail missile simulations

<table>
<thead>
<tr>
<th>Grid</th>
<th>Total Run Time</th>
<th>Total Communication Time</th>
<th>Total Rotation/Reconnection Time</th>
<th>nuvi</th>
<th>POV</th>
<th>% Idle</th>
</tr>
</thead>
<tbody>
<tr>
<td>UVI</td>
<td>2,024,804</td>
<td>547,898</td>
<td>9,032</td>
<td>1</td>
<td>27.06</td>
<td>28.10</td>
</tr>
<tr>
<td>Sliding1</td>
<td>1,114,747</td>
<td>38,319</td>
<td>116</td>
<td>22</td>
<td>3.44</td>
<td>0.44</td>
</tr>
<tr>
<td>Sliding2</td>
<td>1,902,372</td>
<td>46,039</td>
<td>177</td>
<td>32</td>
<td>2.42</td>
<td>0.30</td>
</tr>
</tbody>
</table>

5.3 Case 3: High-Speed Centrifugal Compressor

Sheng [97] has applied the present sliding interface implementation to a simulation of a high-speed centrifugal compressor with tip injection. This work was sponsored by the Army research Laboratory (ARL) under the supervision of Dr. Michael Hathaway and the experimental data was provided by Mr. Gary Skoch. The purpose of the tip injection is to extend the stable flow range of the compressor. Air is injected through the shroud surface or control tubes into the vaneless region of a vane-island diffuser. The configuration is shown in Figure 5.65, with the shroud removed for clarity. It should be noted that these simulations were conducted using an arbitrary Mach number solution algorithm instead of the compressible flow solver described in Chapter 2. The arbitrary Mach number version is a pre-conditioned version of the compressible flow solver and the pre-conditioning is described in references 97 and 98.
The objective of the work was to assess the capability of the current solver, including the sliding interface methodology, to simulate the complicated physics of these types of turbo-machinery flows and determine whether or not the general trend can be captured. This is an internal flow application and is an ideal case to assess the claim that global conservation is not required, and any significant mass leakage or increase due to the non-conservativeness of the interface would result in a poor quality solution. To study the stall inception and stall control, a full annulus simulation is required because
the periodicity assumption of the flow field is invalid since stall cells can rotate around
the annulus of the compressor [99]. Therefore the grid size and resolution requirements
to adequately resolve the flow near the impeller trailing edge make the UVI method
infeasible.

The configuration used in this work is a high-speed centrifugal compressor
designed by Allison Engine Company. The compressor stage contains an impeller and a
vane-island diffuser that has a 90-degree bend at the exit. The impeller contains 15 main
blades and 15 splitter blades and has 50 degrees of backsweep from radial at the
discharge. A detailed view of the impeller blades is shown in Figures 5.66 and 5.67. The
vane-island diffuser contains 24 passages. The stage was designed to produce a 4:1
pressure ratio at the design mass flow condition of 4.54 kg/s.
Figure 5.65  Leading edge of the impeller blades
5.3.1 Grid Construction

The geometric data and surface grid were prepared using SolidMesh and a mixed element anisotropic volume grid was generated using AFLR. The impeller and diffuser grids were generated separately and then combined to form a single grid. The total grid size is approximately 15 million nodes with 61.7 million elements. The impeller has 13.44 million nodes and the vane-island diffuser has 1.6 million nodes without the
injectors and 2.6 million nodes including the injectors. The impeller and diffuser components are coupled using the sliding interface during the solution algorithm. Figure 5.68 shows a cutting plane taken at the mid-span of the diffuser and the arrow indicates the location of the sliding interface. For this configuration, the impeller rotates and the diffuser is stationary.

Figure 5.67 Sliding interface on cutting plane of the compressor
5.3.2 Simulation Conditions and Parameters

The simulations were performed at 100% of the design speed. The compressor entrance conditions were used as the reference values to normalize the solution. The Reynolds number was $3.0387 \times 10^6$ based on the impeller exit diameter and entrance reference velocity and the reference Mach number was 0.32. Standard atmospheric pressure and temperature values were used as total conditions to specify the inflow condition. A constant back pressure corresponding to the near design flow condition was applied at the exit of the diffuser.

Two conditions were run; with and without tip injection. The solution was initialized using the reference values for density and pressure and zero velocity through the impeller. The simulations began in a steady-state mode in order to compute the impeller flow in a rotating reference frame and the diffuser flow in a fixed reference frame without having to actually rotate the impeller grid. The back pressure at the diffuser exit was gradually increased to the design operating condition. After about 10,000 iterations, the solution converged to steady-state and the simulations were switched to an unsteady simulation. Three Newton iterations were used at each time step and 16 linear subiterations were used for each Newton iteration. The time step used corresponded to one tenth of a degree of rotation per time step. Once the simulation was switched to a time-accurate simulation, three to five revolutions were required before the periodicity of the flow field was established.
5.3.3 Simulation Results

The first simulation was for the case without tip injection. Velocity contours across the sliding interface are shown in Figure 5.69 and are observed to be continuous across the interface. Surface pressure contours on the entire compressor are shown in Figure 5.70. Note the smooth pressure values, or rather the lack of discontinuities, at the sliding interface. Figure 5.71 compares the predicted total pressure ratio at near design conditions with measured data [100]. The predicted total pressure ratio is 3.81 and the predicted mass flow rate is 4.7 kg/s and both compare well to experimental data. The sliding interface appears to have minimal effect as the mass imbalance is less than 1%.
Figure 5.68  Computed velocity contours across the sliding interface
Figure 5.69  Computed surface pressure on impeller and vane-island diffuser
Figure 5.70  Total pressure ratio at design speed

The second simulation was for the case with tip injection. The locations of the eight injectors are shown in Figure 5.72 and correspond to the same locations as for the experimental configuration. The injectors are unevenly spaced along the circumference in the vaneless region of the vane-island diffuser and are located near the sliding interface. In the grid, each injector is represented by a circular surface that is placed slightly off the shroud surface as indicated in Figure 5.73. The direction of the injected air is tangent to the direction of rotation. Note that the injection direction can be either forward-tangent or reverse-tangent, and for this study the forward-tangent direction was
used. The injector mass flow was approximately 5% of the compressor design mass flow condition.

Figure 5.71 Diagram of the eight injector locations in the diffuser
Total velocity contours on a cutting plane through the center of the injector at a constant span location of the diffuser are shown in Figure 5.74. The high-speed air stream pushes the flow downstream through the throat of the diffuser. The maximum velocity is achieved at the diffuser throat area. Entropy contours are displayed in Figure 5.75 and note that the entropy value in the injector flow path is significantly higher than in the other passages not affected by the injected flow. For this internal flow application, the qualitative trends and changes in the flow structure due to the air injection are observed in the predicted solution using the sliding interface methodology. However
before any conservation issues due to the sliding interface can be addressed, there are several outstanding issues that affect the accuracy of the solution that must first be resolved, such as grid resolution requirements, accuracy of the turbulence model, and implementation of injection boundary conditions [101].

Figure 5.73  Predicted total velocity contours on a cutting plane passing through the vane-island diffuser
Figure 5.74  Predicted entropy contours on a cutting plane passing through the vane-island diffuser
CHAPTER VI

SUMMARY AND CONCLUSIONS

A sliding interface method has been developed for simulations involving relative rotational grid motion. The method alleviates computationally expensive grid deformation, remeshing, and hole cutting procedures. Rotational relative motion is accomplished by rigidly rotating a subdomain representing the moving component. At the subdomain interface boundary, the faces along the interfaces are extruded into the adjacent subdomain to create new volume elements and provide a one-cell overlap. These new volume elements close the control volumes for the nodes on the interface surface and allow a flux to be computed across the subdomain interface. An interface flux is computed independently for each subdomain, and in doing so the method is not strictly conservative across the subdomain interface. The values of the solution variables and other quantities for the nodes created by the extrusion process are found by interpolation. The extrusion is done so that the interpolation will maintain information as localized as possible. A parallel implementation of the neighbor search is used to find the extruded points in the adjacent subdomain.

The method has been implemented in a parallel, node-centered finite volume, high-resolution viscous flow solver. The method developed is efficient and has been shown to be faster than a current state-of-the-art method, the UVI method, for
unstructured grid applications. The method does not impose any restrictions on the subdomain interface aside from the requirement that the axisymmetric limitation required for rotational relative motion. The grid on the subdomain interface can be arbitrary. The boundary surfaces between the two subdomains can have completely independent grids from one another; meaning they do not have to connect in a one-to-one manner and no symmetry or pattern restrictions are placed on the surface grid.

To address interface flux conservation and validate the method, numerical simulations were performed on a shear flow (steady, inviscid, incompressible), flow through a diverging duct (steady, inviscid), supersonic flow over a ramp (steady, inviscid), a shock tube (unsteady, inviscid), a cylinder in crossflow (steady and unsteady, viscous, incompressible), and a marine propeller (unsteady, viscous, incompressible). Grids without a sliding interface were constructed for comparison. In the shear flow case the discontinuity was maintained using the sliding interface. For the diverging duct, the net mass flow rate using the sliding interface was virtually identical to that for the baseline grid. Rotating the sliding interface had no effect on the mass flow rate either. The supersonic flow over a ramp was used to examine discontinuities across the interface. An interface was placed midway up the ramp. The resulting shock wave passed through the interface and the computed results show that the properties in front of and behind the shock matched very well with theoretical results. The shock tube computations examined unsteady waves passing through the interface. The results using the sliding interface agreed well with the analytical solution and are nearly identical to the solution without an interface. For the cylinder in crossflow, the lower Reynolds number case showed similar pressure and velocity fields for the sliding interface solution.
and the location of the vortex cores behind the cylinder were the same as the baseline case and compare very well to the experimental data. In the higher Reynolds number case, the predicted Strouhal number using the sliding interface was identical to the baseline grid and agreed very well with the experimentally measured value. For the marine propeller, the solution using the sliding interface predicted the same axial or thrust force coefficient as the baseline grid, and the differences in mass flow across the interface surfaces was shown to be very small.

The sliding interface method was also demonstrated and validated on three large-scale geometrically complex cases; these cases are the 1) spinning missile with dithering canards, 2) missile with free-spinning tail, and 3) high-speed centrifugal compressor. The predicted results for the spinning missile with dithering canards using the sliding interface agreed well with available experimental data in the form of roll-averaged force and moment coefficients. Comparisons of the time histories of the force and moment coefficients were made with the current state-of-the-art method for unstructured grid topologies (UVI) and with another high-resolution viscous flow solver (OVERFLOW-D), and the sliding interface results were found to agree very well with both. The free-spinning tail missile case highlighted some of the advantages of the sliding interface method compared to the UVI method. The sliding interface required no geometric modifications and had significantly shorter run times. Similar comparisons of the roll-averaged and time history data were made to the UVI and OVERFLOW-D numerical solutions and excellent agreement was found. The high-speed centrifugal compressor case illustrated an example that cannot be solved using the UVI method and necessitated the use of the sliding interface method. The predicted total pressure ratio and mass flow
rate was found to agree well with measured data. For both the small-scale model problems and the large-scale applications, the numerical solutions were found to be in excellent agreement with analytical solutions and experimental results. Thus, there are no apparent adverse effects on the numerical solutions by not strictly enforcing flux conservation at the subdomain boundary.

A sliding interface method has been successfully demonstrated for problems having relative motion on unstructured grid topologies. Future work might include investigating different updating strategies to decrease some of the parallel communication costs and to investigate ways to enforce flux conservation across the interface.
REFERENCES


[91] Hyams, D. G., Private conversations, ERC Computational Simulation and Design Center, Mississippi State, MS, September 2004.


[101] Sheng, C., Private conversations, ERC Computational Simulation and Design Center, Mississippi State, MS, September 2004.
APPENDIX A

EIGENSYSTEM FOR THE CONVECTIVE FLUX JACOBIAN MATRIX
The eigensystem (eigenvectors and eigenvalues) of the convective flux Jacobian is required by the Roe flux formulation to properly construct the Roe dissipation matrix $\tilde{A}$ [73]. The eigensystem is also required by the characteristic variable approach for the farfield boundary condition implementation. The eigenvalues of the flux Jacobian are

$$\begin{align*}
\lambda_1 &= \lambda_2 = \lambda_3 = \Theta \\
\lambda_4 &= \Theta + c \\
\lambda_5 &= \Theta - c
\end{align*}$$  
(A.1)

where $\Theta$ is the contravariant velocity and $c$ is the local speed of sound.

The eigenvectors and eigenvalues of the flux Jacobian can be used to diagonalize $A$ and are related to the flux Jacobian by

$$A = T \Lambda T^{-1}$$  
(A.4)

where $T$ is the matrix of right eigenvectors of $A$ and $\Lambda$ is diagonal matrix containing the eigenvalues of $A$. With regard to computing the eigenvectors of the flux Jacobian, it is easier to work with the Jacobian using primitive variables or the non-conservative Jacobian, $\hat{A}$, since it has a simple structure. Given the eigenvectors of $\hat{A}$, they can be transformed to find the eigenvectors of $A$. The matrix of transformation from the conservative to the non-conservative is defined by

$$M = \frac{\partial Q}{\partial q}$$  
(A.5)

The conservative and non-conservative Jacobians are related by the similarity transformation

$$\hat{A} = M^{-1} A M \quad \text{or} \quad A = \hat{M} A \hat{M}^{-1}$$  
(A.6)
Since the matrices $A$ and $\hat{A}$ are similar, they share the same eigenvalues and their eigenvectors are related by

$$T = MR$$

(A.7)

where $R$ is the matrix of right eigenvectors of $\hat{A}$. The left eigenvectors of $A$ are therefore

$$T^{-1} = R^{-1}M^{-1}$$

(A.8)

The eigenvectors are based on a normalized vector $\hat{n}$, which is the outward pointing normal of the control volume face. The matrices used in the above expressions to construct the left and right eigenvectors are defined as follows:

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 & 0 \\ 0 & 0 & \lambda_3 & 0 & 0 \\ 0 & 0 & 0 & \lambda_4 & 0 \\ 0 & 0 & 0 & 0 & \lambda_5 \end{bmatrix}$$

(A.9)

$$R = \begin{bmatrix} \hat{n}_x & \hat{n}_y & \hat{n}_z & \rho/c & \rho/c \\ 0 & -\hat{n}_z & \hat{n}_y & \hat{n}_x & -\hat{n}_x \\ \hat{n}_z & 0 & -\hat{n}_x & \hat{n}_y & -\hat{n}_y \\ -\hat{n}_y & \hat{n}_x & 0 & \hat{n}_z & -\hat{n}_z \\ 0 & 0 & 0 & \rho c & \rho c \end{bmatrix}$$

(A.10)

$$R^{-1} = \begin{bmatrix} \hat{n}_x & 0 & \hat{n}_z & -\hat{n}_y & -\hat{n}_x/c^2 \\ \hat{n}_y & -\hat{n}_z & 0 & \hat{n}_x & -\hat{n}_y/c^2 \\ \hat{n}_z & \hat{n}_y & -\hat{n}_x & 0 & -\hat{n}_z/c^2 \\ 0 & \hat{n}_x/2 & \hat{n}_y/2 & \hat{n}_z/2 & 1/(2\rho c) \\ 0 & -\hat{n}_x/2 & -\hat{n}_y/2 & -\hat{n}_z/2 & 1/(2\rho c) \end{bmatrix}$$

(A.11)
where $V$ is the velocity magnitude, $V = \sqrt{u^2 + v^2 + w^2}$. Substituting Equations A.12 and A.10 into Equation A.7 yields the expression for the matrix of right eigenvectors of the flux Jacobian $A$:

$$
M = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
u & \rho & 0 & 0 & 0 \\
v & 0 & \rho & 0 & 0 \\
w & 0 & 0 & \rho & 0 \\
\frac{V^2}{2} & \rho u & \rho v & \rho w & \frac{1}{\gamma-1}
\end{bmatrix} \quad (A.12)
$$

$$
M^{-1} = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
-u/\rho & 1/\rho & 0 & 0 & 0 \\
-v/\rho & 0 & 1/\rho & 0 & 0 \\
-w/\rho & 0 & 0 & 1/\rho & 0 \\
(\gamma-1)V^2/2 & -u(\gamma-1) & -v(\gamma-1) & -w(\gamma-1) & (\gamma-1)
\end{bmatrix} \quad (A.13)
$$

where $\gamma = \frac{1}{\sqrt{\alpha}}$. In order to compute the Roe dissipation matrix, the above matrices are evaluated using Roe averaged variables, that is $\tilde{A} = \tilde{T} \tilde{\Lambda} \tilde{T}^{-1}$. 

$$
\begin{bmatrix}
\hat{n}_x & \hat{n}_y & \hat{n}_z & \frac{\rho}{c} & \frac{\rho}{c} \\
u \hat{n}_x & u \hat{n}_y - \rho \hat{n}_z & u \hat{n}_z + \rho \hat{n}_y & \frac{\rho}{c} (u + c \hat{n}_x) & \frac{\rho}{c} (u - c \hat{n}_x) \\
v \hat{n}_x & v \hat{n}_y & v \hat{n}_z - \rho \hat{n}_x & \frac{\rho}{c} (v + c \hat{n}_y) & \frac{\rho}{c} (v - c \hat{n}_y) \\
w \hat{n}_x - \rho \hat{n}_y & w \hat{n}_y + \rho \hat{n}_x & w \hat{n}_z & \frac{\rho}{c} (w + c \hat{n}_z) & \frac{\rho}{c} (w - c \hat{n}_z) \\
\frac{V^2}{2} \hat{n}_x + \frac{V^2}{2} \hat{n}_y + \frac{V^2}{2} \hat{n}_z & \frac{V^2}{2} \hat{n}_y + \frac{V^2}{2} \hat{n}_z & \frac{V^2}{2} \hat{n}_y + \frac{V^2}{2} \hat{n}_z & \rho \left( \frac{V^2}{2c} + \Theta + \alpha \right) & \rho \left( \frac{V^2}{2c} - \Theta + \alpha \right)
\end{bmatrix}
\quad (A.14)
$$

where $\alpha = \frac{c}{\gamma-1}$. In order to compute the Roe dissipation matrix, the above matrices are evaluated using Roe averaged variables, that is $\tilde{A} = \tilde{T} \tilde{\Lambda} \tilde{T}^{-1}$. 

\[
\frac{V^2}{2} \hat{n}_x + \frac{V^2}{2} \hat{n}_y + \frac{V^2}{2} \hat{n}_z & \frac{V^2}{2} \hat{n}_y + \frac{V^2}{2} \hat{n}_z & \frac{V^2}{2} \hat{n}_y + \frac{V^2}{2} \hat{n}_z & \rho \left( \frac{V^2}{2c} + \Theta + \alpha \right) & \rho \left( \frac{V^2}{2c} - \Theta + \alpha \right)
\]
APPENDIX B

VOLUME WEIGHTS FOR TETRAHEDRAL AND NON-SIMPLICAL ELEMENTS
The volume weights are used by the search algorithm to determine if a point P lies within a given element, and if not, which direction the search should proceed to find the point. Volume weights are the sub-volumes formed by connecting the nodes of each face of the element to point P. The sub-volumes are either a tetrahedral if the element face is a triangle or a five-node pentahedra, or pyramid, if the element face is a quadrilateral. The element connectivity for a tetrahedron, five-node pentahedron, six-node pentahedron, and hexahedron are given in Figures B.1 – B.4, respectively, and the face and sub-volume connectivity are listed in Tables B.1 – B.4. The volume of a tetrahedron is computed by

\[
V_{\text{tet}} = \frac{1}{6} [\vec{r}_{41} \cdot (\vec{r}_{21} \times \vec{r}_{31})]
\]  

(B.1)

where

\[
\vec{r}_{21} = \vec{x}_2 - \vec{x}_1
\]  

(B.2)

\[
\vec{r}_{31} = \vec{x}_3 - \vec{x}_1
\]  

(B.3)

\[
\vec{r}_{41} = \vec{x}_4 - \vec{x}_1
\]  

(B.4)

and \( \vec{x}_i \) are the coordinates of node \( i \). For the volume of a pyramid element, it is divided into four tetrahedra and the volume is taken as the average of the four tetrahedra volumes:

\[
V_{\text{pyramid}} = \frac{1}{8} \left( V_{1423} + V_{4523} \right) + \frac{1}{8} \left( V_{1453} + V_{1523} \right)
\]  

(B.5)

where the indices refer to the nodes that define the connectivity of the pyramid element listed in Figure B.2.
Figure B.1 Tetrahedron connectivity

Table B.1 Tetrahedron volume weight definition

<table>
<thead>
<tr>
<th>Face</th>
<th>Face Connectivity</th>
<th>Sub-Volume Connectivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1-2-3</td>
<td>1-2-3-P</td>
</tr>
<tr>
<td>2</td>
<td>1-4-2</td>
<td>1-4-2-P</td>
</tr>
<tr>
<td>3</td>
<td>1-3-4</td>
<td>1-3-4-P</td>
</tr>
<tr>
<td>4</td>
<td>2-4-3</td>
<td>2-4-3-P</td>
</tr>
</tbody>
</table>
Figure B.2 Five-node pentahedron element (Pyramid) connectivity

Table B.2 Five-node pentahedron element (Pyramid) volume weight definition

<table>
<thead>
<tr>
<th>Face</th>
<th>Face Connectivity</th>
<th>Sub-Volume Connectivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1-4-5-2</td>
<td>1-2-P-4-5</td>
</tr>
<tr>
<td>2</td>
<td>1-2-3</td>
<td>1-2-3-P</td>
</tr>
<tr>
<td>3</td>
<td>1-3-4</td>
<td>1-3-4-P</td>
</tr>
<tr>
<td>4</td>
<td>2-5-3</td>
<td>2-5-3-P</td>
</tr>
<tr>
<td>5</td>
<td>3-5-4</td>
<td>3-5-4-P</td>
</tr>
</tbody>
</table>
Figure B.3 Six-node pentahedron element (Prism) connectivity

Table B.3 Six-node pentahedron element (Prism) volume weight definition

<table>
<thead>
<tr>
<th>Face</th>
<th>Face Connectivity</th>
<th>Sub-Volume Connectivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1-4-5-2</td>
<td>1-2-P-4-5</td>
</tr>
<tr>
<td>2</td>
<td>1-2-3</td>
<td>1-2-3-P</td>
</tr>
<tr>
<td>3</td>
<td>1-3-6-4</td>
<td>3-1-P-6-4</td>
</tr>
<tr>
<td>4</td>
<td>2-5-6-3</td>
<td>2-3-P-5-6</td>
</tr>
<tr>
<td>5</td>
<td>4-6-5</td>
<td>4-6-5-P</td>
</tr>
</tbody>
</table>
Figure B.4 Hexahedron element connectivity

Table B.4 Hexahedron element volume weight definition

<table>
<thead>
<tr>
<th>Face</th>
<th>Face Connectivity</th>
<th>Sub-Volume Connectivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1-2-3-4</td>
<td>2-1-P-3-4</td>
</tr>
<tr>
<td>2</td>
<td>1-5-6-2</td>
<td>5-1-P-6-2</td>
</tr>
<tr>
<td>3</td>
<td>1-4-8-5</td>
<td>4-1-P-8-5</td>
</tr>
<tr>
<td>4</td>
<td>2-6-7-3</td>
<td>2-3-P-6-7</td>
</tr>
<tr>
<td>5</td>
<td>3-7-8-4</td>
<td>7-3-P-8-4</td>
</tr>
<tr>
<td>6</td>
<td>5-8-7-6</td>
<td>5-6-P-8-7</td>
</tr>
</tbody>
</table>
APPENDIX C

FORCE AND MOMENT COEFFICIENT PLOTS FOR THE SPINNING MISSILE WITH DITHERING CANARDS
Figure C.1 Component contributions: axial force coefficient for the 0% command level

Figure C.2 Component contributions: side force coefficient for the 0% cmd level
Figure C.3 Component contributions: normal force coefficient for the 0% cmd level

Figure C.4 Component contributions: rolling moment coefficient for the 0% cmd level
Figure C.5 Component contributions: pitching moment coefficient for the 0% cmd level

Figure C.6 Component contributions: yawing moment coefficient for the 0% cmd level
Figure C.7 Axial force coefficient for the Exp-fit canard command level

Figure C.8 Side force coefficient for the Exp-fit canard command level
Figure C.9 Normal force coefficient for the Exp-fit canard command level

Figure C.10 Rolling moment coefficient for the Exp-fit canard command level
Figure C.11  Pitching moment coefficient for the Exp-fit canard command level

Figure C.12  Yawing moment coefficient for the Exp-fit canard command level
APPENDIX D

FORCE AND MOMENT COEFFICIENT PLOTS FOR THE MISSILE
WITH THE FREE-SPINNING TAIL
Figure D.1 Component contributions: axial force coefficient $\omega = 2500$ rpm

Figure D.2 Component contributions: side force coefficient $\omega = 2500$ rpm
Figure D.3  Component contributions: normal force coefficient $\omega = 2500$ rpm

Figure D.4  Component contributions: rolling moment coefficient $\omega = 2500$ rpm
Figure D.5 Component contributions: pitching moment coefficient $\omega = 2500$ rpm

Figure D.6 Component contributions: yawing moment coefficient $\omega = 2500$ rpm
Figure D.7 Axial force coefficient for the $\omega = 5000$ rpm tail spin rate

Figure D.8 Side force coefficient for the $\omega = 5000$ rpm tail spin rate
Figure D.9 Normal force coefficient for the $\omega = 5000$ rpm tail spin rate

Figure D.10 Rolling moment coefficient for the $\omega = 5000$ rpm tail spin rate
Figure D.11 Pitching moment coefficient for the $\omega = 5000$ rpm tail spin rate

Figure D.12 Yawing moment coefficient for the $\omega = 5000$ rpm tail spin rate
Figure D.13  Axial force coefficient for the $\omega = 1980$ rpm tail spin rate

Figure D.14  Side force coefficient for the $\omega = 1980$ rpm tail spin rate
Figure D.15  Normal force coefficient for the $\omega = 1980$ rpm tail spin rate

Figure D.16  Rolling moment coefficient for the $\omega = 1980$ rpm tail spin rate
Figure D.17 Pitching moment coefficient for the $\omega = 1980$ rpm tail spin rate

Figure D.18 Yawing moment coefficient for the $\omega = 1980$ rpm tail spin rate