I, Lihua Chen, hereby submit this as part of the requirements for the degree of:

Doctor of Philosophy (Ph.D)

in:
Department of Mechanical, Industrial and Nuclear Engineering

It is entitled:
THREE-DIMENSIONAL PHYSIOLOGICAL FLOW SIMULATION
ON MULTI-BOX COMPUTATIONAL DOMAINS

Approved by:
Dr. Urmila Ghia
Dr. Kirti Ghia
Dr. Milind Jog
THREE-DIMENSIONAL PHYSIOLOGICAL FLOW SIMULATION ON
MULTI-BOX COMPUTATIONAL DOMAINS

A dissertation submitted to the
Division of Research and Advanced Studies
of the University of Cincinnati

in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY (Ph.D.)

In the Department of Mechanical, Industrial and Nuclear Engineering of the
College of Engineering

2003

by

Lihua Chen

B.S., National College of Marine Science and Technology, 1983
M.S., National Cheng-Kung University, 1985

Committee Chair: Dr. Urmila Ghia
Abstract

The fluid flow phenomena in biological systems are typically complex. The complexity is originated from, for example, non-Newtonian behavior of body fluids, complicated geometry, as well as the interaction of muscle and fluid. With the advent of modern computational technology, both in hardware and software, gradually these problems can be resolved. The present research illustrates two such examples.

Grid generation is a branch of applied mathematics that is essential for conducting numerical simulation of fluid flow. In this research, a new grid generation technique is developed and implemented in a flow solver. This technique enables one to perform grid generation for complex geometry using only a single computational zone. Fluid flow can then be analyzed without iteration between zones.

The scheme is based on the composite transformation of an algebraic mapping and a mapping governed by the Laplace equation. The governing equations for the grid generation are derived first and then solved numerically. The scheme used for solving the grid generating equations is an extension of the traditional three-dimensional Douglass-Gunn scheme. Areas of extension include the inclusion of mixed derivative terms as well as first-order derivative terms.

A unique feature of the proposed grid generation scheme is the concept of multi-box computational domains. In this scheme, the physical domain is mapped onto a geometry composed of many boxes in the computational space, rather than a single box as the traditional method does. The numerical solution routine is adjusted accordingly to accommodate this new feature.
Grids were generated for two model geometries using the proposed grid generation software. The graft model features one inflow conduit and two outflow conduits, while the left atrium (LA) model has four inflow conduits and one outflow conduit.

Flow simulation was performed using the research code INS3D, which employs the method of artificial compressibility. This method transforms the Navier-Stokes equations into a hyperbolic-parabolic set by adding to them pseudo-pressure gradient terms. The scheme is then marched along the pseudo-time axis, until the velocity field becomes divergence-free.

For the flow simulation in side the graft, the effect of Reynolds number and flow-division ratio is examined. The Reynolds number effect is, as expected, demonstrated via the presence of a helical flow structure as well as the overall pressure drop. The flow-division ratio, on the other hand, alters the flow field in a way that moves the stagnation points. In particular, the flow pattern for the case with 50:50 flow-division ratio closely resembles that observed clinically, and the highlighted low wall stress area on the hood and toe of the reinforce strengthen the hypothesis about the formation of intimal hyperplasia. The complicated flow field demonstrated by the case with 100:0 division ratio, corresponding to an occluded distal artery, demonstrated that three-dimensional numerical simulation of the flow field can assist in interpreting data from a PIV (Particle Image Velocimetry) experimental session.

The steady-state simulation of the flow field in the left atrium of the heart was another subject of interest. Although steady-state simulation is not as realistic as time accurate simulation, it nevertheless gives information on the long-term performance of the chamber. The simulation shows the existence of low wall shear regions. These low shear stress areas in the chamber are areas susceptible to blood clot formation. In fact, clinical evidence shows that of certain strokes are indeed caused by clots forming in the atrium and traveling through the arterial
system and essentially lodging in the brain. Since this phenomenon is geometry-related and there
is no practical way to alter it, common therapy for such conditions is to administer certain ‘blood
thinners’ (Anticoagulation agenes) to reduce the possibility of blood clot formation.

In summary, the present research demonstrates applications of computational fluid
dynamics technique in the analysis of flow in biological system. A new grid generation
technique is realized, and proved to be very useful in simulating these flows. Flow simulation
results provide insights into the system and may be of use for clinic reference.
Acknowledgements

I sincerely thank my advisor, Dr. Urmila Ghia, for her continuous support. As an academic mentor, she provided invaluable guidance throughout my doctoral study. To me, she is not just an academic advisor, but also a patient encourager and supporter. She really made a great impact in my life. Thanks also go to Dr. Kirti Ghia, for his wisdom, advice and supervision. I especially appreciate Dr. Milind Jog for being on the committee and providing invaluable suggestions.

Further thanks go to all of my colleagues in CFDRL for their helping hands. I really enjoyed the time we spent together in discussion, exploration, learning and sometimes arguing. They played an important role along the way in my research.

I am indebted to my wife, Hueiwen, for taking good care of me, day and night. Her support and encouragement are really the motivation for me to move on. She deserves to be a 'PHT' (Push Husband Through). A special thanks to my sons, Tommy and Josh, for being good boys, and making me laugh when I am facing challenges. Thanks to my parents for their expectation and support.

Finally, I thank God who made all this possible.
Table of Contents

ABSTRACT ................................................................................................. i
ACKNOWLEDGEMENTS ........................................................................ iv
TABLE OF CONTENTS ........................................................................... v
LIST OF FIGURES .................................................................................. viii
LIST OF TABLES .................................................................................... xi
NOMENCLATURE .................................................................................... xii

CHAPTER 1 INTRODUCTION AND BACKGROUND INFORMATION
..................................................................................................................1

1.1 Description of the Human Circulatory System and the Left Atrium of Heart .................................1
1.2 Literature Survey and Unresolved Issues ......................... 5
1.3 Computational Challenges ..............................................................9
1.4 Objectives and Plan ................................................................. 11

CHAPTER 2 MATHEMATICAL DESCRIPTION OF THE FLOW PROBLEM .............................................. 13

2.1 Governing Equations in Vector Form ......................................13
2.2 The Boundary-Fitted Generalized Coordinate System .......16
2.3 Governing Equation in Generalized Coordinate System ......21
2.4 Boundary Conditions and Initial Conditions .........................23
CHAPTER 3   NUMERICAL GRID GENERATION  ..........27
  3.1 Grid Generation Equations  ........................................27
      3.1.1 Grid Generation Equations for Initial and Stationary Grid
           Systems .........................................................27
      3.1.2 Time-Varying Grid Generation - the Grid-Transport Equations
           ............................................................................43
  3.2 Extensions to Multi-Box Scheme  ..................................53
      3.2.1 Multi-Box Scheme  ..................................................53
      3.2.2 The H-H Topology ....................................................55
  3.3 Numerical Solution of the Grid-Transport Equations ..........60
      3.3.1 Douglass-Gunn ADI ..................................................60
      3.3.2 Linearization ..........................................................66
      3.3.3 Thomas Algorithm with Blanking  ...............................74

CHAPTER 4   NUMERICAL SOLUTION OF FLOW EQUATIONS
                ....................................................................77
  4.1 Numerical Schemes for the Navier-Stokes Equations ........77
  4.2 Method of Artificial Compressibility  ............................83
  4.3 Implementation – INS3D .................................................86
      4.3.1 The Iteration Process ...............................................86
      4.3.2 Upwind Differencing .................................................90
      4.3.3 Implicit Scheme ......................................................96

CHAPTER 5   SIMULATION OF FLOWS IN VASCULAR GRAFT
              ANASTOMOSIS ............................................... 101
  5.1 Introduction ............................................................101
  5.2 Results and Discussion ..............................................106
CHAPTER 6  FLOW FIELD IN LEFT ATRIUM  ............... 124
  6.1 Review of Cardiac Events  ...................................... 124
  6.2 Some Physiological Conditions Associated with Left Atrium
  .................................................................................129
  6.3 The Atrium Model and Initial/Boundary Conditions ...........132
  6.4 Flow Field in Left Atrium .........................................135

CHAPTER 7  CONCLUSION AND RECOMMENDATIONS.
  ..................................................................................140
  7.1 Computational Accomplishments and Conclusion ............... 140
  7.2 Recommendations .................................................142

REFERENCES  ....................................................................... 145
## List of Figures

<table>
<thead>
<tr>
<th>Figures</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Schematic Representation of Circulatory System</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>The Human Heart</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>Incomplete Cells</td>
<td>17</td>
</tr>
<tr>
<td>4</td>
<td>Two Dimensional Mapping</td>
<td>31</td>
</tr>
<tr>
<td>5</td>
<td>Physical domain D and the bounding faces</td>
<td>35</td>
</tr>
<tr>
<td>6</td>
<td>Three Dimensional Mapping</td>
<td>36</td>
</tr>
<tr>
<td>7a</td>
<td>Traditional Mapping</td>
<td>54</td>
</tr>
<tr>
<td>7b</td>
<td>Multi-rectangular Mapping</td>
<td>54</td>
</tr>
<tr>
<td>8</td>
<td>H-Grid Topology</td>
<td>55</td>
</tr>
<tr>
<td>9</td>
<td>Singular point on a smooth boundary</td>
<td>56</td>
</tr>
<tr>
<td>10</td>
<td>O-Grid Topology</td>
<td>57</td>
</tr>
<tr>
<td>11</td>
<td>H-Grid Topology</td>
<td>57</td>
</tr>
<tr>
<td>12</td>
<td>One Other Topology Setup</td>
<td>57</td>
</tr>
<tr>
<td>13</td>
<td>3-D grid generation by translation</td>
<td>58</td>
</tr>
<tr>
<td>14</td>
<td>3-D grid generation by rotation</td>
<td>58</td>
</tr>
<tr>
<td>15</td>
<td>Junction of Two Pipe</td>
<td>59</td>
</tr>
<tr>
<td>16</td>
<td>Flowchart for stationary grid generation</td>
<td>71</td>
</tr>
<tr>
<td>17</td>
<td>Flowchart for time-dependent grid generation</td>
<td>72</td>
</tr>
<tr>
<td>18</td>
<td>Flowchart for grid generation with time-dependent boundary</td>
<td>73</td>
</tr>
<tr>
<td>19</td>
<td>1-D domain with blank-out area</td>
<td>74</td>
</tr>
<tr>
<td>20</td>
<td>Effect of the number of sweeps</td>
<td>99</td>
</tr>
<tr>
<td>21</td>
<td>Graft Geometry</td>
<td>102</td>
</tr>
<tr>
<td>22</td>
<td>Computational Domain</td>
<td>103</td>
</tr>
<tr>
<td>23</td>
<td>Physical Domain (Model 1)</td>
<td>104</td>
</tr>
<tr>
<td>24</td>
<td>H-H Topology</td>
<td>104</td>
</tr>
<tr>
<td>25</td>
<td>H-H Topology</td>
<td>104</td>
</tr>
<tr>
<td>Page</td>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>------</td>
<td>----------------------------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>32</td>
<td>Cardiac Cycle</td>
<td>124</td>
</tr>
<tr>
<td>33</td>
<td>Cardiac Cycle – PV diagram</td>
<td>125</td>
</tr>
<tr>
<td>34</td>
<td>Mitral Valve Stenosis</td>
<td>129</td>
</tr>
<tr>
<td>35</td>
<td>Mitral Valve Regurgitation</td>
<td>130</td>
</tr>
<tr>
<td>36</td>
<td>Model used in the simulation</td>
<td>135</td>
</tr>
<tr>
<td>37a</td>
<td>Symmetric Cutting Plane (P1)</td>
<td>136</td>
</tr>
<tr>
<td>37b</td>
<td>Velocity vector on cutting plane P1</td>
<td>136</td>
</tr>
<tr>
<td>37c</td>
<td>2D Streamlines on cutting plane P1</td>
<td>137</td>
</tr>
<tr>
<td>38</td>
<td>3D Streamlines</td>
<td>137</td>
</tr>
<tr>
<td>39a</td>
<td>Cutting Plane (P2)</td>
<td>138</td>
</tr>
<tr>
<td>39b</td>
<td>Velocity vectors on cutting plane P2</td>
<td>138</td>
</tr>
<tr>
<td>39c</td>
<td>2D Streamline on cutting plane</td>
<td>139</td>
</tr>
<tr>
<td>40</td>
<td>Surface vorticity contour</td>
<td>139</td>
</tr>
<tr>
<td>41</td>
<td>Internal grid clustering</td>
<td>142</td>
</tr>
</tbody>
</table>
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Model Geometry for Graft</td>
<td>102</td>
</tr>
<tr>
<td>2</td>
<td>Non-Dimensional Model Geometry of Graft</td>
<td>103</td>
</tr>
<tr>
<td>3</td>
<td>Steady-State Simulation Flow Conditions</td>
<td>106</td>
</tr>
</tbody>
</table>
CHAPTER 1
INTRODUCTION AND BACKGROUND INFORMATION

The circulatory system is one of the most important systems in the human body. Knowledge of this system plays a vital role in preventing and managing diseases associated with it. A circulatory system may suffer from various challenging factors, which may be biological, chemical or mechanical in nature. Biological factors such as genetic disorders or bacteria or virus infections may cause severe illness. Malfunction in pH control, such as a high uric acid concentration, is an example of chemical factors that damage normal operation of the circulatory system. The present research confines itself to diseases associated with mechanical factors only, examples of which will be given in the next section.

In the following, Section 1 provides a brief overview of the human circulatory system and details of the function of the left atrium of human heart. Section 2 presents a general survey of past efforts towards addressing cardiovascular issues. The motivation of the present research is thus towards one of the unsolved problems. The use of computational techniques in addressing fluid dynamics problem is the main theme of this research and, hence, the associated computational challenge will be discussed in Section 3. Section 4 provides an overall plan for conducting the problem solution.

1.1 Description of the Human Circulatory System and the Left Atrium of Heart

The human cardiovascular system has two major components: the heart and the blood vessels. The heart acts like a pump, and is divided into four chambers, namely, left and right ventricles, and left and right atria. The vessels are comprised of the artery trees, capillaries and venous trees, and are categorized into two circulatory systems: systemic circulation and pulmonary circulation. The systemic circulation starts from the left ventricle, which pumps
blood into the aorta. In the systemic circulation, the arterial trees gradually decrease in diameter from the aorta to arteries to arterioles. At the end of arteriole is the capillary in which major mass transfer and oxygen/carbon dioxide exchange take place. The blood vessel then gradually increases in diameter from the venule to the vein to the vena cava. The vena cava is connected to the right atrium of the heart, which concludes the systemic circulation. The blood then passes through the tricuspidal valve and fills the right ventricle. It is then ejected into the pulmonary artery, and begins the journey of the pulmonary circulation. In this pulmonary circulation, the diameter of the blood vessels decreases from the pulmonary artery to arterioles to capillaries and then increases in the veins and back into left atrium of heart. The blood is rejuvenated in the pulmonary capillaries inside the lungs. Finally, after crossing the mitral valve, the fluid returns to the left ventricle where it started. For the normal human circulatory system, the whole journey takes about a minute. Figure 1 presents a schematic of the pulmonary and systemic circulation systems. Figure 2 shows a cross section of a human heart with flow directions indicated.

Figure 1. Schematic Representation of Circulatory System.
The ventricles do the major pumping for the heart. The left ventricle pumps blood into systemic circulation, and thereby delivers oxygen to the body, while the right ventricle pumps blood into pulmonary circulation to help discharge carbon dioxide. The “systolic” phase of the heart cycle refers to the part of the cardiac cycle in which the ventricular muscle is contracted, and the blood is squeezed out of the heart. The volume of blood per ‘squeeze’ is defined as ‘stroke volume’. In the “diastolic” phase of the heart cycle, the ventricular muscle is relaxed, and blood from the atrium is filling in the ventricle. The direction of the flow is controlled by a set of valves upstream and downstream of the ventricle. For the left ventricle, the upstream control valve is named the mitral valve, and the downstream control valve, the aortic valve. For right side of the heart, the tricuspidal and pulmonal valves refer to controls upstream and downstream of right ventricle respectively.
The volume of the ventricle is thus a continuous function of time. For a healthy heart, this function is periodic. It can be easily grasped that, late in the diastolic filling, the increase in the ventricle volume is less rapid than that during the early diastolic filling. The effect of the difference in the filling rate is less significant for normal heart rates (approximately 70 beats per minute), but is more pronounced for higher heart rates (approximately 150 beats per minute). It is at this stage that the role of the atrium becomes important. While the functions of the atrium are manifold, fluid dynamicists perceive the atrium as an element that increases the supply of blood to the ventricle during the late diastole-phase of the cycle, thereby producing higher stroke volume. The flow in the arterial system, hence, undergo two velocity peaks, namely E and A waves for early and late diastolic filling, respectively.

As mentioned above, the function of atrium is to increase the ventricle filling rate in the late diastolic phase. In fact, atrial contraction accounts for 30% of the total stroke volume in a cardiac cycle. In addition to this contribution to the circulation system, atrial contraction also helps reduce the sensitivity (and, thus, increase stability) of cardiac performance on the characteristics of individual components of the cardiac system. Zacek [1] reported that for example a 500% change in aortic compliance results in only a few percent shift in the isovolumic pressure in the ventricle. This stabilizing effect can be attributed to the more ‘uniform’ flow pattern across the mitral valve, and indeed, it helps reduce the work-load of the ventricular muscle and, thus, increase the life span of the human heart.

Several physiological conditions are associated with atrial function. Section 6.2 will provide some clinically encountered common problems associated with left atrium performance. Section 6.4 presents some results for the flow in the atrium.
1.2 Literature Survey and Unresolved Issues

A large body of research is being pursued in the area of cardiovascular fluid dynamics, also referred to as hemodynamics, because many severe disorders are associated the hemodynamic conditions in the human circulatory system. Heart attack, for example, the number one killer in many industrialized countries, is related to the hemodynamic conditions in coronary arteries whereas stroke, another life-threatening disease for aging people, is connected to the flow field in the carotid artery.

The two above-mentioned diseases have one thing in common: flow passage blockage. As a result, an enormous amount of effort is directed toward research of flow fields in constricted passages. Young has reported analytical studies as early as 1968 [2]. Lee and Fung [3,4] have conducted a pioneer work in numerical simulation of these flows. A benchmark experimental result of Forester and Young [5] has been used as reference in the work of Thornburg [6]. Stenosed vessels (vessels with constrictions) may occur for several reasons. One of the most common causes is lipid accumulation. Generally speaking, flow though a vessel with stenosis has a higher viscous loss, the effect of which is of twofold. First, a higher-pressure field upstream is required to maintain the given flow rate. In other words, the loading on the heart is increased. Secondly, in case of insufficient blood supply downstream of the stenosis, body tissue or cells may become deceased due to lack of oxygen and/or nutrition been furnished to it. The abrupt change in wall shear stress upstream and downstream of the stenotic area has also been reported to relate to the damage of arterial lumen. [7]

Research effort is aimed toward unsteady flow phenomena. An experiment using Laser Doppler Anemometry (LDA) on pulsatile post-stenotic flow field is described by Ahmed [8] where the absence of a permanent region of separation is remarkably noted. The application of the Magnetic Resonance Imaging (MRI) technique in velocity profile measurement for flow in a
stenotic tube has also been reported [9]. Cheng et al. [10] performed numerical simulation of flow past a square or rectangular constriction with pulsatile as well as oscillatory inflow.

Shear stress in biological flow fields is very important. First of all a low wall shear stress area is the preferred location for plaque formation, which later develops into atherosclerosis. Furthermore, plaque formation induces vessel blockage, which in turn leads to an even larger area of low wall shear. As a result, a small area of plaque accumulation may develop into stenosis, and then atherosclerosis, and finally clogs the flow passage. When this occurs in the coronary artery or its neighboring blood vessels, a heart attack may follow. When this blockage occurs in the common carotid, it may result in a stroke – although carotid atherosclerosis is not the only cause of a stroke.

The other extreme of geometry-related hemodynamic conditions are aneurysms. An aneurysm is sudden enlargement, or bulge, in a blood vessel, and its cause is not yet fully understood. This condition in vessel cells is widely believed to be related to the high wall stress and poor heat dissipation associated with the local flow fields [11]. Flow separation is also highly correlated with the establishment of aneurysms [12]. Usually, flow passages with large curvature or with branching, e.g., the abdominal aorta [13] and the common carotid bifurcation are prime target areas for aneurysm development. A rupture in a brain aneurysm is another major cause of stroke - a severe life-threatening condition.

In addition to examining wall shear, it is also important to study the shear stresses in the interior of the flow field. Although for most part of the circulatory system, the flow is always laminar, there are occasions where localized turbulent regions may exist. Examples of such occurrences are in rear regions of an artificial heart valve. Since the fluid stress in a turbulent flows are almost two orders higher in magnitude higher than that in laminar flows, the possibility
of red blood cell membrane rupture is greatly increased. Release of the erythrocyte contents due
to this damage can result in a variety of serious effects such as anemia, as well as toxic effect of
free hemoglobin.

As mentioned above, flow field in a branching vessel also warrants special attention. This flow field exhibits important features closely related to the origin of some pathological conditions. In his two-dimensional simulation and experimental measurement of flow in the carotid artery bifurcation, Rindt [14] demonstrated the existence of a large separation region with reverse flow velocity, both for steady flow and unsteady flow conditions. Rindt [15] further performed a three-dimensional version of study, and found that the induced secondary flows simulated are comparable to the corresponding LDA measurements. In addition to wall shear stress concerns, there are other reasons that attract an investigator’s interest. As by-pass surgery has become a well-established procedure for treating vessel disease due to atherosclerosis, it has been noted that the red blood cells in daughter branches may differ from those in the parent vessels. In particular, there is a tendency for red blood cell to concentrate toward the core of the vessel. This nonuniform distribution of red blood cells causes a ‘separation surface’ and plays a critical role in the red blood cell count in the daughter branches, and thus the usefulness of this procedure. A “separation surface” is an imaginary surface in the main parent vessel; on one side of it, the fluid is diverted into the daughter branch, and on the other side the, fluid remains in the parent vessel. Enden [16] performed a numerical simulation of the flow field, which compared favorably with the previously obtained experimental data for a wide range of Reynolds numbers and daughter-to-parent vessel diameter ratios. Hence, these results may be used for prediction of red blood cell concentration downstream of daughter branches. The study by Carr [17] pursued the same issues, techniques.
Arterial anastomosis is yet another surgical therapy that bypasses the blood flow over a diseased area. An end-to-side graft is frequently employed for this purpose. However, the technique is often associate with long-term complications. Intimal hyperplasia is often observed in the graft area a few months after the procedure. Although many theories have been postulated, it is believed that a “common denominator in hemodynamics condition” prevails. The experimental work by White [18] illustrates this point, and their data will be used as reference for validation of the present numerical algorithm. The details will be described in Section 5.2

Hoppensteadt [19] demonstrated a simplified model for simulation of a whole-body circulation system. A similar but more sophisticated model has also been used by Zacek [1]. These are one-dimensional simplified models, but can provide an overall view of the function of the whole system. For example, the sensitivity of isovolumic pressure to aorta compliance may be easily determined without extensive computational demands. Use of this model in evaluating the system response in a diseased condition, e.g., aortic valve insufficiency, has been suggested [19].

The one-dimensional approximation concept also been employed by Isaaz [20] in determination of the transmitral pressure-flow relation. In fact, there are cases in which one-dimensional data is the only data available clinically. The flow velocity measurement between left atrium and left ventricle using echocardiography is such an example. Isaaz[20] suggested that the measured time history of transmitral velocity has been used as a diagnostic index for disease assessment. Different diseases are associated with different velocity time history; thus, this velocity measure may help physicians arrive at a better diagnosis of the root cause of a symptom.
However, the significance of different transmitral velocity curves and their relation to ventricular diastolic properties is not clear yet. Lemmon [21] investigated the effect of left ventricular stiffness and relaxation timing on the transmitral velocity wave form, using the immersed boundary method. The current research aims to answer the following questions:

Is there a local flow structure in the left atrium that deteriorates the left ventricle filling?

It is believed that, with these questions answered, understanding of the diastolic function of the heart will be clearer. Furthermore, the medical community can utilize the findings of the present study to design and evaluate new therapies for the related disease.

1.3 Computational Challenges

Applications of CFD techniques to real-world problems, involves overcoming many challenges. Among them, the complexity of the geometry of the physical domain is the most significant. Even after considerable simplification, the physical domain under consideration may still be far more complicated than the familiar rectangular, circular or spherical domains. There are many ways to address the difficulty; two commonly used approaches will be outlined here. The first family of schemes employs multiple-blocks to represent the physical flow domain. In this family, a complicated domain is split into several simpler sub-domains, either overlaid or patched. The flow field is solved separately in each sub-domain. The boundary values for one sub-domain are obtained by interpolation or direct transfer from the field values of the neighboring sub-domains. An iteration procedure is performed to update these boundary values and hence, the interior flow field of each sub-domain, until convergence. The work of Wu [22] is a typical application of the overlaid structured mesh scheme, also referred to as the Chimera scheme, to determine the flow field around multi-element airfoils.
The second family of schemes for dealing with complex domains uses an unstructured mesh system. Schemes in this family are most often used in conjunction with numerical schemes such as the finite element method or the finite volume method in which a structured mesh system is not required. The commercial software package STAR-CD is a typical example of the finite-volume method utilizing unstructured mesh, and has been successfully utilized in industry worldwide.

The present research proposes a new scheme— the multi-box scheme for handling complicated physical domains. It is an extension of the traditional finite-difference scheme. The traditional finite-difference method utilizes mappings which transform the physical domain onto a simpler computational domain, and then performs numerical computations on this transformed domain. This transformed computational domain is usually of rectangular shape in 2D, or of box shape in 3D. A tremendous amount of effort has been directed towards developing schemes for the determination of such mappings and, in fact, these studies form a new branch of applied mathematics called ‘numerical grid generation’. When dealing with certain complex physical domains, this mapping may not be easily achieved. It is proposed in the present research to relax the requirement of “one simple rectangular computational domain in 2D” or “one simple box computational domain in 3D”. In other words, a multi-box shaped computational domain is used in this study. The ultimate goal is still the same, namely; to find a mapping which transforms the complex physical domain onto a simpler computational domain. In the present study, although the computational domain may not be as simple as that in traditional finite-difference approaches, the numerical solution of the flow equations on this transformed domain is still convenient, with suitable modification of the flow field solver. Hence, there is a shift in between
the procedure for obtaining the mapping and the procedure for obtaining the flow solution. The
details of this procedure are to be given in Chapter 3.

The second major challenge when coping with real-world problems is that of accuracy.
Ideally, the higher the spatial resolution, the more accurate the result is. However, in an
application of a numerical scheme, a grid system with high resolution throughout the physical
domain may not be necessary. For example, for a viscous flow over a flat plate, it is necessary to
have higher resolution only near the boundary layer where the flow gradients are high, and for
the areas far away from the flat plate, it is still feasible to use a coarser grid because not much
flow activity is taking place in this area. Hence, for optimal usage of computational resources, a
flow-field dependent-gridding scheme is a plausible avenue. However, the cost associated with
the procedure for re-gridding must be balanced by the saving resulted from reducing the total
number of computational grid points while maintain the same spatial accuracy level. Thus a
trade-off must be exercised between the expense for re-gridding and the benefit from reducing
the number of grid points. In the present research, a parabolic partial differential equation is
used for achieving the re-gridding. This approach provides a very efficient way for grid re-
generation while tracking the temporal evolving flow field. The details are given in section 3.4.

1.4 Objectives and Plan

Every computational simulation starts from an initial grid system. The governing
equations are then described at the grid points in this system, and boundary conditions set on the
point on the physical boundary. Once the physical parameters have been set, one can then
launch a numerical simulation and obtain the result. In the present research, an initial grid
system is generated based on the idea described by Spekreijse [23] and later modified by the
present author for incorporating the proposed multi-box scheme. Section 3.1 describes the details of the initial grid generation procedure.

For obtaining the flow solution, the method of artificial compressibility as described by Rogers and Kwak [25] is used. The research software, INS3D, utilizing this method is employed. This is a powerful software tool that can perform steady and unsteady flow simulations on a structured grid system. Various turbulence models are available, although only laminar flow is of concern in the present study of biological flows which are generally unsteady, but are characterized by low Re. Chapter 4 will provide the details of the numerical scheme and its implementation.

Two physiological cases are examined, using the computational technique to be outlined in chapters 1 through 4. The first case simulated is the flow in the vicinity of an ilio-femoral arterial graft. For comparison purposes, the results of White [18] and Taylor [26] are used as a reference baseline. This step facilitates validation of the overall procedures – grid generation as well as flow field calculation and, hence paves the way for exploration of the more complex flow field, i.e., flow field in the left atrium.

The flow field in the left atrium model is to be studied next. As mentioned in section 1.1, the left atrium is responsible of the diastolic filling of the cardiac cycle. In the present study, a steady-state flow field is assumed. This assumption corresponds to a long –term characteristic of the flow field. The geometry-related flow phenomenon is the goal of this study. The details will be given in chapter 6.
CHAPTER 2

MATHEMATICAL DESCRIPTION OF THE FLOW PROBLEM

2.1 Governing Equations in Vector Form

The present research deals with the human cardiovascular system. Therefore, the human blood is the major fluid of concern. The blood is incompressible in the normal operation of circulatory system. In addition, even though the blood exhibits some non-Newtonian fluid characteristics, it has been shown [28] that for flow in major arteries, the physics of blood can be well approximated by that of Newtonian fluid. In addition, for the velocity range considered, it is reasonable to assume stable laminar flow behavior.

Hence, assumption was made of an incompressible Newtonian fluid with constant viscosity throughout the research. The mathematical model describing the flows of such fluid is given by the well-known Navier-Stokes equation

\[ \rho \left( \frac{\partial \vec{v}}{\partial t} + \vec{v} \cdot \nabla \vec{v} \right) = -\nabla p + \mu \nabla^2 \vec{v} \]

(2.1)

which is nothing but a statement of the Newton’s second law of motion for control volume. The left hand side represents the change of momentum of the fluid particle in the control volume. The change was composed of two parts. The first part, \( \rho \frac{\partial \vec{v}}{\partial t} \), is termed “local acceleration” and the second part, \( \rho \vec{v} \cdot \nabla \vec{v} = \vec{v} \cdot \nabla (\rho \vec{v}) \), is called “convective acceleration”. The right hand side of equation (2.1) is the external force acting on the fluid particles in the control volume. The first force considered is the pressure force and is represented by the term \( -\nabla p \). The second term is the force due to viscous effect. As stated previously, this force is modeled by Newtonian approximation and hence by given in the form \( \mu \nabla^2 \vec{v} \).
In equation (2.1), there is no gravity force considered, since the existence of such term in the equation merely offset the solution by a constant value. Therefore, as long as the pressure and velocity is concerned, equation (2.1) is sufficient to describe the physics inherent.

In addition to the law of motion, the conservation of mass needs to be assured. In the language of mathematics, this can stated as
\[ \nabla \cdot \vec{v} = 0 \]  
(2.2)
for incompressible fluid. Equation (2.2) is known as the continuity equation. One should note that equation (2.2) holds for steady and unsteady flows, although this equation did not consist time-derivative term as the momentum equation does. In other word, for any time instant, the time-dependent velocity vector must obey the “stationary” continuity equation.

Equation (2.1) is a vector equation that can be expanded into three scalar equations while equation (2.2) itself is a scalar equation. Thus an equations set of four scalar governing differential equation is available. As far as unknown variables are concerned, the velocity vector has three scalar components, and the pressure is yet another scalar quantity to be determined. Other quantities appeared in equation (2.1) and (2.2), \( \rho \) and \( \mu \), are pre-assigned constant as described earlier in the present section. Hence, a total of four unknown in four equations is to be solved numerically.

Equation (2.1) can be cast into so-called “conservative form” as follows:
\[
\rho \left( \frac{\partial \vec{v}}{\partial t} + \nabla \cdot (\vec{v} \vec{v}) \right) = -\nabla p + \mu \nabla^2 \vec{v} 
\]  
(2.3)
Although equation (2.3) and equation (2.1) are equivalent mathematically, there is a benefit using equation (2.3) over equation (2.1) when using numerical scheme. This benefit arises mainly through the exact cancellation of divergence term, also known as “telescoping effect”, so
that global conservation over a finite control volume is preserved. Consequently, in this research, only conservative form, equation (2.3) is employed.

Equation (2.3) is a nonlinear, elliptical-parabolic equation. The nonlinearity comes from the convective acceleration \( \mathbf{v} \cdot \nabla (\rho \mathbf{v}) = \nabla \cdot (\rho \mathbf{v} \mathbf{v}) \). This term causes one of the major difficulties in obtaining the solution. As consequence, many numerical schemes were developed aiming to resolve this difficulty. Central differencing, upwind differencing, and flux splitting bias differencing scheme, for example, are some typical effort for this goal. The elliptical-parabolic characteristics of equation (2.3) arise from the coefficient of second order derivatives. For example, in x-t plane, the coefficient of \( v_{xx} \) is \( \mu \), coefficient of \( v_u \) is 0 and that of \( v_{xu} \) is 0. So

\[
B^2 - 4AC = 0^2 - 4 \times 0 \times \mu = 0
\]

Hence, equation (2.3) is parabolic in x-t plane. The same argument applies to y-t plane and z-t plane. Likewise, the equation is elliptic in x-y, y-z and x-z plane, as can be verified easily.

Yet another major difficulty in solving the Navier-Stoke system of equation is that this system is not completely coupled. While equation (2.3) contains the unknown u, v, w (the x-, y- and z- components of velocity, respectively) and p, equation (2.2) only contains u, v, w but not p. The absence of p variable in continuity equation prohibits the linking between velocity and pressure in this equation and makes it impossible to use a simple explicit scheme that avoids solving system of algebraic equation. This difficulty can be relieved by many schemes, among which is the method of artificial compressibility that will be used in the present study and will be discuss in detail in chapter 4.

In order to perform numerical solution using the NASA research code INS3D, it is necessary to perform non-dimensionalization on the governing equation. Let \( L_R \) and \( U_R \) be
reference length and velocity, respectively. Following standard procedure of dimensional analysis, the following equation is obtained.

\[ \nabla^* \cdot \vec{v}^* = 0 \quad (2.4) \]

\[ \frac{\partial \vec{v}^*}{\partial t} + \nabla^* \cdot (\vec{v}^* \vec{v}^*) = -\nabla^* p^* + \frac{1}{\text{Re}} \nabla^{**} \vec{v}^* \quad (2.5) \]

where

\[ \nabla^* = i \frac{\partial}{\partial x^*} + j \frac{\partial}{\partial y^*} + k \frac{\partial}{\partial z^*} \quad \text{and} \quad x^* = \frac{x}{L_R}, \quad y^* = \frac{y}{L_R}, \quad z^* = \frac{z}{L_R} \]

\[ \vec{v}^* = \frac{\vec{v}}{U_R} \]

\[ p^* = \frac{p}{\rho U_R^2} \]

\[ t^* = \frac{t}{(\frac{U_R}{l_R})} \]

\[ \text{Re} = \frac{\rho U_R L_R}{\mu} \]

As mentioned earlier, the present research concerns with blood flows in human body, so the Reynolds number (Re) falls in the range from 0 to 1000, according to [12]. In later sections, the * in dimensionless equation (2.4) and (2.5) are dropped because only dimensionless equation is of interest.

### 2.2 The Boundary-Fitted Generalized Coordinate System

When finite difference scheme is employed in the numerical solution of partial differential equations, it is necessary to make use of a rectangular grid system for the discretization of solution domain. If the domain is of rectangular shape, it should be easy to create a uniform rectangular grid system. However, most of the “real life” application of
computational fluid dynamics involves a more general solution domain. When such domain is
discretized by rectangular grid system, some “incomplete” cells will be created, i.e., cells in
which some of the defining nodes are outside of the solution domain whereas the other nodes,
inside the domain. (Figure 3)

![Incomplete Cells](image)

**Figure 3. Incomplete Cells.**

Note that these “incomplete cells” are all appeared near the boundary of solution domain.
The existence of incomplete cells makes the imposition of boundary conditions a difficult task,
especially when boundary conditions of Neumann type are considered. Although the use of
interpolation schemes may resolve this implementation issue, the solution accuracy still suffered
from these lower order approximations.

Yet another dilemma associated with the “incomplete cells” is the numerical stability.
Whenever explicit schemes is used to solve a time-dependent equation, it is well known that the
temporal step size ($\Delta t$) and spatial step size ($\Delta x$) must meet an inequality of the form

$$\alpha \frac{\Delta t}{\Delta x^2} \leq c$$

in order for the scheme to be numerically stable. Examples of such inequality is seen in the
stability requirement for the FTCS (Forward Time, Central Space) approximation of the
parabolic equation

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}$$
i.e. \[ \alpha \frac{\Delta t}{\Delta x^2} \leq \frac{1}{2} \]

In other words, for the solution to be stable for the whole domain, \( \Delta t \) must be chosen such that equation (2.6) is satisfied, even for the minimum \( \Delta x \). It follows that

\[ \Delta t_{\text{allowed}} \leq \left( \frac{c}{\alpha} \right) \Delta x_{\min}^2 \]  \hspace{1cm} (2.7)

As can be seen from equation (2.7), the smaller \( \Delta x_{\min} \) is, the smaller \( \Delta t_{\text{allowed}} \) must be. In cases that involve incomplete cells, \( \Delta x_{\min} \) can be very small. Consequently, the allowed \( \Delta t \) is forced to take tiny value in order to fulfill equation (2.6). Therefore, with incomplete cells, the stability requirement renders an explicit scheme very inefficient.

The above-mentioned reasons motivate schemes utilizing boundary fitted coordinate systems. With this technique, an imaginary space, the computational space, is introduced. A mapping between the physical space and this computational space is defined such that the general nonrectangular domain in physical space is mapped onto a rectangular domain in computational space. The term 'boundary fitted coordinate system' reflects the fact that the boundary of physical domain corresponds to the boundary of computational domain that aligned to the computational coordinate system. When rectangular discretization is performed in the computational domain, the corresponding physical domain is also been discretized in such a way that no incomplete cell is generated. Furthermore, it is also possible to setup the mapping such that the computational coordinate lines are orthogonal at the boundaries. In this manner, the imposition of the boundary condition, especially those of Neumann type and Robin type, become much easier because no interpolation is necessary. The governing equations, with physical spatial coordinates as independent variables, also need to be transformed onto computational domain, with computational spatial coordinate as independent variables. These transformed
equations contain certain coefficients that describes the mapping relation. These coefficients, termed the transformation metrics, are in the form of derivatives, \( \frac{\partial \xi}{\partial x} \), where \( \xi \) is the computational coordinate and \( x \) is the physical coordinate. When evaluate these coefficients numerically, it should be careful because this could be another source of error in the numerical solutions of the governing equations.

The boundary fitted coordinate system is very useful when dealing with moving boundary problem, as in the simulation of heart chamber flows that the present research is focused on. In this case, the mapping between computational space and physical space is a time dependent function, and the grid system in physical space needs to be reconstructed for each time instant. Despite an additional metrics terms in the governing equation, the implementation of boundary condition is as simple as for the stationary boundary cases. In contrast, the so called "immersed boundary method" [29,30], which essentially a time-dependent version of the method utilizing Cartesian grid (with incomplete cells) as discussed above, required a time-dependent interpolation for imposition of boundary condition. Thus the use of time dependent boundary fitted coordinate system simplifies the implementation of boundary condition at an expense that grid must be regenerated at each time instant.

So far, the discussion on generalized coordinate system has been focused on the boundary conforming issues. Nevertheless, there are more that the generalized coordinate system can bring into. Under the framework of boundary fitted coordinate system, there are still lots of freedom of choice what the mapping can be. One possibility is to cluster interior grid inside the domain to meet certain desired feature. An obvious example is the clustering of grid point in regions with high solution gradient, or regions deserve higher spatial resolution [6]. This concept of solution adaptive grid generation can be applied to steady flow problem as well as unsteady flow
problem. For steady flow problem, successive improvement on the grid as well as flow solution can be made in a sequence and the result is a more efficient deployment of grid points. For unsteady flow problem, the grid-clustering region should develop in a synchronized manner with the evolving critical flow regimes (regions with high solution gradient) for improved spatial resolution. Note also that these schemes can apply to both stationary grid and moving grid problem.

However, cares need to be taken when using the boundary fitted coordinate system. First, as point out by Fletcher [31], the smoothness of the boundary fitted coordinate system plays a vital role in the accuracy and efficiency of the numerical solution, especially if second order equations are considered. Fletcher (section 12.4) provided an example demonstrating such effect. As a rule of thumb, the variation of dimensions between neighbor cells should be of order 1, i.e. \( \Delta x_2 = [1 + O(\Delta x)] \Delta x_1 \). Secondly, when numerically evaluating metric terms in the governing equations, it is generally recommended that the same discretizing formula been used as those used in the discretization of derivatives of dependent variables [32]. This leads to a cancellation of a major portion of truncation error and thus achieves a smaller solution error. As was demonstrated by Fletcher [31], violation of this rule may result a loss of conservative property in the discretized version of the governing equation.

In the next section, the transformed governing equation in computational coordinate system is presented. Numerical schemes for solving these equations will be discussed in Chapter 4.
2.3 Governing Equation in Generalized Coordinate System

Let the coordinate system in physical space be designated by \((x, y, z, t)\) whereas those in the computational space being \((\xi, \eta, \zeta, \tau)\). A mapping is established between these two spaces, i.e.

\[
\begin{align*}
  x &= x(\xi, \eta, \zeta, \tau) \\
  y &= y(\xi, \eta, \zeta, \tau) \\
  z &= z(\xi, \eta, \zeta, \tau) \\
  t &= t
\end{align*}
\]

or, conversely

\[
\begin{align*}
  \xi &= \xi(x, y, z, t) \\
  \eta &= \eta(x, y, z, t) \\
  \zeta &= \zeta(x, y, z, t) \\
  \tau &= \tau
\end{align*}
\]

Obviously, it is required that this mapping been one-to-one, that is, a coordinate point in physical space corresponds to, and only to, a point in computational space. Mathematically, this criterion may be written as

\[ J \neq 0 \]

where \(J\) is the determinant of the transformation matrix

\[
\begin{vmatrix}
  \xi_x & \xi_y & \xi_z \\
  \eta_x & \eta_y & \eta_z \\
  \zeta_x & \zeta_y & \zeta_z
\end{vmatrix}
\] and referred to as the Jacobian of the transformation. Furthermore, without loss of generality in future discussions, it is permissible to assume that \(J > 0\), thus, a right-handed system in computational space will be mapped onto a right-handed system in physical space.

The continuity equation, (2.4), can be cast in the form

\[
\frac{\partial}{\partial \xi} \left( \frac{U}{J} \right) + \frac{\partial}{\partial \eta} \left( \frac{V}{J} \right) + \frac{\partial}{\partial \zeta} \left( \frac{W}{J} \right) = 0 \quad (2.8)
\]
where

\[ U = \xi_x u + \xi_y v + \xi_z w \]
\[ V = \eta_x u + \eta_y v + \eta_z w \]
\[ W = \zeta_x u + \zeta_y v + \zeta_z w \]

are the contravariant components of the velocity vector and \( J \) is the Jacobian of the transformation. Notice that equation (2.8) is structurally similar to its Cartesian version in physical domain. Therefore, the issue of lacking of pressure linking in this equation in the physical domain persists in the computational domain.

The momentum equation, (2.5), takes the following form in computational space:

\[
\frac{\partial \hat{u}}{\partial \tau} = - \frac{\partial}{\partial \xi} (\hat{e} - \hat{e}_v) - \frac{\partial}{\partial \eta} (\hat{f} - \hat{f}_v) - \frac{\partial}{\partial \zeta} (\hat{g} - \hat{g}_v) \tag{2.9}
\]

In equation (2.9),

\[
\hat{u} = \frac{1}{J} \begin{pmatrix} u \\ v \\ w \end{pmatrix}, \quad \hat{e} = \frac{1}{J} \begin{pmatrix} \xi_x p + u U + \xi_x u \\ \xi_y p + v U + \xi_y v \\ \xi_z p + w U + \xi_z w \end{pmatrix}, \quad \hat{f} = \frac{1}{J} \begin{pmatrix} \eta_x p + u V + \eta_x u \\ \eta_y p + v V + \eta_y v \\ \eta_z p + w V + \eta_z w \end{pmatrix}, \quad \hat{g} = \frac{1}{J} \begin{pmatrix} \zeta_x p + u W + \zeta_x u \\ \zeta_y p + v W + \zeta_y v \\ \zeta_z p + w W + \zeta_z w \end{pmatrix}
\]

and

\[
\hat{e}_v = \frac{1}{\text{Re} J} \begin{pmatrix} (\nabla \xi \cdot \nabla \xi) u_\xi \\ (\nabla \xi \cdot \nabla \xi) v_\xi \\ (\nabla \xi \cdot \nabla \xi) w_\xi \end{pmatrix}, \quad \hat{f}_v = \frac{1}{\text{Re} J} \begin{pmatrix} (\nabla \eta \cdot \nabla \eta) u_\eta \\ (\nabla \eta \cdot \nabla \eta) v_\eta \\ (\nabla \eta \cdot \nabla \eta) w_\eta \end{pmatrix}, \quad \hat{g}_v = \frac{1}{\text{Re} J} \begin{pmatrix} (\nabla \zeta \cdot \nabla \zeta) u_\zeta \\ (\nabla \zeta \cdot \nabla \zeta) v_\zeta \\ (\nabla \zeta \cdot \nabla \zeta) w_\zeta \end{pmatrix}
\]
\( \hat{e}_v, \hat{f}_v, \hat{g}_v \) are called the viscous flux. As mentioned in section 2.1, constant viscosity is assumed in the present research. Therefore, there is no such term as viscosity gradient \( (\nabla \mu) \), as well as Reynolds number gradient terms appeared in the expressions for viscous flux.

It should be pointed out that, equation (2.9) contains both Cartesian component as well as contravariant component of the velocity vector. While in generalized coordinate system with the transformation metric predetermined, one may view the Cartesian component as the dependent variables and numerically solve for these unknowns, as the software INS3D does. On the other hand, it is also possible to utilize the contravariant or covariant component of velocity vector as the unknown variables in the equation and numerically solve for it. [33 - 35] provided such examples.

2.4 Boundary Conditions and Initial Conditions

For steady-state flow simulation, only boundary conditions are necessary for a well-posed problem. However, due to iterative nature of the numerical scheme (Chapter 4) for the solution of the governing equation, it is also necessary to set an initial guess value to start up the iteration process. A properly chosen initial guess is important. Although these initial guess eventually get 'washed out' and hence has no effect on the final solution, it greatly affect the rate of convergence. A poorly chosen initial guess may even diverge the iteration and no result can be obtained.

As a rule of thumb, the initial guesses need to satisfy the continuity equation, i.e., the divergence of velocity field should be zero. As the numerical process tend to reduce the magnitude of divergence of velocity field, it would be very difficult for the scheme to render a finite-valued divergence at initial time step to a zero-valued divergence at the immediate next time step. Such abrupt changes may results in the iteration procedure diverge.
Following the same thought of smooth iterative procedure, one should expect a larger number of iteration if the initial guess differs from the expected solution more. Therefore, if a reasonably close approximation is available, the numerical solution process is likely to be more efficient. A procedure called ‘incremental loading’ [36] may serve for this purpose. For example, to simulation a flow field with Reynolds number 1000, one could start with a initial flow field with zero velocity everywhere (hence fulfill the continuity equation) and perform a simulation with very low Reynolds number, say 200. Once obtained a numerical solution for Re=200, use it as initial guess value for yet another simulation with Re=400. Again, once obtained a numerical solution for Re=400, use it as initial guess value for yet another simulation with Re=600, and so forth until the case Re=1000 is simulated. Parameters other than Reynolds number may also serve for this purpose, as long as by changing the value of these parameters, the problem may get simplified.

For time-accurate simulation with temporal derivative terms in the governing equation, in addition to the boundary condition, the specification of initial condition is required for a well-posed problem. Unlike initial guess for steady-state problem, these initial conditions are physically meaningful and has certain impact on the transient solution been sought. Just because these initial conditions are physically meaningful, they must satisfy the continuity equation. The specification of such condition depends solely on the physical problem been studied. In the numerical solution using INS3D, there is a series of sub-iteration (pseudo-time marching) for each physical time marching steps and each sub-iteration series needs a starting value. Again, the concept of ‘incremental loading’ applies in this context where the converged solution for time step n is used as the initial guess for time step n+1.
Special cases associated with the time-marching problem in which the specification of initial condition is less important are when the governing equation and/or boundary conditions contain periodic time-varying terms. This situation is similar to that of steady state problem. Although a time-dependent problem, in the long run, the physical system is dominated by the governing equations and boundary conditions only but not the initial condition. This is the case for problems of flow simulation in human cardiovascular system. In the simulation of flow field in graft geometry (section 5.3), the flow velocity at inflow boundary is specified as a prescribed periodic function whereas in the simulation of flow in the left atrium (Chapter 6), the inflow pressure boundary condition is also prescribed as periodic.

On solid wall, one may utilize the criteria of no-slip/no penetration of velocity vectors as boundary conditions, i.e., there may be no relative motion between the fluid and the wall on the wall surface. No independent pressure boundary condition may be specified. For non-staggered grid system such as those employed in the solver INS3D, the boundary condition for pressure may be obtained by substituting the known velocity vector on the boundary into the Navier-Stokes equations and solve for the pressure term. If the above procedure is carried out, one may obtain an expression like

\[ \nabla p = \text{RHS} \]

where RHS is an expression involving wall velocity and their derivatives in both directions. However, for moderate to large Reynolds number, the value of RHS is usually small. So in INS3D, the pressure boundary condition is approximated as \( \frac{\partial p}{\partial n} = 0 \), that is, zero pressure gradient normal to the surface. On the other hand, if staggered grid system is used, one may choose to setup the grid so that the boundary contains only those points with which velocity is
specified. Since no 'pressure point' exists on the boundary, so no pressure boundary condition is required. This procedure is given in details in Peyret and Taylor [37], and Fletcher [38].

For inflow and outflow boundaries, it is appropriate to specify all but one of the dependent variables. Thus, the three Cartesian component of velocity vector may be specified on the inlet and left the pressure terms determined from the governing equation. It is also feasible to prescribed the pressure and velocity direction at inlet, leaving velocity magnitude determined from the governing equation.

If the physical domain of interest is properly selected, the viscous term in the governing equation at outlet may be negligibly small. The flow field may then be viewed as locally inviscid. In this case, instead of specifying three of the unknown variables as boundary condition, one is sufficient to close the problem. Usually the outflow pressure is the pre-assigned, and then the velocity components are determined from the inviscid version of governing equation, i.e., Euler equation.
CHAPTER 3

NUMERICAL GRID GENERATION

This chapter is dedicated to the grid generation phase of the simulation work. The scheme employed is based on a similar approach proposed by Spekreijse [23]. The present author has made several extensions of that scheme to accommodate the challenge presented in the target problem – the heart chamber. Areas extended include the treatment of a time-varying domain, multi-box scheme, and modified Thomas algorithm. In the following, Section 3.1 provides a detailed description of the derivation of the grid-transport equations, which will be used throughout this research. Section 3.2 and 3.3 discuss the extension beyond Spekreijse's work for handling the current problem. The Numerical solution scheme are also adjusted accordingly.

3.1 Grid Generation Equations

This section consists of two parts. The first part deals with stationary grid generation, whereas the second part is for time-dependent grid generation. Stationary grids may be used as the initial grids for time-varying grid problems such as moving boundary or flow-adaptive grid simulations, or for use in fixed grid problems with steady and/or unsteady flows.

3.1.1 Grid Generation Equations for Initial and Stationary Grid Systems

Let domain $D$ be the physical domain of interest on which a proper grid system is required. As shown in Figure 6, $D$ is bounded by the curve $E1, E2, E3$ and $E4$, and the physical coordinate is denoted by a real number pair $(x, y)$. The concept of grid generation is to devise a mapping that maps each and every point from the domain $D$, one-to-one, onto a computational space $C$, and vice versa. The domain $C$ is defined as a unit square in the computational space with computational coordinates $(\xi, \eta)$. The boundary point distribution is prescribed;
mathematically, this means that the mapping \( X: \partial C \rightarrow \partial D \) is given. Assume that the edge 1 on physical domain corresponds to the edge  (1) on computational domain and thus has \( \xi \)-coordinates of 0.0. Similarly, 2 is mapped onto \( E_2 \), and thus \( \xi = 1 \). Likewise, 3 and 4 maps to \( E_3 \) and \( E_4 \), and thus \( \eta = 0.0 \) and 1.0, respectively.

The grid generation problem can be formulated mathematically as a boundary value problem to find the mapping \( X: C \rightarrow D \) possessing the following properties:

- satisfy the boundary condition prescribed, i.e., \( X: \partial C \rightarrow \partial D \)
- be differentiable, and hence one-to-one
- interior grid points are 'good' reflection of the boundary grid point distribution.

Spekreijse [23] split the mapping \( X \) to two parts: an algebraic mapping \( P \) from the computational domain \( C \) onto a parameter space \( P \), then another mapping \( S \) from the parameter space \( P \) to the physical domain \( D \), based on the solution of Laplace equation. The mapping \( S \) is described first (refer to Figure 6).

The parameter space \( P \) is yet another unit square with coordinates \((s, t)\). The values of \( s \) and \( t \) for points on the boundary of domain \( D \) are defined by the normalized arc length along the boundary curve. Thus, the mapping \( S^{-1}: \partial D \rightarrow \partial P \) is defined to be

\[
\begin{align*}
&\text{on } E_1 \quad s = \text{normalized arclength along } E_1 \text{ and } t = 0 \\
&\text{on } E_2 \quad s = \text{normalized arclength along } E_2 \text{ and } t = 1 \\
&\text{on } E_3 \quad s = 0 \text{ and } t = \text{normalized arclength along } E_3 \\
&\text{on } E_4 \quad s = 1 \text{ and } t = \text{normalized arclength along } E_3
\end{align*}
\]  

(3.1)

For interior points, the mapping \( S^{-1}: D \rightarrow P \) is described by the system of Laplace equations
\begin{align}
\Delta s &= \frac{\partial^2 s}{\partial x^2} + \frac{\partial^2 s}{\partial y^2} = 0 \\
\Delta t &= \frac{\partial^2 t}{\partial x^2} + \frac{\partial^2 t}{\partial y^2} = 0
\end{align} 

(3.2)

With Eq. (3.2) being the governing equations and Eq. (3.1) being the boundary conditions, a well-posed boundary value problem (BVP) has been formulated for the unknown variable \( s \) and \( t \). Hence, the mapping \( S^{-1}: D \rightarrow P \) is defined. In view of the maximum principle for the solutions of Laplace's equation, the mapping \( S^{-1}: D \rightarrow P \) is differentiable and one-to-one, so is its inverse. That is, \( S: P \rightarrow D \) exists.

It should be noted that, when constructing the mapping \( S \), no boundary point distribution is involved. In light of this, it is clear that this mapping \( S \) depends solely on the geometry of the domain \( D \), and, thus may be considered as a property of the geometry of domain \( D \). In contrast, the mapping \( P \) maps a unit square in the computational space onto another unit square in the parameter space. In this way, the shape of the domain \( D \) is not contributing to the construction of the mapping \( P \). The single purpose for utilizing this mapping is to 'propagate' the boundary point distribution from the boundary to the interior of \( P \).

From the above discussion, \( X: \partial C \rightarrow \partial D \) is given and \( S^{-1}: \partial D \rightarrow \partial P \) is prescribed, so the mapping \( P: \partial C \rightarrow \partial P \) can be defined accordingly. In other words, the values of \( s \) and \( t \) are specified for points on the boundary \( 1, 2, 3 \) and \( 4 \) of the computational domain, as follows

\[
\begin{cases}
  s(0, \eta) = 0, & t(0, \eta) = t_{E_1}(\eta) \\
  s(1, \eta) = 0, & t(1, \eta) = t_{E_2}(\eta) \\
  s(\xi_0) = s_{E_3}(\xi), & t(\xi_0, 0) = 0 \\
  s(\xi_1) = s_{E_4}(\xi), & t(\xi_1, 1) = 0
\end{cases}
\]

(3.3)
Note also that $t_{E1}$ and $t_{E2}$ are monotonically increasing functions of $\eta$, whereas $s_{E3}$ and $s_{E4}$ are monotonically increasing functions of $\xi$.

It is then necessary to set up a differentiable, one-to-one mapping for interior points. For this purpose, the so called "algebraic straight line transformation" is employed. This mapping is spelled out as

\[
\begin{align*}
  s &= s_{E3}(\xi)(1 - t) + s_{E4}(\xi)t \\
  t &= t_{E1}(\eta)(1 - s) + t_{E2}(\eta)s
\end{align*}
\]  

(3.4)

By evaluating the Jacobian $J = s_\xi t_\eta - s_\eta t_\xi$, it can be shown that $J > 0$, and so it is indeed differentiable and one-to-one. In addition, this mapping maps a coordinate line ($\xi = \text{const.}$ or $\eta = \text{const.}$) in the computational domain to a straight line in parameter space. For example, if we take $\xi$ to be constant and allow $\eta$ to vary, the first equation in Eq. (3.4) reveals that $s$ is a linear function of $t$, so it is a straight line. On the other hand, the second equation indicates that $t$ is a linear function of $s$ if $\eta$ is considered constant. For a given point $(\xi, \eta)$ in computational domain, the corresponding $(s, t)$ point is found by the intersection of two straight lines formed by the first and second part of equation (3.4). Due to the uniqueness of the mapping $P: C \rightarrow P$, the surface $s(\xi, \eta)$ is a monotonically increasing function of $\xi$ and surface of $t(\xi, \eta)$ is a monotonically increasing function of $\eta$. This completes the description of the elliptic mapping $S: P \rightarrow D$ and the algebraic mapping $P: C \rightarrow P$. Combining these two mappings, a new mapping $X: C \rightarrow D$ can be formed accordingly.
To obtain a governing differential equation for the composite grid generating system, consider first the covariant base vector

$$\xi = \xi \frac{\partial}{\partial \xi}, \quad \eta = \eta \frac{\partial}{\partial \eta}$$

and introduce the covariant metric tensor components

$$a_{ij} = (a_i, a_j), \quad i = \{1, 2\}, \quad j = \{1, 2\}.$$  

The contravariant base vector is then defined according to the condition of orthogonality that

$$\left( a^i, a_j \right) = \delta_{ij}, \quad i = \{1, 2\}, \quad j = \{1, 2\}$$

where $\delta_{ij}$ is the Kronecker delta. The contravariant metric tensor components are then defined as

$$a^{ij} = (a^i, a^j), \quad i = \{1, 2\}, \quad j = \{1, 2\}.$$  

It can be verified easily that

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} a^{11} & a^{12} \\ a^{21} & a^{22} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

and
\[ a^1 = a^{11}a_1 + a^{12}a_2 \]
\[ a^2 = a^{21}a_1 + a^{22}a_2 \]

Consider an arbitrary function \( \phi = \phi(\xi, \eta) \). Because \((\xi, \eta)\) and \((x, y)\) have a one-to-one correspondence, functions of \((\xi, \eta)\) are also functions of \((x, y)\). The following relation expresses the relation of derivatives in these two systems.

\[
\Delta \phi = \phi_{xx} + \phi_{yy} = \frac{1}{J} \left\{ \left[ \left( J \phi^{11}_{\xi} + J \phi^{12}_{\eta} \right)_{\xi} \right] + \left[ \left( J \phi^{21}_{\xi} + J \phi^{22}_{\eta} \right)_{\eta} \right] \right\}
\]  (3.5)

where \( J \) is the square root of the determinant \( J^2 \) of the covariant metric tensor, evaluated as

\[ J^2 = a_{11}a_{22} - a_{12}^2. \]

Taking as special case that \( \phi = \xi \) and \( \phi = \eta \), respectively, the following is obtained:

\[
\Delta \xi = \frac{1}{J} \left\{ \left[ \left( J \phi^{11}_{\xi} \right)_{\xi} + \left( J \phi^{21}_{\xi} \right)_{\eta} \right] \right\}
\]  (3.6)

\[
\Delta \eta = \frac{1}{J} \left\{ \left[ \left( J \phi^{12}_{\xi} \right)_{\xi} + \left( J \phi^{22}_{\xi} \right)_{\eta} \right] \right\}
\]

Equation (3.5) can be re-written, together with equation (3.6), as

\[
\Delta \phi = a^{11}\phi_{\xi\xi} + 2a^{12}\phi_{\xi\eta} + a^{22}\phi_{\eta\eta} + \Delta \xi \phi_{\xi} + \Delta \eta \phi_{\eta}
\]  (3.7)

which will serve as a fundamental relation in the following derivation.

Let \( \phi = s \) and \( \phi = t \), respectively, in Eq. (3.7); a relation between \( \Delta s \), \( \Delta t \) and \( \Delta \xi \), \( \Delta \eta \) is then obtained as

\[
\Delta s = a^{11}s_{\xi\xi} + 2a^{12}s_{\xi\eta} + a^{22}s_{\eta\eta} + \Delta \xi s_{\xi} + \Delta \eta s_{\eta}
\]
\[
\Delta t = a^{11}t_{\xi\xi} + 2a^{12}t_{\xi\eta} + a^{22}t_{\eta\eta} + \Delta \xi t_{\xi} + \Delta \eta t_{\eta}
\]

The assertion that \( \Delta s = 0 \) and \( \Delta t = 0 \) allow expressions for \( \Delta \xi \) and \( \Delta \eta \) to be written in terms of \( s_\xi \), \( s_{\xi\xi}, t_\xi, t_{\xi\xi} \), etc.
so $\Delta \xi$ and $\Delta \eta$ may be reduced to

$$\begin{pmatrix} \Delta \xi \\ \Delta \eta \end{pmatrix} = a^{11}P_{11} + 2a^{12}P_{12} + a^{22}P_{22}$$

(3.8)

where

$$P_{11} = -T^{-1}\begin{pmatrix} S_{\xi \xi} \\ t_{\xi \xi} \end{pmatrix}, \quad P_{12} = -T^{-1}\begin{pmatrix} S_{\xi \eta} \\ t_{\xi \eta} \end{pmatrix}, \quad P_{22} = -T^{-1}\begin{pmatrix} S_{\eta \eta} \\ t_{\eta \eta} \end{pmatrix},$$

and

$$T = \begin{pmatrix} S_{\xi} & S_{\eta} \\ t_{\xi} & t_{\eta} \end{pmatrix}.$$ 

Let $\phi = x$ in Eq. (3.7); then one obtain

$$\Delta x = a^{11}x_{\xi \xi} + 2a^{12}x_{\xi \eta} + a^{22}x_{\eta \eta} + \Delta \xi x_{\xi} + \Delta \eta x_{\eta}.$$  

(3.9)

Using the fact that $\Delta x \equiv 0$, and substituting $\Delta \xi$ and $\Delta \eta$ from Eq. (3.8), equation Eq. (3.9) may be simplified to

$$a^{11}x_{\xi \xi} + 2a^{12}x_{\xi \eta} + a^{22}x_{\eta \eta} + \left(a^{11}P_{11} + 2a^{12}P_{12} + a^{22}P_{22}\right)x_{\xi} + \left(a^{11}P_{11}^2 + 2a^{12}P_{12}^2 + a^{22}P_{22}^2\right)x_{\eta} = 0$$

(3.10)

This is the governing equation for the composite grid generation system. However, a more concise version of Eq. (3.10) may be obtained which consists of only covariant metric tensor components rather than the contravariant counterparts. Multiplying by $J^2$ throughout Eq. (3.10) and employing the relation

$$J^2 a^{11} = a_{22} = (x_{\eta}, x_{\eta}), \quad J^2 a^{12} = -a_{12} = -(x_{\xi}, x_{\eta}), \quad J^2 a^{22} = a_{11} = (x_{\xi}, x_{\xi})$$

the final governing equation is obtained as

$$\alpha^{11}x_{\xi \xi} + 2\alpha^{12}x_{\xi \eta} + \alpha^{22}x_{\eta \eta} + \left(\alpha^{11}P_{11}^1 + 2\alpha^{12}P_{12}^1 + \alpha^{22}P_{22}^1\right)x_{\xi} + \left(\alpha^{11}P_{11}^2 + 2\alpha^{12}P_{12}^2 + \alpha^{22}P_{22}^2\right)x_{\eta} = 0$$

(3.11)
where

\[ \alpha^{11} = \left(\xi, \eta\right) \]
\[ \alpha^{12} = -\left(\xi, \eta\right) \]
\[ \alpha^{22} = \left(\xi, \eta\right) \]

In the next sub-section where the time-dependent grid equations are derived, alternative form of (3.11) will be referred. This form is consistent with that of Thompson and is written as

\[
\begin{align*}
\alpha^{11} x_{\xi\xi} + \alpha^{22} x_{\eta\eta} + 2\alpha^{12} x_{\xi\eta} + J^2 (P_{\xi\xi} + Q_{\xi\eta}) &= 0 \\
\alpha^{11} y_{\xi\xi} + \alpha^{22} y_{\eta\eta} + 2\alpha^{12} y_{\xi\eta} + J^2 (P_{\xi\eta} + Q_{\xi\eta}) &= 0
\end{align*}
\]

(3.12)

where

\[
P = \frac{1}{J^2} \left( \alpha^{11} P_{11}^1 + 2\alpha^{12} P_{12}^1 + \alpha^{22} P_{22}^1 \right)
\]
\[
Q = \frac{1}{J^2} \left( \alpha^{11} P_{11}^2 + 2\alpha^{12} P_{12}^2 + \alpha^{22} P_{22}^2 \right)
\]

The solutions of equations (3.11) define the mapping X between the computational domain C and the physical domain D. Because X is composed of an algebraic mapping P and a Laplace mapping S, and both are differentiable and one-to-one, the mapping X itself is then differentiable and one-to-one. Also, the interior grid point distribution in parametric space P is mainly governed by the mapping P, which is a good reflection of the boundary point distribution. Hence it is anticipated that, with another harmonic Laplace mapping S, the interior grid point distribution in D is also a good reflection of its boundary point distribution.

Spekreijse [23] also presented methods to implement orthogonality at the boundaries. Orthogonality at boundaries is frequently used to match coordinate lines from adjacent zones when multi-zone topology is employed. However, with such implementation, the mapping P is no longer one-to-one and, as a result, the generated grid is not guaranteed to be fold-free. Furthermore, when implementing the orthogonality condition on current author's multi-rectangular domain, frequently it is difficult to obtain a monotonically increasing \( s(\xi, \eta) \) and
t(ξ,η). This situation will be elaborated in the next section. With the fact in mind that, for multi-rectangular scheme, no inter-zonal coordinate-line matching is necessary, it seems less important to implement orthogonality at the boundaries. Hence, this is not described in this section. Interested reader may consult Spekreijse [23].

For three-dimensional grid generation, the concepts are similar, but require some modification. Suppose that a mesh system for the physical domain D is desired. As shown in Figure 7, domain D has the physical coordinate denoted by a real triplet (x, y, z) and is bounded by the six faces F1, F2, F3, F4, F5 and F6. Face F1 (hidden) and face F2 (colored as purple) are opposite faces, F3 (colored as cyan) and F4 (hidden) are opposite faces and F5 (hidden) and F6 (colored as green) are opposite faces.

![Figure 7. Physical Domain D and the Bounding Faces.](image)

The aim is to devise a mapping that maps each and every point from the domain D, one-to-one, onto a computational space C, and vice versa. The computational space C is a unit cube, with computational coordinates (ξ, η, ζ). The boundary point distribution is prescribed; mathematically, this implies that the mapping X:∂C → ∂D is given. Now, assume that the following conditions holds on the boundary of domain D.
\[ \xi = 0 \text{ on face } F_1, \quad \xi = 1 \text{ on face } F_2, \]
\[ \eta = 0 \text{ on face } F_3, \quad \eta = 1 \text{ on face } F_4, \]
\[ \zeta = 0 \text{ on face } F_5, \quad \zeta = 1 \text{ on face } F_6 \]

As in two-dimensions, the grid generation problem can be formulated mathematically as a boundary value problem of finding the mapping \( X : C \rightarrow D \) processing the following properties:

- satisfy the boundary condition prescribed, i.e., \( X : \partial C \rightarrow \partial D \)
- be differentiable one-to-one
- interior grid points be a 'good' reflection of the boundary grid point distribution.

The mapping \( X \) again is split into two parts, an algebraic mapping \( P \) and Laplacian mapping \( S \). The algebraic mapping maps the computational space \( C \) onto a parametric space \( P \). The parameter space \( P \) is yet another unit cube, with coordinates \((s, t, u)\). The values of \( s, t, u \) on the edges of the domain \( D \), i.e., the intersection of \( F_1 \) and \( F_3 \), \( F_1 \) and \( F_5 \) … etc., are defined as the normalized arc length along the edges. Require that \( s, t, u \) obeys the following conditions
\( s \equiv 0 \) on face F1, \( s \equiv 1 \) face on F2,
\( t \equiv 0 \) on face F3, \( t \equiv 1 \) face on F4,
\( u \equiv 0 \) on face F5, \( u \equiv 1 \) face on F6

\( s \) is the normalized arc length along the 4 edges connecting F1 and F2.
\( t \) is the normalized arc length along the 4 edges connecting F3 and F4.
\( u \) is the normalized arc length along the 4 edges connecting F5 and F6.

Combining this definition of \((s, t, u)\) on the 12 edges of domain D, i.e., \( S^{-1}: \partial_{edge} D \rightarrow \partial_{edge} P \), together with the prescribed mapping of boundary point distributions on all 6 face, i.e. \( X^{-1}: \partial_{face} D \rightarrow \partial_{face} C \) (and hence, includes 12 edges, \( X^{-1}: \partial_{face} D \rightarrow \partial_{face} C \)), the mapping from computational space C to parameter space P can be established for points on the edges; in other word, the mapping \( P: \partial_{edge} C \rightarrow \partial_{edge} P \) is defined. This concept is outlined in the following paragraph.

Combining equation (3.13) and the first three equation in (3.14), one may conclude that
\[ s(0, \eta, \zeta) = 0, \quad s(1, \eta, \zeta) = 1 \]
\[ t(\xi, 0, 0) = t_{E1}(\eta), \quad t(\xi, 1, 0) = t_{E2}(\eta), \quad t(\xi, 0, 1) = t_{E3}(\eta), \quad t(\xi, 1, 1) = t_{E4}(\eta). \]
\[ u(0, 0, \zeta) = u_{E1}(\xi), \quad u(1, 0, \zeta) = u_{E2}(\xi), \quad u(0, 1, \zeta) = u_{E3}(\xi), \quad u(1, 1, \zeta) = u_{E4}(\xi). \]

The last three criteria in Eq. (3.14) can be formulated as
\[ s(\xi,0,0) = s_{E1}(\xi), \quad s(\xi,1,0) = s_{E2}(\xi), \quad s(\xi,0,1) = s_{E3}(\xi), \quad s(\xi,1,1) = s_{E4}(\xi). \]
\[ t(0,\eta,0) = t_{E1}(\eta), \quad t(1,\eta,0) = t_{E2}(\eta), \quad t(0,\eta,1) = t_{E3}(\eta), \quad t(1,\eta,1) = t_{E4}(\eta). \]
\[ u(0,0,\zeta) = u_{E1}(\xi), \quad u(1,0,\zeta) = u_{E2}(\xi), \quad u(0,1,\zeta) = u_{E3}(\xi), \quad u(1,1,\zeta) = u_{E4}(\xi). \]

where the functions \( s_{E1}(\xi), \ldots \) etc, are monotonically increasing functions representing the normalized arc length.
Having \((s, t, u)\) defined for points on the edges, the algebraic mapping \(P\) is established via an "algebraic bilinear transformation" as follows

\[
\begin{align*}
    s &= s_E(1-t)(1-u) + s_E(1-t)u + s_E u, \\
    t &= t_E(1-s)(1-u) + t_E(1-s)u + t_E u, \\
    u &= u_E(1-s)(1-t) + u_E(1-s)t + u_E t,
\end{align*}
\]  

(3.15)

This algebraic bilinear transformation is a three-dimensional extension of the algebraic straight-line interpolation. It can be easily seen that a coordinate plane in the computational space \(C\) \((\xi=\text{const.}, \eta=\text{const.} \text{ or } \zeta=\text{const.})\) is mapped onto a bilinear surface in parameter space \(P\). For example, a \(\xi=\text{const.}\) plane is mapped to a bilinear surface in which \(s\) is a bilinear function of \(t\) and \(u\). For a given \((\xi, \eta, \zeta)\) triplet, the corresponding \((s, t, u)\) coordinates are determined by the intersection points of the three bilinear surfaces described by the first, second and last equation of Eq. (3.15), respectively. By evaluating the Jacobian of the algebraic mapping, it is shown that the mapping is one-to-one and differentiable. Note also that this algebraic mapping depends on the point distribution on the edge only, rather than points on the surface. This concludes the description of the algebraic mapping, and the Laplacian mapping will be explained next.

From the boundary point distribution \(X:\partial\text{face}C \rightarrow \partial\text{face}D\) and the previously defined algebraic transformation \(P:C \rightarrow P\) (and thus \(P:\partial\text{face}C \rightarrow \partial\text{face}P\)), one can set up a mapping between points on the boundary of the parameter space and the boundary of the physical space, \(S'\)  

\[
\begin{align*}
    \Delta s &= s_{xx} + s_{yy} + s_{zz} = 0, \\
    \Delta t &= t_{xx} + t_{yy} + t_{zz} = 0, \\
    \Delta u &= u_{xx} + u_{yy} + u_{zz} = 0
\end{align*}
\]  

(3.16)
Thus, a well posed linear elliptic boundary value problem (BVP) defines the mapping from the physical domain to the parametric space $S^{-1}: D \rightarrow P$.

In contrast to its two-dimensional counterpart, this three-dimensional Laplacian mapping has two distinct features worth to note. First, for three-dimensional problems, there is no maximum principle available for the solution of the 3D Laplace equation. As a consequence, this 3D Laplace mapping is no longer guaranteed to be one-to-one and differentiable. For this reason, in the following work, the mapping is assumed to be one-to-one and differentiable, i.e., $S: P \rightarrow D$ is assumed to exist. Secondly, the mapping $S: P \rightarrow D$ is not independent of the boundary point distribution and, thus, may not be considered as a property of the physical domain. This is because the $(s, t, u)$ values at the 6 boundary faces depend on the boundary grid distribution. It is possible to make this mapping $S: P \rightarrow D$ independent of the boundary point distribution by requiring the point distributions on the 6 bounding faces follow the Laplace-Beltrami equation. But this approach will render the simple algebraic bilinear transformation no longer feasible.

Combining the two mappings $P: C \rightarrow P$ and $S: P \rightarrow D$, a new mapping $X: C \rightarrow D$ may be formed which meets the 3 criteria cited previously. To obtain a governing differential equation for the composite grid generating system, consider first the covariant base vector

$$a_1 = \frac{\partial x}{\partial \xi} = x_\xi, \quad a_2 = \frac{\partial x}{\partial \eta} = x_\eta, \quad a_3 = \frac{\partial x}{\partial \zeta} = x_\zeta$$

and introduce the covariant metric tensor components

$$a_{ij} = (a_i, a_j), \quad i = \{1, 2, 3\}, \quad j = \{1, 2, 3\}.$$ 

The contravariant base vector is then defined according to the condition of orthogonality that

$$a_i^j = \delta_{ij}, \quad i = \{1, 2, 3\}, \quad j = \{1, 2, 3\}$$

where $\delta_{ij}$ is the Kronecker delta. The contravariant metric tensor components are then defined as
\[ a^i = (a^i, a^j), \quad i = \{1, 2, 3\}, \quad j = \{1, 2, 3\} \]

It can be verified easily that
\[
\begin{pmatrix}
    a_{11} & a_{12} & a_{13} \\
    a_{21} & a_{22} & a_{23} \\
    a_{31} & a_{32} & a_{33}
\end{pmatrix}
\begin{pmatrix}
    a^1 \\
    a^2 \\
    a^3
\end{pmatrix} =
\begin{pmatrix}
    1 & 0 & 0 \\
    0 & 1 & 0 \\
    0 & 0 & 1
\end{pmatrix}
\]

and
\[
\begin{align*}
    a^1 &= a^{11}a_1 + a^{12}a_2 + a^{13}a_3 \\
    a^2 &= a^{21}a_1 + a^{22}a_2 + a^{23}a_3 \\
    a^3 &= a^{31}a_1 + a^{32}a_2 + a^{33}a_3
\end{align*}
\]

As in two dimensions, the following equation expresses the relation of derivatives in these two systems:
\[
\Delta \phi = \frac{1}{J} \left\{ J a^{11} \phi_{\xi} + J a^{12} \phi_{\eta} + J a^{13} \phi_{\zeta} \right\} + \left\{ J a^{21} \phi_{\xi} + J a^{22} \phi_{\eta} + J a^{23} \phi_{\zeta} \right\} + \left\{ J a^{31} \phi_{\xi} + J a^{32} \phi_{\eta} + J a^{33} \phi_{\zeta} \right\}
\]

(3.17)

where \( J \) is the square root of the determinant \( J^2 \) of the covariant metric tensor. Taking a special case that \( \phi = \xi, \phi = \eta \) and \( \phi = \zeta \) respectively, the following expressions are obtained.
\[
\begin{align*}
    \Delta \xi &= \frac{1}{J} \left\{ J a^{11} \right\} + \left\{ J a^{21} \right\} + \left\{ J a^{31} \right\} , \\
    \Delta \eta &= \frac{1}{J} \left\{ J a^{12} \right\} + \left\{ J a^{22} \right\} + \left\{ J a^{32} \right\} , \\
    \Delta \zeta &= \frac{1}{J} \left\{ J a^{13} \right\} + \left\{ J a^{23} \right\} + \left\{ J a^{33} \right\} .
\end{align*}
\]

(3.18)

Equation (3.17) can be re-written, together with equation (3.18), as
\[
\Delta \phi = \left( a^{11} \phi_{\xi} + a^{12} \phi_{\eta} + a^{13} \phi_{\zeta} \right) + \left( a^{21} \phi_{\xi} + a^{22} \phi_{\eta} + a^{23} \phi_{\zeta} \right) + \left( a^{31} \phi_{\xi} + a^{32} \phi_{\eta} + a^{33} \phi_{\zeta} \right) + \left( a^{11} \phi_{\eta} + a^{22} \phi_{\eta} + a^{33} \phi_{\zeta} \right) + \left( a^{11} \phi_{\zeta} + a^{22} \phi_{\zeta} + a^{33} \phi_{\zeta} \right)
\]

(3.19)

which will serve as a fundamental relation in the following derivation.
Let \( \phi = s, \phi = t \) and \( \phi = u \), respectively, in Eq. (3.19), a relation between \( \Delta s, \Delta t, \Delta u \) and \( \Delta \xi, \Delta \eta, \Delta \zeta \) is then achieved.

\[
\begin{align*}
\Delta s &= (a^{11} s_{\xi} + a^{22} s_{\eta} + a^{33} s_{\zeta}) + 2(a^{12} s_{\eta} + a^{23} s_{\xi} + a^{31} s_{\zeta}) + (\Delta \xi s_{\xi} + \Delta \eta s_{\eta} + \Delta \zeta s_{\zeta}) \\
\Delta t &= (a^{11} t_{\xi} + a^{22} t_{\eta} + a^{33} t_{\zeta}) + 2(a^{12} t_{\eta} + a^{23} t_{\xi} + a^{31} t_{\zeta}) + (\Delta \xi t_{\xi} + \Delta \eta t_{\eta} + \Delta \zeta t_{\zeta}) \\
\Delta u &= (a^{11} u_{\xi} + a^{22} u_{\eta} + a^{33} u_{\zeta}) + 2(a^{12} u_{\eta} + a^{23} u_{\xi} + a^{31} u_{\zeta}) + (\Delta \xi u_{\xi} + \Delta \eta u_{\eta} + \Delta \zeta u_{\zeta})
\end{align*}
\]

The assertion that \( \Delta s = 0, \Delta t = 0 \) and \( \Delta u = 0 \) allows expressions for \( \Delta \xi, \Delta \eta \) and \( \Delta \zeta \) to be written in terms of \( s_{\xi}, s_{\xi\eta}, t_{\xi}, t_{\xi\eta} \) etc.

\[
\begin{bmatrix}
\Delta \xi \\
\Delta \eta \\
\Delta \zeta 
\end{bmatrix} =
\begin{bmatrix}
0 \\
0 \\
0 
\end{bmatrix}
\]

where

\[
\begin{bmatrix}
P_{11} \\
P_{22} \\
P_{33} \\
P_{12} \\
P_{23} \\
P_{31}
\end{bmatrix} =
\begin{bmatrix}
P_{11}^{-1} \\
P_{22}^{-1} \\
P_{33}^{-1} \\
P_{12}^{-1} \\
P_{23}^{-1} \\
P_{31}^{-1}
\end{bmatrix}
\]

Let \( \phi = x \) in Eq. (3.19) and realize that \( \Delta x \equiv 0 \); one then obtains

\[
\begin{bmatrix}
(a^{11} x_{\xi} + a^{22} x_{\eta} + a^{33} x_{\zeta}) + 2(a^{12} x_{\eta} + a^{23} x_{\xi} + a^{31} x_{\zeta}) + (\Delta \xi x_{\xi} + \Delta \eta x_{\eta} + \Delta \zeta x_{\zeta}) = 0
\end{bmatrix}
\]

Substituting \( \Delta \xi, \Delta \eta \) and \( \Delta \zeta \) from Eq. (3.20) and expressing the contravariant tensor components in terms of covariant tensor components, equation (3.23) may be further reduced to
\[
\begin{align*}
(\alpha^{11}x_{\xi\xi} + \alpha^{22}x_{\eta\eta} + \alpha^{33}x_{\zeta\zeta}) + 2(\alpha^{12}x_{\xi\eta} + \alpha^{23}x_{\eta\zeta} + \alpha^{31}x_{\zeta\xi}) & + \left[\left(\alpha^{11}P_{11}^1 + \alpha^{22}P_{22}^1 + \alpha^{33}P_{33}^1\right) + 2\left(\alpha^{12}P_{12}^1 + \alpha^{23}P_{23}^1 + \alpha^{31}P_{31}^1\right)\right]x_{\xi} \\
+ \left[\left(\alpha^{11}P_{11}^2 + \alpha^{22}P_{22}^2 + \alpha^{33}P_{33}^2\right) + 2\left(\alpha^{12}P_{12}^2 + \alpha^{23}P_{23}^2 + \alpha^{31}P_{31}^2\right)\right]x_{\eta} & + \left[\left(\alpha^{11}P_{11}^3 + \alpha^{22}P_{22}^3 + \alpha^{33}P_{33}^3\right) + 2\left(\alpha^{12}P_{12}^3 + \alpha^{23}P_{23}^3 + \alpha^{31}P_{31}^3\right)\right]x_{\zeta} = 0
\end{align*}
\]

(3.24)

This is the grid generating equation for three-dimensional space. The numerical solution of Eq. (3.24) will be described in the flowing sections.

In the next sub-section where time dependent grid equation is derived, an alternative form of Eq. (3.24) will be referenced. This form is consistent with that of Thompson and is written as

\[
\begin{align*}
\alpha^{11} = a_{12}a_{33} - a_{23}^2 & \quad \alpha^{12} = a_{13}a_{23} - a_{12}a_{33} \\
\alpha^{22} = a_{11}a_{33} - a_{13}^2 & \quad \alpha^{23} = a_{21}a_{33} - a_{23}a_{11} \\
\alpha^{33} = a_{11}a_{22} - a_{23}^2 & \quad \alpha^{31} = a_{32}a_{12} - a_{31}a_{22}
\end{align*}
\]

(3.25)

and

\[
\begin{align*}
a_{11} = (x_\xi, x_\eta) & \quad a_{22} = (x_\eta, x_\eta) \quad a_{33} = (x_\zeta, x_\zeta) \\
a_{12} = (x_\xi, x_\eta) & \quad a_{23} = (x_\eta, x_\zeta) \quad a_{31} = (x_\zeta, x_\xi)
\end{align*}
\]

This completes the description of grid generating equation for a stationary grid system.

Time-dependent grid systems will be treated in the next section. To utilize the concept of the
multibox scheme, it is necessary to implement several modifications. These modifications will be discussed in Sections 3.2 and 3.3.

3.1.2 Time-Varying Grid Generation - the Grid-Transport Equations

In this section, the well-known grid transport equation will be derived using an alternative procedure other than the one given in Thompson et al. [56]. From section 3.1.1, it is known that one of the elliptic grid generating equation for fixed domain is

\[
\begin{align*}
\alpha^{11} x_{\xi\xi} + \alpha^{22} x_{\eta\eta} + 2\alpha^{12} x_{\xi\eta} + J^2 (P_x + Q_x\eta) &= 0 \\
\alpha^{11} y_{\xi\xi} + \alpha^{22} y_{\eta\eta} + 2\alpha^{12} y_{\xi\eta} + J^2 (P_y + Q_y\eta) &= 0
\end{align*}
\]

for 2-D

and

\[
\begin{align*}
\left(\alpha^{11} x_{\xi\xi} + \alpha^{22} x_{\eta\eta} + \alpha^{33} x_{\zeta\zeta}\right) + 2\left(\alpha^{12} x_{\xi\eta} + \alpha^{13} x_{\xi\zeta} + \alpha^{23} x_{\eta\zeta}\right) + J^2 (P_x + Q_x\eta + R_x\zeta) &= 0 \\
\left(\alpha^{11} y_{\xi\xi} + \alpha^{22} y_{\eta\eta} + \alpha^{33} y_{\zeta\zeta}\right) + 2\left(\alpha^{12} y_{\xi\eta} + \alpha^{13} y_{\xi\zeta} + \alpha^{23} y_{\eta\zeta}\right) + J^2 (P_y + Q_y\eta + R_y\zeta) &= 0 \\
\left(\alpha^{11} z_{\xi\xi} + \alpha^{22} z_{\eta\eta} + \alpha^{33} z_{\zeta\zeta}\right) + 2\left(\alpha^{12} z_{\xi\eta} + \alpha^{13} z_{\xi\zeta} + \alpha^{23} z_{\eta\zeta}\right) + J^2 (P_z + Q_z\eta + R_z\zeta) &= 0
\end{align*}
\]

for 3-D

Thompson [56] shows that these equations are actually the inverse form of the relation

\[
\begin{align*}
\nabla^2 \zeta &= \xi_{xx} + \xi_{yy} = P \\
\nabla^2 \eta &= \eta_{xx} + \eta_{yy} = Q
\end{align*}
\]

for 2-D

and

\[
\begin{align*}
\nabla^2 \zeta &= \xi_{xx} + \xi_{yy} + \xi_{zz} = P \\
\nabla^2 \eta &= \eta_{xx} + \eta_{yy} + \eta_{zz} = Q \\
\nabla^2 \zeta &= \zeta_{xx} + \zeta_{yy} + \zeta_{zz} = R
\end{align*}
\]

for 3-D

which are the Poisson equation in physical domain.

To extend the above elliptic generation technique to time-varying grid problem, a straight-forward augmentation is to "parabolize" the Poisson equation so that the time-varying version of the grid generation equation become

\[
\begin{align*}
\xi_t &= \xi_{xx} + \xi_{yy} - P \\
\eta_t &= \eta_{xx} + \eta_{yy} - Q
\end{align*}
\]

for 2-D

and

\[
\begin{align*}
\xi_t &= \xi_{xx} + \xi_{yy} + \xi_{zz} - P \\
\eta_t &= \eta_{xx} + \eta_{yy} + \eta_{zz} - Q \\
\zeta_t &= \zeta_{xx} + \zeta_{yy} + \zeta_{zz} - R
\end{align*}
\]

for 3-D

For the same rational as for steady-grid case, it is necessary to interchange the role of dependent and independent variables so that a finite difference solution procedure may be carried
on the computational domain. The resulting equation, with \((\xi, \eta, \zeta)\) as independent variables and 
\((x,y,z)\) as dependent variables, is known as the grid transport equation because its structure 
resembles that of the transport equation in fluid mechanics. The algebra involved in this 
inversion derivation is quite tedious, so the symbolic algebra package Mathematica is employed 
to reduce the human labor and chances of error.

The derivation starts from forming the Jacobian matrix of the transformation \(X:C \rightarrow D\) 
which can be written as (text written in bold are user input whereas text written in normal are 
Mathematica response)

\[
J = \begin{bmatrix}
\frac{\partial x}{\partial \tau}, \frac{\partial x}{\partial \xi}, \frac{\partial x}{\partial \eta}, \frac{\partial x}{\partial \zeta} \\
\frac{\partial y}{\partial \tau}, \frac{\partial y}{\partial \xi}, \frac{\partial y}{\partial \eta}, \frac{\partial y}{\partial \zeta} \\
\frac{\partial z}{\partial \tau}, \frac{\partial z}{\partial \xi}, \frac{\partial z}{\partial \eta}, \frac{\partial z}{\partial \zeta}
\end{bmatrix};
\]

MatrixForm[J]

\[
\begin{bmatrix}
1 & 0 & 0 & 0 \\
x_\tau & x_\xi & x_\eta & x_\zeta \\
y_\tau & y_\xi & y_\eta & y_\zeta \\
z_\tau & z_\xi & z_\eta & z_\zeta
\end{bmatrix}
\]

To evaluate the inverse matrix of \(J\), the Mathematica function Inverse is used,

\[
\begin{bmatrix}
a & b & c & d \\
\xi_\tau & \xi_\xi & \xi_\eta & \xi_\zeta \\
\eta_\tau & \eta_\xi & \eta_\eta & \eta_\zeta \\
\zeta_\tau & \zeta_\xi & \zeta_\eta & \zeta_\zeta
\end{bmatrix} = \text{Inverse}[J];
\]

By inspection of the matrix \(J\), the value of \(a, b, c\) and \(d\) should be evaluated to 1, 0, 0 and 0, the 
following command confirms.

\{a,b,c,d\}

\{1,0,0,0\}
The value of transformation metrics, i.e., $\xi_x$, $\xi_y$, etc. may hence be extracted from this inverse matrix. For example

$$
\xi_x
$$

$$
\frac{z_\xi y_\eta - y_\xi z_\eta}{z_\xi y_\eta x_\xi - y_\xi z_\eta x_\xi - z_\xi x_\eta y_\xi + x_\xi z_\eta y_\xi + y_\xi x_\eta z_\xi - x_\xi y_\eta z_\xi}
$$

To obtain expression in the Poisson equation, it is necessary to utilize the chain rule. For example,

$$
\xi_{xx} = \frac{\partial}{\partial x} (\xi_x) = \frac{\partial}{\partial \xi} (\xi_x) \frac{\partial \xi}{\partial x} + \frac{\partial}{\partial \eta} (\xi_x) \frac{\partial \eta}{\partial x} + \frac{\partial}{\partial \zeta} (\xi_x) \frac{\partial \zeta}{\partial x}
$$

The following command demonstrates how a computer algebra system such as Mathematica can help in tedious algebraic manipulation and thus avoid error. The Mathematica command Expand and Together are simply used to simplify the resulting expression. Expand expands all the product and Together put every terms over a common denominator.

$$
\xi_{xx} = \text{Together}[\text{Expand}[\partial_\xi \xi_x \xi_x + \partial_\eta \xi_x \eta_x + \partial_\zeta \xi_x \xi_x]]
$$
\[(z_y y_x x_n z_y^2 y_y^2 - y_c z_y x_n z_y^2 y_y^2 - z_c x_y x_n y_y z_y^2 + x_c z_c x_y x_n y_y z_y^2 + y_c x_c z_c z_n^3 y_y^2 - x_c y_c z_c z_n^3 y_y^2 + 2 z_c y_y z_n x_n z_y^2 y_y^2 - 2 y_c z_c z_n^2 x_n z_n y_y z_y^2 - 2 z_c^2 x_n z_n y_y z_y^2 + 2 z_c z_n^2 x_n z_n y_y y_y^2 + 2 y_c z_c z_n^2 x_n z_n y_y y_y^2 + 2 z_c y_n^2 x_n y_y y_y^2 - 2 z_c^2 x_n y_n y_y y_y^2 + 2 y_c z_c y_n^2 x_n y_n y_y y_y^2 + 2 y_c z_c x_n y_n y_y y_y^2 + 2 z_c^2 y_n y_n y_y y_y^2 + 2 y_c z_c x_n y_n y_y y_y^2 + 2 y_c z_c x_n y_n y_y y_y^2)\]

Now one may form the 'parabolized' Poisson equation as follows:

\[
eq 1 = \text{Numerator[Together[(P + } z_c) - (z_y z_x + z_y z_y + z_y z_n)]};
\]

\[
eq 2 = \text{Numerator[Together[(Q + } z_c) - (z_y z_x + z_y z_y + z_y z_n)]};
\]

\[
eq 3 = \text{Numerator[Together[(R + } z_c) - (z_y z_x + z_y z_y + z_y z_n)]};
\]
All of the terms in the 'parabolized' Poisson equation was moved to the left hand side of the equation and so leaves the right hand side to be zero. Because LHS is equal to 0, so does its numerator. The command **Numerator** is used to extract the numerator of a fractional expression.

It should be easy to imagine how huge an expression like \( eq1 \) is. However, it is still possible to cast this expression into a canonical form

\[
\begin{align*}
eq 1: & \quad a_1 x_{\xi \xi} + a_2 x_{\eta \eta} + a_3 x_{\xi \xi} + a_4 x_{\xi \eta} + a_5 x_{\eta \xi} + a_6 x_{\eta \eta} + \\
eq 2: & \quad b_1 x_{\xi \xi} + b_2 x_{\eta \eta} + b_3 x_{\xi \xi} + b_4 x_{\xi \eta} + b_5 x_{\eta \xi} + b_6 x_{\eta \eta} + \\
eq 3: & \quad c_1 x_{\xi \xi} + c_2 x_{\eta \eta} + c_3 x_{\xi \xi} + c_4 x_{\xi \eta} + c_5 x_{\eta \xi} + c_6 x_{\eta \eta} + \\
\end{align*}
\]

The following command is used for this purpose.

\[
\text{eq1} = \text{Collect}[\text{eq1}, \{
\partial_{\xi, \xi} x[\tau, \xi, \eta, \xi], \partial_{\eta, \eta} x[\tau, \xi, \eta, \xi], \partial_{\xi, \xi} x[\tau, \xi, \eta, \xi], \\
\partial_{\eta, \eta} x[\tau, \xi, \eta, \xi], \partial_{\eta, \eta} y[\tau, \xi, \eta, \xi], \partial_{\xi, \xi} y[\tau, \xi, \eta, \xi], \\
\partial_{\xi, \xi} z[\tau, \xi, \eta, \xi], \partial_{\eta, \eta} z[\tau, \xi, \eta, \xi], \partial_{\xi, \xi} z[\tau, \xi, \eta, \xi]
\}];
\]

the coefficient 'a1' is extracted to be

\[
\text{al} = \text{Coefficient}[\text{eq1}, \partial_{\xi, \xi} x[\tau, \xi, \eta, \xi]]
\]

\[
\begin{align*}
y_\xi x_\xi^2 y_\eta + z_\xi^3 x_\eta + 2 x_\xi^2 y_\xi x_\eta^2 + x_\xi^2 z_\xi^3 y_\eta^3 + 2 x_\xi^2 z_\xi^3 y_\eta + y_\xi^2 z_\xi^3 y_\eta + & \\
2 x_\xi y_\xi z_\xi x_\eta + 2 x_\xi^2 y_\xi^2 x_\eta + 2 x_\xi^2 z_\xi^2 x_\eta - 2 x_\xi^2 y_\xi^2 y_\eta + & \\
2 x_\xi y_\xi z_\xi y_\eta^2 + 2 x_\xi^2 y_\xi^2 y_\eta + 3 y_\xi^2 z_\xi^2 y_\eta^2 + & \\
\end{align*}
\]
As a matter of fact, it is found out that none of the set of numbers \(a_1, a_2, a_3, \ldots, a_{18}, b_1, b_2, \ldots, b_{18}, c_1, c_2, \ldots, c_{18}\) are zero. In other words, the equations described by 'eq1=0', 'eq2=0' and 'eq3=0' are coupled equation with unknown \(x, y, z\) appeared simultaneously. In view of eq1, it is anticipated that by performing certain linear combination, one can eliminates this coupling between unknowns. In fact, the parameter \(\alpha, \beta\) is sought in the expression eq4, a linear combination of eq1, eq2 and eq3, which renders the coefficient of \(y\) and \(z\) to be zero.

\[
\text{Clear}[\alpha, \beta] \\
\text{eq4}=\text{eq1}+\alpha \text{eq2}+\beta \text{eq3};
\]

\[
\text{tmp} = \text{Expand[eq4]};
\]

\[
\text{cond} = \{\text{Coefficient[tmp, } \partial_{\xi,\xi} y[\tau, \xi, \eta, \xi]] = 0, \text{Coefficient[tmp, } \partial_{\xi,\xi} z[\tau, \xi, \eta, \xi]] = 0\};
\]

\[
\text{s} = \text{Solve[cond, \{\alpha, \beta\}]};
\]

The variable \(s\) contains expression for \(\alpha, \beta\) that will 'decouple' the unknowns in eq1. Performing substitution and re-cast them into the canonical form

\[
\text{eq5} = \text{eq4} / . \text{s[[1]]};
\]

\[
\text{eq5} = \text{Numerator[Together[eq5]]};
\]

it is shown that eq5 contains only second order derivatives of \(x\) but not those of \(y\) and \(z\), i.e.,

\[
\text{eq5} = \text{Collect[eq5,}
\{\partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\eta,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi], \partial_{\xi,\xi} x[\tau, \xi, \eta, \xi], \partial_{\xi,\eta} x[\tau, \xi, \eta, \xi]\};
\]
\( x_7 = \text{Coefficient}[\text{eq5}, \partial_{\xi,\xi} y[\tau, \xi, \eta, \xi]] \)
\( x_8 = \text{Coefficient}[\text{eq5}, \partial_{\eta,\eta} y[\tau, \xi, \eta, \xi]] \)
\( x_9 = \text{Coefficient}[\text{eq5}, \partial_{\xi,\xi} y[\tau, \xi, \eta, \xi]] \)
\( x_{10} = \text{Coefficient}[\text{eq5}, \partial_{\xi,\eta} y[\tau, \xi, \eta, \xi]] \)
\( x_{11} = \text{Coefficient}[\text{eq5}, \partial_{\eta,\xi} y[\tau, \xi, \eta, \xi]] \)
\( x_{12} = \text{Coefficient}[\text{eq5}, \partial_{\xi,\xi} z[\tau, \xi, \eta, \xi]] \)
\( x_{13} = \text{Coefficient}[\text{eq5}, \partial_{\eta,\eta} y[\tau, \xi, \eta, \xi]] \)
\( x_{14} = \text{Coefficient}[\text{eq5}, \partial_{\eta,\eta} z[\tau, \xi, \eta, \xi]] \)
\( x_{15} = \text{Coefficient}[\text{eq5}, \partial_{\xi,\eta} z[\tau, \xi, \eta, \xi]] \)
\( x_{16} = \text{Coefficient}[\text{eq5}, \partial_{\xi,\eta} z[\tau, \xi, \eta, \xi]] \)
\( x_{17} = \text{Coefficient}[\text{eq5}, \partial_{\xi,\xi} z[\tau, \xi, \eta, \xi]] \)
\( x_{18} = \text{Coefficient}[\text{eq5}, \partial_{\xi,\eta} z[\tau, \xi, \eta, \xi]] \)

0
0
0
0
0
0
0
0
0
0
0
0

It can also be noticed that \text{eq5} can be further simplified.

\( \text{eq6} = \text{Factor}[\text{eq5}] \)
The first factor is simply the determinant of the Jacobian matrix that should never be zero. So 

eq5 can be reduced to eq6 by dividing this determinant

\[ \text{eq6} = \text{Simplify} \left( \frac{\text{eq6}}{\text{Det}[J]} \right) ; \]

Putting them into the canonical form again, one can achieve the following coefficients:

\[ \text{eq6} = \text{Collect}[\text{eq6}, \{ \partial_{\zeta, \xi} x[t, \xi, \eta, \zeta], \partial_{\eta, \eta} x[t, \xi, \eta, \zeta], \partial_{\zeta, \zeta} x[t, \xi, \eta, \zeta], \partial_{\zeta, \eta} x[t, \xi, \eta, \zeta], \partial_{\zeta, \zeta} x[t, \xi, \eta, \zeta], \partial_{\zeta, \xi} x[t, \xi, \eta, \zeta], \partial_{\zeta, \eta} y[t, \xi, \eta, \zeta], \partial_{\zeta, \xi} y[t, \xi, \eta, \zeta], \partial_{\zeta, \eta} y[t, \xi, \eta, \zeta], \partial_{\zeta, \zeta} y[t, \xi, \eta, \zeta], \partial_{\zeta, \xi} y[t, \xi, \eta, \zeta], \partial_{\zeta, \eta} y[t, \xi, \eta, \zeta], \partial_{\zeta, \zeta} y[t, \xi, \eta, \zeta], \partial_{\zeta, \xi} z[t, \xi, \eta, \zeta], \partial_{\zeta, \eta} z[t, \xi, \eta, \zeta], \partial_{\zeta, \zeta} z[t, \xi, \eta, \zeta] \} \} ; \]

A1 = Coefficient[eq6, \partial_{\zeta, \xi} x[t, \xi, \eta, \zeta]]

A2 = Coefficient[eq6, \partial_{\eta, \eta} x[t, \xi, \eta, \zeta]]

A3 = Coefficient[eq6, \partial_{\zeta, \zeta} x[t, \xi, \eta, \zeta]]

A4 = Coefficient[eq6, \partial_{\zeta, \xi} x[t, \xi, \eta, \zeta]]

A5 = Coefficient[eq6, \partial_{\eta, \xi} x[t, \xi, \eta, \zeta]]

A6 = Coefficient[eq6, \partial_{\zeta, \xi} x[t, \xi, \eta, \zeta]]
These expression are the coefficient of the second derivatives. So eq6 may be re-written as

\[ \text{eq6} = A_1 \ x_{\xi\xi} + A_2 \ x_{\eta\eta} + A_3 \ x_{\zeta\zeta} + A_4 \ x_{\xi\eta} + A_5 \ x_{\eta\zeta} + A_6 \ x_{\zeta\xi} + S \]

where \( S \) can be evaluated as

\[
S = \text{Expand} [ \text{eq6} - (A_1 \cdot \partial_{\xi,\xi} x[\tau, \xi, \eta, \zeta]) +
A_2 \cdot \partial_{\eta,\eta} x[\tau, \xi, \eta, \zeta] +
A_3 \cdot \partial_{\zeta,\zeta} x[\tau, \xi, \eta, \zeta] +
A_4 \cdot \partial_{\xi,\eta} x[\tau, \xi, \eta, \zeta] +
A_5 \cdot \partial_{\eta,\zeta} x[\tau, \xi, \eta, \zeta] +
A_6 \cdot \partial_{\zeta,\xi} x[\tau, \xi, \eta, \zeta] ) ];
\]

\[
S = \text{Factor} [ \text{Expand}[S] ]
\]

\[
(-z_\xi y_\eta x_\xi + y_\xi z_\eta x_\xi + z_\xi x_\eta y_\xi - x_\xi z_\eta y_\xi + y_\eta z_\xi x_\eta + x_\eta y_\xi z_\eta) \cdot (\text{R} \cdot x_\xi + Q \cdot x_\eta + P \cdot x_\zeta - x_\tau)
\]

which is simply

\[ J^2(P x_\xi + Q x_\eta + Rx_\zeta - x_\tau) \]

So the equation 'eq6=0' can be written as

\[
A_1 x_{\xi\xi} + A_2 x_{\eta\eta} + A_3 x_{\zeta\zeta} + A_4 x_{\xi\eta} + A_5 x_{\eta\zeta} + A_6 x_{\zeta\xi} + J^2(P x_\xi + Q x_\eta + Rx_\zeta - x_\tau) = 0 \quad (3.1a)
\]

Applying the same procedure of decoupling of second derivatives terms in eq2 and eq3 via linear combination, and factor-out the Jacobian in the resulting expression, one may obtains the equations

\[
A_1 y_{\xi\xi} + A_2 y_{\eta\eta} + A_3 y_{\zeta\zeta} + A_4 y_{\xi\eta} + A_5 y_{\eta\zeta} + A_6 y_{\zeta\xi} + J^2(P y_\xi + Q y_\eta + Ry_\zeta - y_\tau) = 0
\]

\[
(3.28b)
\]

\[
A_1 z_{\xi\xi} + A_2 z_{\eta\eta} + A_3 z_{\zeta\zeta} + A_4 z_{\xi\eta} + A_5 z_{\eta\zeta} + A_6 z_{\zeta\xi} + J^2(P z_\xi + Q z_\eta + Rz_\zeta - z_\tau) = 0
\]

\[
(3.28c)
\]

Note also that the coefficient of \( x_{\xi\xi} \) is identical to that of \( y_{\xi\xi} \) and \( z_{\xi\xi} \), etc.

In summary, the resulting equation (3.1a-c) provides a system of partial differential equations that is to be solved in the computational domain for generating time-varying grid
system. This grid system is useful for flow problems with moving boundary as well as flow-adaptive grid generation applications. Numerical solution of these equations are detailed in the next sections.

3.2 Extensions to Multi-Box Scheme

3.2.1 Multi-Box Scheme
As mentioned in Section 1.3, traditional finite-difference methods utilize mappings that transform the physical domain onto a simpler computational domain, and then perform numerical solution on this transformed domain. The transformed computational domain is usually of rectangular shape in 2D, or of box shape in 3D. For complex physical domains, this mapping may not be easily found. In the research, it is proposed to relax the requirement of "one simple rectangular computational domain in 2D" or "one simple box computational domain in 3D". In other words, a multi-box shape is employed. Therefore, the goal for numerical grid generation is modified to be "finding a mapping which transforms the active part of the computational domain onto the entire physical domain."

It is best to illustrate this multi-box concept by a simple 2D example. In Figure 9, the same physical domain is discretized in two different ways. Figure 9a demonstrates the traditional way that maps the entire physical domain onto the entire computational domain. On the other hand, Figure 9b illustrates the proposed multi-rectangular mapping which maps the entire physical domain onto the active part of the computational domain. The definition of active part is controlled by an IBLANK array in which IBLANK=1 indicates an active point, IBLANK=0 indicate an inactive point and IBLANK=2 indicate point on the boundary of active part. Thus the numerical solution scheme checks for active points. If, at a point, IBLANK is greater than or equal to 1, then the governing equation is applied to this point and the solution procedure executed.
It can be seen that, with the new mapping, the grid distortion at the corner is much smaller than with the traditional mapping. If the geometry of the active part in the computational domain is more or less resembles the geometry of the physical domain, this distortion can be minimized. Therefore, in contrast to traditional mapping, the multi-rectangular mapping provides an added flexibility in controlling grid distortion.

Figure 9a. Traditional Mapping.

Figure 7b. Multi-Rectangular Mapping.
On the other hand, there are some disadvantages in the multi-rectangular mappings. As there are more than 4 corners in the active part of the computational domain, quite often it is necessary to map these corners of the computational domain onto points on a smooth boundary in the physical domain. This creates singularities at these boundary points (Jacobian=0). In the flow solution phase, this issue is more pronounced if Neumann type of boundary condition is applied. As a result, it is anticipated that the flow results may be less accurate at the vicinity of these singularities, unless only Dirichlet type boundary conditions are used.

The concept of multi-rectangular mapping can be extended straightforward to three dimensions, thus multi-box mappings. Development and application of such a mapping is one of the main themes of the present research.

3.2.2 The H-H Topology

In most cases, the H-grid, O-grid or C-grid is employed in the CFD simulation. An H-grid system is usually employed in simply connected regions. In this system, the same curvilinear coordinate is held constant on each member of a pair of generally opposing boundary segments. For example, segment 1-2 in Figure 10 has a constant η value (η=η₁), while segment 4-3 possesses another value (η=η₂). In fact, the H-grid system resembles the Cartesian coordinate system in such a way that it can be thought of as a distorted version of the Cartesian system.

Figure 10. H-Grid Topology. (Illustration Adapted from Thompson)
With the H-grid, it is necessary to assign four points in the physical domain as "corners" that map onto the corners of the square in the computational domain. However, quite often, there are fewer less real corners than required. An easy example of this situation is a triangular area. In this situation, one is forced to choose points on a smooth curve or straight line as corner. This situation is illustrated in Figure 11,

![Figure 11. Singular Point on a Smooth](image)

It is apparent that this will create a singular point on the boundary because the Jacobian $J$ at this point is zero. Hence the deterioration of the solution near this point is anticipated, unless Dirichlet boundary conditions are used.

As illustrated in the previous section, application of the concept of multi-rectangular mapping to H-grid topology is straightforward and has been demonstrated in Figure 9.

For multiply-connected regions, either O-grid or C-grid topology is commonly used. Just as H-grid corresponds to a Cartesian system, so does O-grid to the polar coordinate system. In the O-grid system, a branch cut is introduced (segment 2-1 in Figure 12 ) and data communication across the branch cut needs to be implemented cautiously. A typical O-grid system is illustrated in Figure 12.
C-grid is another variant of topology for handling multiply-connected domains. In C-grid, a branch cut is also introduced. However, what makes C-grid different from O-grid is the location of branch cut in computational domain. Figure 12 and 11 shows that the branch cut is on the opposite side of the computational domain for a O-grid, whereas it is on the same side for a C-grid system.

Other topology set up is possible, though seldom employed. For example, if we take segments 4-5 and 1-2 in Figure 15 as branch cut, a new grid system can be obtained.
In three dimension, it is necessary to use the terminology such as "H-grid in xy plane, O-grid in y-z plane", etc., to describe the topology employed. A typical method of generating three-dimensional grids is to "translate" a "parent two-dimensional grid", as illustrated in Figure 15. In this case, an appropriate description of the grid system would be "H-grid in xy plane, H grid in yz plane".

![Figure 15. 3-D grid generation by translation. (Illustration Adapted from Thompson [56])](image)

Another method of generating 3D grid is by rotating a "parent 2D grid", as shown in Figure 16. An appropriate description would be "C-grid in meridional plane, O-grid in radial plane".

![Figure 16. 3-D Grid Generation by Rotation. (Illustration Adapted from Thompson [56])](image)

When dealing with complex three-dimensional domains, a composite grid structure is often used. However, as advised by Thompson [56], grid configurations with polar axis should not be used in composite grid structures. Therefore, physical domains such as a T-junction of
two pipes should only be treated by H-H type of grid systems (H grid in xy plane and H in yz plane), as illustrated in Figure 17.

The present research also employed this topology. In fact, for the graft problem in Section 5.2, all of the CFD simulations that have been surveyed use this topology, despite the error introduced in the vicinity of singular points on the boundary.

The surface grid system is also noteworthy. With the daughter branch joining the parent branch, the surface on the parent branch forms a multiply connected region. The grid system on the surface is also of H-type. As mentioned at the beginning of this section, H-grid is typically used for simply connected domains. For multiply connected regions, H-grid is not applicable without modification. The multi-rectangular technique dictated in section 3.2.1 provides exactly such a remedy for the problem. An example of the resulting surface grid system for the graft problem is shown in Figure 25.

One of the key features in the grid generation for multi-box (or multi-rectangular) domain is interpolation of the parameter. This issue is explained in the next section.
3.3 Numerical Solution of the Grid-Transport Equations

3.3.1 Douglas-Gunn ADI

The system of grid transport equations derived in Section 3.1.2 enjoys several mathematical characteristics. These characteristics can be spelled out as coupled, second-order, nonlinear, partial differential equation system of three unknowns, in three independent spatial variables and one temporal variable. To find their solutions, it is very helpful to find a similar "canonical problem" where a standard solution scheme is available. Such a problems is available, certain modifications can then be applied on the solution scheme to account for the current situation.

A very close family of problems arose from the classical mathematical physics - the three-dimensional unsteady diffusion problem. For the problem, governing equation is expressed as the diffusion equation

\[
\frac{\partial u}{\partial t} = \alpha \left( \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} \right). \tag{3.29}
\]

With proper linearization and simplification (see 3.3.2), the coupled system for the grid transport equation can be de-coupled into three uncoupled scalar equations, each of which resembles Eq. (3.29), with additional mixed derivative terms and first order convective terms. Therefore, the numerical scheme for solving the diffusion equation will be presented in this section. The necessary steps towards grid applying it for the transport equation will be discussed in the next section.

There are several classical numerical schemes for solving Eq. (3.29). One of the most promising groups is the Alternating Direction Implicit scheme (ADI). For the two-dimensional version of the diffusion equation, the Peaceman-Rachford ADI (1955) [57] method provides a scheme with second order temporal local truncation error (LTE) and is unconditionally stable.
Douglas-Rachford ADI (1956) [58] is another scheme with unconditional stability, but is only first-order accurate in temporal LTE. Improvement in temporal accuracy has been made by Mitchell and Fairweather (1964) [59] who devised a scheme of fourth-order temporal LTE. This scheme is also unconditionally stable. Finally, Douglas (1962)[60] demonstrated yet another unconditional scheme with second-order temporal LTE.

When the three-dimensional diffusion equation (3.29) is of concern, certain immediate extensions of the above scheme fail. For example, a straightforward extension of Peaceman-Rachford ADI to 3-D loses its unconditional stability. The first 3-D ADI algorithm that remains unconditionally stable is an extension of the Douglas-Rachford scheme. However, as in the 2-D case, this scheme is only of moderate temporal accuracy. Brain (1961) [61] and Douglas (1962) [62] independently developed a more accurate scheme based on the concepts of the Crank-Nicolson scheme. Mitchell and Fairweather (1965) [63] also derived an algorithm that is a 3D counter-part of their 2D ADI scheme. This scheme, however, is only conditionally stable. Later, Fairweather et. al. (1967) [64] developed another scheme, now referred to as the extended Mitchell-Fairweather scheme, that is unconditionally stable and is of fourth-order temporal LTE.

The present research employs the method of Douglas (1962)[62], usually referred to as the Douglas-Gunn scheme due to the famous paper (1964)[65], for its simplicity and accuracy. In its original form for the diffusion equation (3.29), the Douglas-Gunn scheme reads as follows:

\[
\begin{align*}
\left(1 - \frac{\rho_x}{2} \delta_{xc}\right) u^* &= \left(1 + \frac{\rho_x}{2} \delta_{ xc} + \rho_y \delta_{yy} + \rho_z \delta_{zz}\right) u^n \\
\left(1 - \frac{\rho_y}{2} \delta_{yy}\right) u^{**} &= u^* - \frac{\rho_y}{2} \delta_{yy} u^n \\
\left(1 - \frac{\rho_z}{2} \delta_{zz}\right) u^{n+1} &= u^{**} - \frac{\rho_z}{2} \delta_{zz} u^n
\end{align*}
\]  

(3.30)
where \[ \rho_x = \frac{\alpha \Delta t}{\Delta x^2}, \rho_y = \frac{\alpha \Delta t}{\Delta y^2}, \rho_z = \frac{\alpha \Delta t}{\Delta z^2} \]

\[ \delta_{xz} u^n = u^n_{x+i,j,k} - 2u^n_{i,j,k} + u^n_{i-1,j,k} \]

\[ \delta_{yy} u^n = u^n_{i,j+1,k} - 2u^n_{i,j,k} + u^n_{i,j-1,k} \]

\[ \delta_{zz} u^n = u^n_{i,j,k+1} - 2u^n_{i,j,k} + u^n_{i,j,k-1} \]

With proper scaling in x, y and z axes, respectively, the same scheme can be applied to the more general problem

\[
\frac{\partial u}{\partial t} = A_1 \frac{\partial^2 u}{\partial x^2} + A_2 \frac{\partial^2 u}{\partial y^2} + A_3 \frac{\partial^2 u}{\partial z^2} \tag{3.31}
\]

where the coefficients A1, A2 and A3 are either constant or functions of x, y and z only. In this case, only the coefficients need to be changed:

\[ \rho_x = \frac{A_1 \Delta t}{\Delta x^2}, \rho_y = \frac{A_2 \Delta t}{\Delta y^2}, \rho_z = \frac{A_3 \Delta t}{\Delta z^2} \]

For an equation with mixed cross derivative terms, McKee and Mitchell (1970) [66] provided an illustration on how these terms are handled. In their 2-D model equation,\[ u_t = \alpha u_{xx} + \beta u_{xy} + \gamma u_{yy} , \]

the ADI scheme is modified to become

\[
\left(1 - \frac{\rho_x}{2} \delta_{xc}\right) u^* = \left(1 + \frac{\rho_x}{2} \delta_{xc} + \rho_y \delta_{yy} + \tau \delta_x \delta_y\right) u^n
\]

\[
\left(1 - \frac{\rho_y}{2} \delta_{yy}\right) u^{n+1} = u^* - \frac{\rho_y}{2} \delta_{yy} u^n
\]

where \( \tau = \frac{\beta \Delta t}{\Delta x \Delta y} \) and \( \delta_x u^n = \frac{u^n_{i+1,j,k} - u^n_{i-1,j,k}}{2} \ldots \) etc.
Notice that the only modification is what appears in the first phase of the ADI algorithm where an additional term \( \tau \delta_x \delta_y u^n \) was added to the right hand side. The second phase of the ADI scheme remains unchanged. Another noteworthy issue is the accuracy. If \( \beta = 0 \), i.e., there is no mixed derivative terms, the ADI scheme is second-order accurate in temporary LTE; however with the addition of mixed derivative terms, this temporary LTE is changed to first order.

In the same token, the 3-D version of the model equation used by McKee and Mitchell can be written as

\[
\frac{\partial u}{\partial t} = A_1 \frac{\partial^2 u}{\partial x^2} + A_2 \frac{\partial^2 u}{\partial y^2} + A_3 \frac{\partial^2 u}{\partial z^2} + B_1 \frac{\partial^2 u}{\partial x \partial y} + B_2 \frac{\partial^2 u}{\partial y \partial z} + B_3 \frac{\partial^2 u}{\partial z \partial x},
\]

and the ADI algorithm is extended to be

\[
\left(1 - \frac{\rho_x}{2} \delta_{xc}\right) u^* = \left[1 + \frac{\rho_x}{2} \delta_{xc} + \rho_y \delta_{yy} + \rho_z \delta_{zz} + (\tau_1 \delta_x \delta_y + \tau_2 \delta_y \delta_z + \tau_3 \delta_z \delta_x)\right] u^n,
\]

\[
\left(1 - \frac{\rho_y}{2} \delta_{yy}\right) u^{**} = u^* - \frac{\rho_y}{2} \delta_{yy} u^n,
\]

\[
\left(1 - \frac{\rho_z}{2} \delta_{zz}\right) u^{n+1} = u^{**} - \frac{\rho_z}{2} \delta_{zz} u^n.
\]

Indeed, by combining this split form of the ADI method into one composite form as

\[
\left(1 - \frac{\rho_x}{2} \delta_{zz}\right) \left(1 - \frac{\rho_y}{2} \delta_{yy}\right) \left(1 - \frac{\rho_x}{2} \delta_{xc}\right) u^{n+1} = (O_{\text{comp}}) u^n
\]

and performing Taylor series expansion about the point \((i,j,k)\) and time instant \(n\), the original governing differential equation can be recovered with the added local truncation error which is of the first order in time. In the above,

\[
\rho_x = \frac{A_1 \Delta t}{\Delta x^2}, \quad \rho_y = \frac{A_2 \Delta t}{\Delta x^2}, \quad \rho_z = \frac{A_3 \Delta t}{\Delta x^2}, \quad \tau_1 = \frac{B_1 \Delta t}{\Delta x \Delta y}, \quad \tau_2 = \frac{B_2 \Delta t}{\Delta y \Delta z}, \quad \tau_3 = \frac{B_3 \Delta t}{\Delta z \Delta x}
\]
and (Opcomp) denotes a very complicated spatial discretization operator.

Finally, the first-order convective derivative terms need to be considered. Given the equation

$$\frac{\partial u}{\partial t} = A_1 \frac{\partial^2 u}{\partial x^2} + A_2 \frac{\partial^2 u}{\partial y^2} + A_3 \frac{\partial^2 u}{\partial z^2} + B_1 \frac{\partial^2 u}{\partial x \partial y} + B_2 \frac{\partial^2 u}{\partial y \partial z} + B_3 \frac{\partial^2 u}{\partial z \partial x} + C_1 \frac{\partial u}{\partial x} + C_2 \frac{\partial u}{\partial y} + C_3 \frac{\partial u}{\partial z}$$

(3.34)

it is desired to develop the corresponding ADI method for solving this equation. By Taylor series expansion (which is actually performed using the computer algebra system Mathematica), one can establish that the following ADI scheme is indeed the reserved extension of the Douglass-Gunn scheme:

$$\left(1 - \frac{\mu_y \delta_y + \rho_y \delta_{yy}}{2}\right)u^* = \left[1 + \frac{\mu_x \delta_x + \rho_x \delta_{xx}}{2} + (\mu_y \delta_y + \rho_y \delta_{yy}) + (\mu_z \delta_z + \rho_z \delta_{zz}) + (\tau_x \delta_x + \tau_y \delta_y + \tau_z \delta_z + \tau_x \delta_x)\right]u^n$$

$$\left(1 - \frac{\mu_y \delta_y + \rho_y \delta_{yy}}{2}\right)u^{**} = u^* - \left(\frac{\mu_y \delta_y + \rho_y \delta_{yy}}{2}\right)u^n,$$  

(3.35)

$$\left(1 - \frac{\mu_z \delta_z + \rho_z \delta_{zz}}{2}\right)u^{n+1} = u^{**} - \left(\frac{\mu_z \delta_z + \rho_z \delta_{zz}}{2}\right)u^n.$$

Each of the three steps in Eq. (3.35) involves a tri-diagonal system, and can be solved by the well-known Thomas algorithm. For the first phase of 'relaxation' in x-direction, it is necessary to 'sweep' through all y and z direction. Similarly, the second phase is for 'relaxation' in the y-direction, so that 'sweeps' in x and all z directions are required. The third phase of 'relaxation' is in z-direction and needs to accompanied by 'sweeps' in the x and y directions. Once these three phases are completed, the solution has been marched from time step n to time step n+1. This marching process can then be continued until the desired time level is reached.
As mentioned earlier in this section, the Douglass-Gunn scheme was initially designed to solve unsteady linear problems. In this situation, the superscript \( n \) represents the real physical time such that \( t = n \Delta t \), where \( \Delta t \) is the prescribed time increment. Application of the Douglass-Gunn scheme for solving steady linear problems is also possible. Starting with the initial estimated \( u^0 \) and treating the superscript \( n \) as an iteration counter; one can go through the Douglass-Gunn "marching" steps to obtain the next approximation \( u^1, u^2 \ldots \) etc. Only the converged value \( u^\infty \) has physical meaning - it is the solution of the steady equation. Intermediate values such as \( u^1, u^2 \ldots \), etc do not have physical significance and can be discarded once a better approximation has been obtained. This procedure has been implemented into a FORTRAN subroutine ADI that solves the general scalar steady PDE of the form

\[
A_1 \phi_{\xi} + A_2 \phi_{\eta} + A_3 \phi_{\zeta} + B_1 \phi_{\xi \eta} + B_2 \phi_{\eta \zeta} + B_3 \phi_{\xi \zeta} + C_1 \phi_{\xi} + C_2 \phi_{\eta} + C_3 \phi_{\zeta} = 0
\]

with Dirichlet boundary conditions. Numerically, it is not possible to obtain a true "converged" solution; consequently a certain tolerance \( \varepsilon_1 \) is necessary to be incorporated into the ADI subroutine that informs the process when to stop marching. The value of \( \varepsilon_1 \) is usually determined empirically and for most of the time in this research, \( \varepsilon_1 = 10^{-8} \).

When the Douglass-Gunn marching scheme is employed to solve steady linear problems, the effects of time step size (\( \Delta t \)) on the final converged solution is relatively insignificant. However, the time step \( \Delta t \) has a tremendous influence on the rate of convergence. A poorly selected \( \Delta t \) may require enormously larger amount of computational resource than would a properly selected one. Therefore, the question of choosing the optimal value of \( \Delta t \) is raised. Unfortunately, for all but the simplest problems there is no closed form expression for the optimal \( \Delta t \) as function of the problem parameters, and this \( \Delta t \) is problem-dependent. In practice,
the determination of optimal $\Delta t$ is done experimentally. That is, before executing a "full" run to converge to $\varepsilon_1$, certain "short runs", each for, say, 20 steps with different $\Delta t$ settings are executed, and the convergence performance is compared. Obviously, the optimal $\Delta t$ is the one with the best convergence performance. It is assumed that this optimal $\Delta t$, obtained with only 20 steps, is still the optimal $\Delta t$, among other choices for $\Delta t$, after 2000 steps, for example.

In this research, a "golden section search" [67] procedure is set up for the optimal $\Delta t$ search. This procedure is embedded in the grid solver package, and automatically chooses the $\Delta t$ that makes the convergence speed the fastest. Performing this optimal $\Delta t$ search demands certain CPU overhead, but the saving by using optimal $\Delta t$ is well paid-off.

The grid transport equation is non linear. Techniques for its linearization, and the solution procedure for solution of the time-dependent grid equation will be discussed in the next section.

3.3.2 Linearization

A closer look at equations (3.25) and (3.26) reveals that they are nonlinear. That is, the coefficients $A1$, $A2$… $A6$ are functions of $x$, $y$ and $z$. In order to obtain a numerical solution, it is necessary to linearize these equations. Although Newton's type of linearization provides a better convergence behavior, the resulting coefficients are tremendously complicated and, when it is necessary to re-evaluate them several times for restoring the nonlinearity, reduce the overall computational efficiency. As a result, the lagging-coefficient scheme that is of first order convergence has been employed.

For grid equations for the stationary grid system, the linearized version becomes:
\begin{align}
&\left(A_1^m x^{m+1}_{\xi\xi} + A_2^m x^{m+1}_{\eta\eta} + A_3^m x^{m+1}_{\xi\zeta}\right) + \left(A_4^m x^{m+1}_{\xi\eta} + A_5^m x^{m+1}_{\eta\zeta} + A_6^m x^{m+1}_{\xi\zeta}\right) + \left(J^m\right) \left(p^m x^{m+1}_{\xi\xi} + Q^m x^{m+1}_{\eta\eta} + R^m x^{m+1}_{\xi\zeta}\right) = 0 \\
&\left(A_1^m x^{m+1}_{\eta\eta} + A_2^m x^{m+1}_{\eta\eta} + A_3^m x^{m+1}_{\eta\zeta}\right) + \left(A_4^m x^{m+1}_{\eta\xi} + A_5^m x^{m+1}_{\eta\zeta} + A_6^m x^{m+1}_{\eta\zeta}\right) + \left(J^m\right) \left(p^m x^{m+1}_{\eta\xi} + Q^m x^{m+1}_{\eta\xi} + R^m x^{m+1}_{\eta\zeta}\right) = 0 \\
&\left(A_1^m x^{m+1}_{\zeta\zeta} + A_2^m x^{m+1}_{\eta\eta} + A_3^m x^{m+1}_{\zeta\eta}\right) + \left(A_4^m x^{m+1}_{\zeta\xi} + A_5^m x^{m+1}_{\zeta\eta} + A_6^m x^{m+1}_{\zeta\eta}\right) + \left(J^m\right) \left(p^m x^{m+1}_{\zeta\xi} + Q^m x^{m+1}_{\zeta\eta} + R^m x^{m+1}_{\zeta\eta}\right) = 0
\end{align}

(3.36a) (3.36b) (3.36c)

where the superscript \( m \) indicates iteration count (this iteration count is independent of the iteration count WITHIN the subroutine ADI, i.e., the superscript \( n \) in Eq. (3-35)). Also note that in equation (3.36), the coupling between unknowns, i.e., \( x \) and \( y \), \( y \) and \( z \) and \( z \) and \( x \), only appears through the coefficients. In other words, the unknowns \( y \) and \( z \) and their derivatives only appear in the coefficients of equation (3.36a), whereas the unknowns \( z \) and \( x \) and their derivatives only appear in the coefficients of equation (3.36b), etc. By lagging the coefficients in Eq. (3.36) so that the coefficients are treated as known quantities, the three equations in Eq. (3.36) are automatically decoupled. So it is only necessary to solve the scalar equation (3.36a) for \( x \), the scalar equation (3.36b) for \( y \), and the scalar equation (3.36c) for \( z \), rather than the vector equation (3.36).

The subroutine ADI is used as the solution scheme. Thus, it is assumed that, at the \( m^{th} \) iteration level, the solution \( x^m \), \( y^m \), and \( z^m \) has been obtained. The next iteration cycle begins by re-computing the coefficients \( A_1^m, A_2^m, \) etc., in Eq. (3.36) from the present values of \( x^m, y^m, \) and \( z^m \). Realizing that the coefficients are the same for all three equations in Eq. (3.36), it is only necessary to evaluate them once. This re-evaluation of the coefficients not only restores the non-linearity, but also the coupling between \( x \), \( y \) and \( z \). Once the coefficients have been evaluated, the subroutine ADI is called to solve the unknown \( x^{m+1} \) in equation (3.36a). With the same
coefficients, Eq. (3.36b) is solved by another call to ADI. Then again Eq. (3.36c) is solved.

Once the values of \( x^{m+1}, y^{m+1} \) and \( z^{m+1} \) are computed, yet another iteration cycle may start by re-computing the coefficients. This procedure is repeated until another preset tolerance criterion

\[
\text{Max}_{i,j}(x_{i,j}^{m+1} - x_i^m) < \varepsilon_2
\]

is satisfied. It is recommended [68] that the inner tolerance \( \varepsilon_1 \) (within ADI) be at least one order smaller than the outer tolerance \( \varepsilon_2 \). For the 3-D models studied in the present research, \( \varepsilon_2 = 10^{-6} \).

Yet another strategy for saving computational resource is employed by noting that the coefficients \( P, Q \) and \( R \), of the first order terms are made up of two parts, namely, the derivatives of the parameters \( s, t \) and \( u \), and the metrics of the transformation \( \alpha_{11} \ldots \text{etc.} \).

\[
\begin{align*}
P &= \frac{1}{J^2} \left[ (\alpha_{11}^1 p_{11}^1 + \alpha_{22}^2 p_{22}^2 + \alpha_{33}^3 p_{33}^3) + 2(\alpha_{12}^1 p_{12}^1 + \alpha_{23}^2 p_{23}^2 + \alpha_{31}^3 p_{31}^3) \right] \\
Q &= \frac{1}{J^2} \left[ (\alpha_{11}^2 p_{11}^2 + \alpha_{22}^2 p_{22}^2 + \alpha_{33}^3 p_{33}^3) + 2(\alpha_{12}^2 p_{12}^2 + \alpha_{23}^2 p_{23}^2 + \alpha_{31}^3 p_{31}^3) \right] \\
R &= \frac{1}{J^2} \left[ (\alpha_{11}^3 p_{11}^3 + \alpha_{22}^3 p_{22}^3 + \alpha_{33}^3 p_{33}^3) + 2(\alpha_{12}^3 p_{12}^3 + \alpha_{23}^3 p_{23}^3 + \alpha_{31}^3 p_{31}^3) \right]
\end{align*}
\]

The quantities \( P_{11}^1 \ldots \text{etc.} \) are evaluated from the derivatives of the parameters \( s, t, u \), as described in Section 3.1.1. The quantities \( \alpha_{11} \ldots \text{etc.} \), are metrics relating the computational and physical domains. During the iterating process, only metrics need to be updated, whereas derivatives of parameters stay the same. Thus, it is beneficial to pull out the part for computing \( P_{11}^1 \ldots \text{etc.} \), which does not need to be updated, out of the iteration cycle. This strategy saves some CPU time in updating the coefficients \( P, Q \) and \( R \). The overall iteration process is illustrated by the flowchart shown in Figure 18.

For the time-varying grid system Eq. (3.28), casting the grid-transport equations into a "canonical form" yields
Applying the Douglass-Gunn marching scheme, Eq. (3.35), to this nonlinear problem, the scheme can be expressed as

\[
\begin{align*}
\delta_\rho + \delta_\mu - \frac{\mu^m_x \delta_x + \rho^m_x \delta_{xx}}{2}
&= u^* \\
\delta_\rho + \delta_\mu - \frac{\mu^m_y \delta_y + \rho^m_y \delta_{yy}}{2}
&= u^{**} - \left(\frac{\mu^m_y \delta_y + \rho^m_y \delta_{yy}}{2}\right)u^n, \\
\delta_\rho + \delta_\mu - \frac{\mu^m_z \delta_z + \rho^m_z \delta_{zz}}{2}
&= u^{n+1,m+1} - \left(\frac{\mu^m_z \delta_z + \rho^m_z \delta_{zz}}{2}\right)u^n,
\end{align*}
\]  

(3.38)

where \(u^n\) is the known solution at time instant \(n\), \(u^*\) and \(u^{**}\) are intermediate values in the Douglass-Gunn ADI scheme and \(u^{n+1,m+1}\) is the \((m+1)\)th approximation for \(u^{n+1}\), the solution at the \((n+1)\)th time instant. The coefficients \(\mu^m_x\), \(\rho^m_x\), etc. are evaluated from \(u^{n+1,m}\), the \(m\)th approximation of \(u^{n+1}\). As a good starting point, the value of \(u^n\) is used to approximate \(u^{n+1}\), i.e., \(u^{n+1,1} = u^n\). The scheme given by the Eq. (3.38) can be interpreted as follows: in marching from the physical time instant \(n\) to time instant \((n+1)\), there is a local iteration. The newly computed \(u^{n+1,m}\) will be used to update the coefficients \(\mu^m_x\), \(\rho^m_x\), etc., and the same marching process
repeated again (that is, from time step \( n \) to \( n+1 \)) until the coefficients achieve a converged value. This local iteration for time instant \( n+1 \) is essential for restoring the nonlinearity of the equations. Only when converged coefficients and solution \( u^{n+1,\infty} \) have converged can then the next marching step be started. Schematically, this process is illustrated in Figure 19.

For grid generation involving a moving boundary, the boundary grid distribution is then specified as a function of time. The value of \( s, \ t \) and \( u \) on the boundary as well as in the interior therefore needs to be updated at every physical time step. As a result, this process needs to be included in the time-marching loop, as shown in Figure 20.

This concludes the description of numerical scheme for solving the nonlinear grid transport equations, for both stationary and time-dependent cases. In the next section, special consideration for multi-rectangular or multi-box computational domains will be presented; it essentially consists of a modification of the Thomas Algorithm.
Evaluation of normalized arc length $s_{E_1}$ etc. on the edges from boundary-point distribution

Evaluation of $s$, $t$ and $u$ for points interior to the domain and on the bounding surface, from the "algebraic" transformation

Evaluation of derivatives of parameters $s_{\xi\xi}$ etc. and then the quantities $P_{ij}^1$, ...etc. via Eq. (3.21)

Initial/Updated $x^n$, $y^n$ and $z^n$

Evaluate coefficient $\alpha^{1\prime}$ etc. via

Evaluate coefficient $P, Q, R$ via Eq.

Call ADI for solution of Eq. (3.26a) for unknown $x^{n+1}$

Call ADI for solution of Eq. (3.26b) for unknown $y^{n+1}$

Call ADI for solution of Eq. (3.26c) for unknown $z^{n+1}$

Evaluate discrepancy $\delta = \max_{i,j}(x^{n+1} - x^n)$

No

$\delta \leq \varepsilon_2$ ?

Yes

Save Data $x^{n+1}$, $y^{n+1}$, $z^{n+1}$

Exit

Figure 18. Flowchart for Stationary Grid Generation.
Input boundary-point distribution

Evaluation of normalized arc length $s_{E1}$, etc. on the edges from boundary-points distribution

Evaluation of $s$, $t$, and $u$ for point interior to the domain and on the bounding surface from the "algebraic" transformation

Evaluation of derivatives of parameter $s_{E}$, etc. and then the quantities $P_{i1}$, etc. via Eq. (3.21)

Initial $x^n$, $y^n$, and $z^n$

Approximate $x^{n+1,1} = x^n$, $y^{n+1,1} = y^n$, $z^{n+1,1} = z^n$

$m = 1$

Evaluate coefficient $\alpha_{11}$, etc. via Eq. (3.25) and using $x^{n+1,m}$, $y^{n+1,m}$, $z^{n+1,m}$

Evaluate coefficient $P$, $Q$, $R$ via Eq. (3.27) and using $x^{n+1,m}$, $y^{n+1,m}$, $z^{n+1,m}$

Go through Douglass-Gunn ADI marching process for variable $x^{n+1,m+1}$

Go through Douglass-Gunn ADI marching process for variable $y^{n+1,m+1}$

Go through Douglas-Gunn ADI marching process for variable $z^{n+1,m+1}$

Evaluate discrepancy $\delta = \max_{i,j} (x^{n+1,m+1} - x^{n+1,m})$

Figure 19. Flowchart for Time-Dependent Grid Generation.
Input boundary-point distribution

Evaluation of normalized arc length \( s_{E1} \ldots \text{etc.} \) on the edges from boundary-point distribution

Evaluation of \( s, t \) and \( u \) for point interior to the domain and on the bounding surface from the "algebraic" transformation (3.15)

Evaluation of derivatives of parameter \( s_{\xi} \ldots \text{etc.} \) and then the quantities \( P_{11} \ldots \text{etc.} \) via Eq. (3.21)

Approximate \( x^{n+1,1}=x^n, y^{n+1,1}=y^n, z^{n+1,1}=z^n \)

\( m=1 \)

Evaluate coefficient \( \alpha_{11} \ldots \text{etc.} \) via Eq.(3.25) and using \( x^{n+1,m}, y^{n+1,m}, z^{n+1,m} \)

Evaluate coefficient \( P, Q, R \) via Eq. (3.27) and using \( x^{n+1,m}, y^{n+1,m}, z^{n+1,m} \)

Go through Douglass-Gunn ADI marching process for variable \( x^{n+1,m+1} \)

Go through Douglass-Gunn ADI marching process for variable \( y^{n+1,m+1} \)

Go through Douglass-Gunn ADI marching process for variable \( z^{n+1,m+1} \)

Evaluate discrepancy \( \delta = \text{Max}_i \text{Max}_j (x^{n+1,m+1}_{i,j} - x^{n+1,m}_{i,j}) \)

Save Data \( x^{n+1}, y^{n+1}, z^{n+1} \)

\( \delta \leq \varepsilon_3 \) ?

\( n \geq n_{t_{\text{max}}} \) ?

Exit

Figure 20. Flowchart for Grid Generation with Time-Dependent Boundary.
3.3.3 Thomas Algorithm with Blanking

Solving equation (5) involves solution of a tridiagonal system in three mutually orthogonal directions in the computational domain. To take into account the inclusion of the IBLANK array which blanks out certain portions of the domain, it is only necessary to consider the problem in one direction; the remaining two directions follow exactly the same procedure.

Consider the following tridiagonal equation: \( a_i \phi_{i-1} + b_i \phi_i + c_i \phi_{i+1} = d_i \)
in three, separate, one-dimension domains (as shown in Figure 21).

| Domain 1 | Blanked-out | Domain 2 | Blanked-out | Domain n |

Figure 21. 1-D Domain with Blank-Out

A straightforward scheme to handle this situation is to deal with each active (nonblank) domain one by one. In dealing with each domain, the standard Thomas algorithm is applied. The Thomas algorithm consists of two steps, each corresponding to an inversion process as if the tridiagonal system has been pre-LU factorized. The first step finds the coefficients in the upper triangular matrix (e's and f's), and is thus named the 'ef step'; the second steps finds the solution \( \phi_i \) based on back-substitution and is named the '\( \phi \)-step'. When applied to the domain with blanking, the steps may appear as follows:

```
Ef step for domain 1
\downarrow
\phi \text{ step for domain 1}
\downarrow
Ef step for domain 2
\downarrow
\phi \text{ step for domain 2}
\downarrow
.......... 
\downarrow
Ef step for domain n
\downarrow
\phi \text{ step for domain n}
```
When dealing with multi-dimensional problems, the upper and lower bound of the relaxation domain varies; in fact, it is a function of the index other than the relaxing index. Furthermore, for particular \( i \) and \( j \) values, there may be more than one corresponding pairs of \( k_{\text{lower}} \) and \( k_{\text{upper}} \), each pair corresponding to one unblanked segment. This renders the bookkeeping task very difficult and, hence, triggers the development of an alternative procedure to handle the problem.

Instead of dealing with the domain(s) one by one, it is possible to treat the whole domain of interest (with AND without blanking area) as one. By introducing an IBLANK array that records the blanking status of each grid point, one can perform the ef step and then the \( \phi \)-step over the whole domain without concern about the domain boundary index – that is, only the min and max index of the domain of interest is of concern. The key is that the procedure simply skips over those inactive points. As a result, the solution sequence can be displayed as follows.
The following piece of FORTRAN segment implemented this modified Thomas algorithm together with Dirichlet boundary conditions;

```
C ---- EF STEP -----
DO I=1,N-1
  IF ( IB(I) .EQ. 2 .AND. IB(I+1) .EQ. 1) THEN ! I is the minimum of a domain
    E(I) = 0.D00
    F(I) = PHI(I)
  ELSEIF ( IB(I) .EQ. 1) THEN ! I is in a domain
    BOT = A(I)*E(I-1)+B(I)
    E(I) = -C(I) /BOT
    F(I) = (D(I)-A(I)*F(I-1))/BOT
  END IF
END DO
C ---- PHI STEP -----
DO I=N-1,1,-1
  IF ( IB(I) .EQ. 1 ) THEN ! only active points need to be considered
    PHI(I) = E(I)*PHI(I+1)+F(I)
  END IF
```

Note:  
IBLANK = 1 for points inside the active parts  
IBLANK = 0 for points outside the active parts  
IBLANK = 2 for points on the boundary of active parts; this is used to inform the routine that a new domain is encountered.

This procedure differs from the traditional Thomas algorithm by an additional condition check. This check consists of only an integer operation rather than a floating-point operation. Therefore, the additional overhead on the resulting code is insignificant. This extension of the Thomas algorithm is one of the major innovations presented in this research.
CHAPTER 4

NUMERICAL SOLUTION OF FLOW EQUATIONS

4.1 Numerical Schemes for the Navier-Stokes Equations

In the context of finite-difference solution of the incompressible Navier-Stokes equations, there are broadly two families of schemes to choose from. The first, namely the primitive-variable formulation, utilizes the original form of the Navier-Stokes equations (Eq. 2.1). In the second family are vorticity-based schemes wherein variants of the Navier-Stokes equations are employed. These variants range from the vorticity-stream function formulation in 2D, the vorticity-vector potential formulation in 3D, to the vorticity-velocity formulation in 3D.

For two-dimensional or axisymmetric problems, the scheme utilizing the vorticity-stream function formulation is widely used [6]. In this method, the curl of the momentum equation is employed instead of the original momentum equation. The resulting equation, with vorticity as the dependent variable, also takes the form of a transport equation, and is, therefore, named the vorticity transport equation. The vorticity in these cases has only one component and, therefore, the vorticity-transport equation a scalar equation. Also, there is no explicit appearance of the pressure term in the vorticity-transport equation.

\[
\frac{\partial \omega}{\partial t} + \frac{\partial u \omega}{\partial x} + \frac{\partial v \omega}{\partial y} = \frac{1}{Re} \left( \frac{\partial^2 \omega}{\partial x^2} + \frac{\partial^2 \omega}{\partial y^2} \right)
\]  

(4.1)

On the other hand, the continuity equation is eliminated from the system by the introduction of the stream function as yet another dependent variable. The coupling of vorticity and stream function is brought in by substituting the defining expression for the stream function into the defining expression for vorticity, resulting in a Poisson equation for stream function with vorticity as the source term.
The solution scheme is to iterate and march between these two equations. The iteration process is needed for the solution of the Poisson equation for the stream function $\psi^n$, with $\omega^n$, the vorticity at time instant $n$, known. Marching occurs when the vorticity-transport equation is solved for $\omega^{n+1}$, once the values of $\psi^n$ (and hence $u^n$ and $v^n$) are inserted into the equation as coefficients. For axisymmetric geometry, the Stokes stream function serves the same purposes.

In three dimensions, the vorticity-stream function formulation does not have a straightforward extension because two stream functions are required to define a 3-D flow. Instead of using the stream functions, a vector potential is defined such that

$$\vec{u} = \text{curl}\vec{\psi}$$

where $\vec{\psi}$ is the vector potential. By substituting this defining expression for $\vec{\psi}$ into the definition of the vorticity vector, one again obtains the Poisson equation for $\vec{\psi}$, with $\vec{\omega}$ on the right hand side. The governing equations in this formulation are thus the (vector) vorticity-transport equation for $\vec{\omega}$

$$\frac{\partial \vec{\omega}}{\partial t} + \nabla \cdot (\vec{u}\vec{\omega}) - (\vec{\omega} \cdot \nabla)\vec{u} - \frac{1}{Re} \nabla^2 \vec{\omega} = 0,$$

the (vector) Poisson equation for $\vec{\psi}$

$$\frac{\partial^2 \vec{\psi}}{\partial x^2} + \frac{\partial^2 \vec{\psi}}{\partial y^2} = -\vec{\omega},$$

and the defining equation for $\vec{\psi}$ (Eq 4.3)

The solution procedure is similar to that for the 2-D counterpart. Since each equation has three components, the numerical solution of this system is less economical as compared to methods using primitive variables. As a result, this method has not been used very often [39].
Yet another vorticity-based scheme in 3-D is the vorticity-velocity formulation [40]. The vector form of the vorticity-transport equation (Eq. 4.4) is retained in this scheme. By combining the definition of vorticity $\vec{\omega} = \text{curl}\vec{u}$ and the continuity equation $\text{div}\vec{u} = 0$, one may arrives at the following Poisson equations for velocity:

$$
\begin{align*}
\nabla^2 u &= \frac{\partial \omega_x}{\partial y} - \frac{\partial \omega_y}{\partial z}, \\
\nabla^2 v &= \frac{\partial \omega_y}{\partial x} - \frac{\partial \omega_x}{\partial z}, \\
\nabla^2 w &= \frac{\partial \omega_z}{\partial y} - \frac{\partial \omega_y}{\partial x}.
\end{align*}
$$

(4.6a,b,c)

Thus, the process of iteration for the Poisson equations (for the velocity components) and marching the vorticity-transport equation (for the vorticity components) may be carried on in the same manner as in the vorticity-stream function formulation. Again, there are six unknown variables in this formulation rather than four as found in the primitive-variable formulation. As expected, the use of such a scheme is less economical.

Generally speaking, the use of vorticity-based schemes in three-dimensional applications require more storage space (at least 6:4 ratio), and is less efficient as compared to schemes using primitive variables. Therefore, unless the vortex motion is of special interest, primitive-variable formulation is recommended. Furthermore, almost all practical turbulence models were developed using primitive variables.

The most common scheme for solution of the Navier-Stokes equations using primitive variables is the Marker and Cell method (MAC) introduced by Harlow and Welch [41]. In this method, the momentum equation, Eq. (2.5), is first temporally discretized, resulting in a first-order accurate approximation

$$
\frac{\vec{u}^{n+1} - \vec{u}^{n}}{\Delta t} + \nabla \cdot (\vec{u}^{n} \vec{u}^{n}) = -\nabla p^{n+1} + \frac{1}{\text{Re}} \nabla^2 \vec{u}^{n}.
$$

(4.7)
Then, the (spatial) divergence operator is applied to Eq. (4.7) to yield

\[
\frac{\nabla \cdot \vec{u}^{n+1} - \nabla \cdot \vec{u}^n}{\Delta t} + \nabla \cdot \left[ \nabla \cdot \left( \vec{u}^n \vec{u}^n \right) \right] = -\nabla^2 p^{n+1} + \nabla \cdot \left[ \frac{1}{\text{Re}} \nabla^2 \vec{u}^n \right].
\] (4.8)

Utilizing the continuity criterion that

\[
\nabla \cdot \vec{u}^{n+1} = 0,
\]

Eq. (4.8) my be reduced to

\[
\nabla^2 p^{n+1} = \nabla \cdot \left[ \frac{1}{\text{Re}} \nabla^2 \vec{u}^n - \nabla \cdot \left( \vec{u}^n \vec{u}^n \right) + \frac{\vec{u}^n}{\Delta t} \right].
\] (4.9)

Equation (4.9) is simply the Poisson equation for \(p^{n+1}\). With proper spatial discretization, a marching procedure may be devised for its solution. Equation (4.9) is first solved for \(p^{n+1}\) by an iterative method up to an acceptable accuracy, then eq. (4.7) is used to advance the velocity in time and \(u^{n+1}\) evaluated explicitly.

The projection method, also referred to as the fractional-step method by Chorin [42], is another popular method for numerical solution of the Navier-Stokes equations. This method splits the physical time step into two phases: a predictor phase and a corrector phase. In the predictor phase, the pressure term in the momentum equation, which bring in the pressure-velocity coupling, is temporarily dropped. The resulting equation is then temporally discretized explicitly:

\[
\frac{\vec{u}^* - \vec{u}^n}{\Delta t} + \nabla \cdot \left( \vec{u}^n \vec{u}^n \right) - \frac{1}{\text{Re}} \nabla^2 \vec{u}^n = 0.
\] (4.10a)

In the corrector phase, the pressure term, together with the continuity equation, is used to correct the predicted velocity field:

\[
\frac{\vec{u}^{n+1} - \vec{u}^*}{\Delta t} + \nabla p^{n+1} = 0,
\] (4.10b)
\[ \nabla \cdot \mathbf{u}^{n+1} = 0. \]  
\[ (4.10c) \]

It should note that, by adding Eqs. (4.10a) and (4.10b), the original discretized momentum equation (4.1) is recovered. Taking divergence of Eq.(4.10b) and making use of the continuity requirement, Eq. (4.10c), one arrives at the Poisson equation for pressure:

\[ \nabla^2 p^{n+1} = \frac{\nabla \cdot \mathbf{u}^*}{\Delta t}. \]  
\[ (4.11) \]

Thus, a marching procedure may also be devised as follows: The velocity field (\( \mathbf{u}^n \)) at time instant \( n \) is used to generate the predicted field (\( \mathbf{u}^* \)) explicitly via Eq.(4.10a); then, the Poisson equation (4.11) is solved for \( p^{n+1} \), and finally, \( \mathbf{u}^{n+1} \) is evaluated explicitly by Eq. (4.10b).

Yet another splitting method that has been developed focuses on the delta form for pressure. In this scheme, the momentum equation is temporally discretized as follows:

\[ \frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} + \nabla \cdot \left( \mathbf{u}^n \mathbf{u}^{n+1} \right) + \nabla \left( p^n + \delta p^n \right) - \frac{1}{Re} \nabla^2 \mathbf{u}^{n+1} = 0 \]  
\[ (4.12) \]

where \( p^{n+1} = p^n + \delta p^n \). To advance the solution, Eq.(4.12) is spit into predictor and corrector parts. The equation for the predictor (\( \mathbf{u}^* \)) is obtained from Eq.(4.12) by replacing the \((n+1)\) level by the predictor level (*), and dropping the \( \delta p^n \) term:

\[ \frac{\mathbf{u}^* - \mathbf{u}^n}{\Delta t} + \nabla \cdot \left( \mathbf{u}^n \mathbf{u}^* \right) + \nabla p^n - \frac{1}{Re} \nabla^2 \mathbf{u}^* = 0. \]  
\[ (4.13a) \]

The predicted velocity field \( \mathbf{u}^* \) is later corrected by the remaining part of momentum equation and the continuity requirement:

\[ \frac{\mathbf{u}^{n+1} - \mathbf{u}^*}{\Delta t} + \nabla (\delta p^n) = 0, \]  
\[ (4.13b) \]

\[ \nabla \cdot \mathbf{u}^{n+1} = 0. \]  
\[ (4.13c) \]
Again by adding Eqs.(4.13a) and Eqs.(4.13b), Eq. (4.12) is restored. Note, however, the predicted velocity \( \bar{\mathbf{u}}^* \) must be evaluated implicitly from Eq.(4.13a) rather than explicitly as in Eq.(4.10a). Taking divergence of Eq.(4.13b) and utilizing the continuity requirement, Eq.(4.13c), it is concluded that

\[
\nabla^2 (\delta p^n) = \frac{\nabla \cdot \bar{\mathbf{u}}^*}{\Delta t}.
\]

Consequently, a marching can again be designed. The velocity and pressure fields at time instant \( n \) (\( \bar{\mathbf{u}}^n, p^n \)) are used to generate the prediction field \( \bar{\mathbf{u}}^* \) implicitly via Eq.(4.13a); then, the Poisson equation (4.14) is solved for \( \delta p^n \) and \( p^{n+1} \), and finally, \( \bar{\mathbf{u}}^{n+1} \) is evaluated explicitly by equation (4.13b).

In the above description, no particular spatial discretization is specifically implied. In fact, application of different spatial discretization schemes (finite volume, finite difference on colocated grid, staggered grid, etc.) to any one of the schemes cited above may result in different schemes and, hence, be referred to by different names. For example, the aforementioned splitting method in the context of a finite-difference method may comprise one of the members of the SIMPLE-family schemes [43] when finite-volume discretization on a staggered grid is utilized. Patankar’s SIMPLE scheme (Semi-Implicit Method for Pressure-Linked Equations) is also very popular in industrial applications. Other variants in this family such as SIMPLEC (SIMPLE-Consistent), SIMPLER (SIMPLE-Revised) or PISO (Pressure Implicit with Splitting Operators) may be viewed in a similar manner.

In the present research, the method of artificial compressibility is employed. This method also uses a primitive-variable formulation. It was first introduced independently by Vladimirova et al. [44] and Chorin [45] and later extended by Kwak et al. [46] and Rogers et al. [47, 48]. This scheme will be discussed in detail in the next two sections.
4.2 Method of Artificial Compressibility

The method of artificial compressibility is one of the numerical schemes that has been used in a wide range of industrial applications. Since its introduction, this scheme has been extended from the simple Cartesian coordinate system, to a generalized curvilinear coordinate system, and from steady-state solutions to time-accurate simulation. It has been used to simulate the hot gas flow in the space shuttle [49], to analyse the mass diffusion and convection of smoke in the spacetab [50], and in the design of airplane ventilation system [51] and modeling the flow field in an artificial heart device [52].

As mentioned in Section 2.1, one of the difficulties inherent in the Navier-Stokes equations is incomplete coupling, i.e., absence of the pressure term in the continuity equation. The spirit of the method of artificial compressibility is to add an "artificial coupling" between pressure and velocity in the continuity equation. When solving the steady-state Navier-Stokes equations in a pseudo-transient setup, only the terminal solution is of interest, and all of the transient solutions are intermediate solutions during the iteration process and have no physical meaning. Thus, although adding an "artificial coupling" in the continuity equation perturbs the original equation set, its net effect is restricted to only the transient phase, which will be eventually discarded. In the long run, when the solution no longer changes with (pseudo) time, the governing equations will be restored to the original steady-state set, and, therefore, the solution would approach the steady-state solution.

Peyret [37] has illustrated this method using an explicit marching scheme on a staggered grid system. By using the staggered grid system, no implementation of pressure boundary condition is necessary. In addition, the velocity boundary condition is of Dirichlet type; hence,
setting up the boundary condition is very straightforward. In the example provided, the perturbed continuity equation takes the form

\[ \frac{\partial p}{\partial \tau} + \beta \nabla \cdot \vec{u} = 0 \]  

(4.15)

where \( \tau \) is the pseudo time and \( \beta \) is a constant used to drive the divergence of velocity to zero.

This equation has no physical meaning before the steady state \( \frac{\partial}{\partial \tau} = 0 \) is reached. The parameter \( \beta \) needs to be carefully picked to ensure convergence, and unfortunately, that is problem-dependent. Though general guidelines exist for choosing the value of \( \beta \), a trial-and-error procedure is still the most practical way to obtain the optimum value of \( \beta \) for a given problem.

The term "artificial compressibility" was introduced because Eq (4.15) could have been derived from the Navier-Stokes equations for a compressible fluid for which the equation of state would be

\[ p = \beta \rho . \]  

(4.16)

With Eq.(4.15) replacing the continuity equation, the system consisting of Eq.(2.1) (with \( t \) replaced by \( \tau \)) and Eq.(4.15) form a hyperbolic-parabolic system of equations, and thus any marching procedure, e.g., Explicit BTCS (Backward Time Central Space) may be applied until a steady-state solution is reached. Peyret's illustration is subject to a limit set by numerical stability, since an explicit marching scheme is employed.

Fletcher [38] utilized the method of artificial compressibility on a similar staggered grid system, and devised an implicit scheme for the marching procedure. In this work, the Beam and Warming type of approximate factorization (AF) method is also employed to split the multi-dimensional calculation into a series of one-dimensional computations.
In an incompressible fluid, a disturbance in the flow field is propagated throughout the domain at a speed of infinite magnitude. This may be seen from equation Eqs.(4.15) and (4.16). By letting $\beta$ approach infinity, Eq. (4.16) approaches the incompressible condition

$$\rho = \text{constant}$$

while Eq.(4.15) approaches the continuity equation for an incompressible fluid:

$$\nabla \cdot \vec{u} = 0$$

In fact, the quantity $\sqrt{\beta}$ may be interpreted as the "artificial sound speed", i.e., the speed at which a perturbation is propagated. In a numerical calculation, the pseudo-time step $\Delta \tau$ and the artificial compressibility parameter $\beta$ take finite values. Therefore, by introducing a finite artificial compressibility parameter in the governing equation, the artificial speed is reduced from the infinite magnitude to a finite value, and the pressure field, originally affected instantaneously by a disturbance, now has a time lag in responding to the pressure fluctuation. As a rule of thumb, the artificial sound speed $\sqrt{\beta}$ should be less than or equal to the minimum convective speed in the field. This criterion places a severe restriction on the choice of $\Delta \tau$ and $\beta$ values. The convective speed of the fluid is highly problem-dependent, so is the choice of $\Delta \tau$ and $\beta$ pairs. An optimal combination of $\Delta \tau$ and $\beta$ may result in a scheme that converges faster. However, this optimal combination needs to be found on a trial-and-error basis for a given problem.

Kwak [48] extended the idea of artificial compressibility to unsteady flow problems by creating a pseudo-time axis for each real time instant. Subiteration (marching along the pseudo-time axis) is performed until the flow field is divergence free for each real time step. In this work, a co-located grid system is employed. Other features such as spatial upwind differencing and temporal implicit descretization are also implemented. The resulting algebraic linear system may be solved by either a line relaxation method or the GMRES (Generalized Minimum
The next section will provide more details about this work and its implementation, i.e., – the INS3D flow solver developed at NASA Ames Research Center.

Obtaining a flow field that is truly divergence-free, i.e. numerically convergent to machine zero, is difficult. However, it has been reported [25] that a convergence criterion of \( \text{div}_{\text{max}} = 10^{-4} \) may be sufficient to yield satisfactory results for most problems.

### 4.3 Implementation – INS3D Flow Solver

#### 4.3.1 The Iteration Process

The INS3D flow solver is a CFD software for the numerical solution of Incompressible Navier-Stokes equations in 3D. It was developed in the late 80's by Kwak and Rogers [47] at NASA Ames Research Center. Continuous improvements have been made during early 90's, and the software has been matured into a widely used engineering analysis tool. This section will provide insight into this software. Steady-state flow simulation is considered first.

Applying the concept of artificial compressibility, Eq.(4.15), to the incompressible continuity equation written in generalized coordinates, Eq.(2.8), yields the equation:

\[
\frac{\partial \bar{p}}{\partial \tau} = -\beta \left[ \frac{\partial}{\partial \xi} \left( \frac{U}{J} \right) + \frac{\partial}{\partial \eta} \left( \frac{V}{J} \right) + \frac{\partial}{\partial \zeta} \left( \frac{W}{J} \right) \right].
\]  
(4.17)

The steady-state momentum equations (2.9) are also perturbed to include a pseudo-time derivative of velocity:

\[
\frac{\partial \bar{u}}{\partial \tau} = -\frac{\partial}{\partial \xi} (\bar{e} - \bar{e}_v) - \frac{\partial}{\partial \eta} (\bar{f} - \bar{f}_v) - \frac{\partial}{\partial \zeta} (\bar{g} - \bar{g}_v).
\]  
(4.18)

Eqs. (4.17) and (4.18) may be combined, to be written as

\[
\frac{\partial \bar{D}}{\partial \tau} = -\frac{\partial}{\partial \xi} (\bar{E} - \bar{E}_v) - \frac{\partial}{\partial \eta} (\bar{F} - \bar{F}_v) - \frac{\partial}{\partial \zeta} (\bar{G} - \bar{G}_v) = -\bar{R}
\]  
(4.19)

where
\[
\mathbf{D} = \begin{bmatrix}
\mathbf{p} \\
\mathbf{u} \\
\mathbf{v} \\
\mathbf{w}
\end{bmatrix}
\quad \mathbf{E} = \begin{bmatrix}
\beta^U \\
\beta^V \\
\beta^W
\end{bmatrix}
\quad \mathbf{F} = \begin{bmatrix}
\beta^U \\
\beta^V \\
\beta^W
\end{bmatrix}
\quad \mathbf{G} = \begin{bmatrix}
\beta^U \\
\beta^V \\
\beta^W
\end{bmatrix}
\]

and \( \mathbf{R} \) is the residual vector.

Performing a backward differencing, the resulting implicit finite difference equation (FDE) reads

\[
\frac{\mathbf{D}^{m+1} - \mathbf{D}^m}{\Delta \tau} = -\mathbf{R}^{m+1}
\]

whose RHS is a nonlinear combination of velocity components and their spatial derivatives.

Evaluating Eq.(4.20) at a general node \((p,q,r)\) and replacing \( \mathbf{R}^{m+1}_{p,q,r} \) by its Taylor's series expansion gives rise to

\[
\frac{\mathbf{D}^{m+1}_{p,q,r} - \mathbf{D}^m_{p,q,r}}{\Delta \tau} = -\mathbf{R}^m_{p,q,r} - \sum_{j,k,l} \left( \frac{\partial \mathbf{R}^m_{p,q,r}}{\partial \mathbf{D}^j_{j,k,l}} \right) \left( \mathbf{D}^{m+1}_{j,k,l} - \mathbf{D}^m_{j,k,l} \right)
\]

which may then be cast in the form

\[
\sum_{j,k,l} \left\{ \frac{\delta_{(p,q,r),(j,k,l)}}{\Delta \tau} + \left( \frac{\partial \mathbf{R}^m_{p,q,r}}{\partial \mathbf{D}^j_{j,k,l}} \right) \right\} \left( \mathbf{D}^{m+1}_{j,k,l} - \mathbf{D}^m_{j,k,l} \right) = -\mathbf{R}^m_{p,q,r}
\]

where \( \delta_{(p,q,r),(j,k,l)} \) is a generalized Kronekar delta defined to be

\[
\delta_{(p,q,r),(j,k,l)} = 1 \quad \text{when} \quad (p,q,r) = (j,k,l)
\]

\[
= 0 \quad \text{otherwise}
\]

Equation (4.21) may be used to form a linear system of algebraic equations

\[ A\mathbf{x} = \mathbf{b} \]
where A is a square matrix of dimensions \((JM \times KM \times LM)\) by \((JM \times KM \times LM)\), formed by the array of the quantities

\[
\frac{\delta_{(p,q,r,j,k,l)}}{\Delta t} + \left( \frac{\partial \mathbf{R}_{p,q,r}}{\partial \mathbf{D}_{j,k,l}} \right)^m,
\]

\(x\) is the unknown column vector of \((JM \times KM \times LM)\) elements in the array

\[
(\tilde{D}^{m+1} - \tilde{D}^m)_{j,k,l},
\]

and \(b\) is the column vector of \((JM \times KM \times LM)\) element, generated from \((-\mathbf{R}^m_{p,q,r})\).

Equation (4.21) is used to march along the pseudo-time axis until the residual vector \(\mathbf{R}\) converges.

Next, the time-accurate procedure is presented. For time-accurate flow field simulation, the Navier-Stokes equations for unsteady flow are written as

\[
\begin{align*}
\frac{\partial \mathbf{u}}{\partial t} &= -\mathbf{r} \quad \text{(momentum equation)} \\
\nabla \cdot \mathbf{u} &= 0 \quad \text{(continuity equation)}
\end{align*}
\]

where the vector \(\mathbf{r}\) is the same as for steady-state equation and defined in the right hand side of Eq.(2.9)

Performing a second order backward differencing on the momentum equation results in

\[
\frac{3\mathbf{u}^{n+1} - 4\mathbf{u}^n + \mathbf{u}^{n-1}}{2\Delta t} = -\mathbf{r}^{n+1}
\]

(4.22)

where the superscript \(n\) denotes the real physical time \(t = n\Delta t\). To solve Eq.(4.22) for the unknown velocity vector \(\mathbf{u}^{n+1}\) at the \((n+1)\)th real time level, a pseudo-time axis is created for this physical-time instant, and a pseudo-time marching (iteration) is performed along this axis to
obtain a divergence-free flow field for this instant. The pseudo-time is denoted by the superscript \( m \). Adding a pseudo-time derivative of the velocity vector to Eq.(4.22) gives rise to

\[
\frac{\partial \tilde{u}^{n+1}}{\partial \tau} + \frac{3\tilde{u}^{n+1} - 4\tilde{u}^n + \tilde{u}^{n-1}}{2\Delta t} + \tilde{r}^{n+1} = 0.
\] (4.23)

Combining Eq.(4.23) and the perturbed continuity equation (4.17), which is also valid for unsteady cases, yields

\[
\begin{pmatrix}
0 \\
\end{pmatrix} + \frac{1}{2\Delta t} \begin{pmatrix}
3\tilde{u}^{n+1} - 4\tilde{u}^n + \tilde{u}^{n-1} \\
\end{pmatrix} = -\tilde{R}^{n+1}
\] (4.24)

where \( \tilde{D} \) and \( \tilde{R} \) are the same as those in equation Eq.(4.19).

Performing a backward differencing in the pseudo-time axis, the resulting implicit finite-difference equation (FDE) reads

\[
\frac{\tilde{D}^{n+1,m+1} - \tilde{D}^{n+1,m}}{\Delta \tau} + \frac{1}{2\Delta t} \begin{pmatrix}
0 \\
\end{pmatrix} = -\tilde{R}^{n+1,m+1}
\]

which may be re-arranged as

\[
I_u \left( \tilde{D}^{n+1,m+1} - \tilde{D}^{n+1,m} \right) = -\tilde{R}^{n+1,m+1} - \frac{I_m}{\Delta t} \left( 1.5\tilde{D}^{n+1,m} - 2\tilde{D}^n + 0.5\tilde{D}^{n-1} \right)
\] (4.25)

where

\[
I_u = \text{diag} \left[ \begin{array}{c}
\frac{1}{\Delta \tau}, \frac{1.5}{\Delta \tau}, \frac{1}{\Delta \tau}, \frac{1.5}{\Delta \tau}, \frac{1}{\Delta \tau}, \frac{1.5}{\Delta \tau}
\end{array} \right]
\]

and

\[
I_m = \text{diag}[0,1,1,1].
\]

As in the steady-state case, Eq.(4.25) is evaluated at node \((p,q,r)\) and, replacing \( \tilde{R}^{n+1,m+1}_{p,q,r} \) by its Taylor's series expansion gives rise to
\[
I_{tr} \left( \vec{D}_{p,q,r}^{n+1,m+1} - \vec{D}_{p,q,r}^{n+1,m} \right) = - \left[ \vec{R}_{p,q,r}^{n+1,m} + \sum_{j,k,l} \frac{\partial \vec{R}_{p,q,r}}{\partial \vec{D}_{j,k,l}} \left( \vec{D}_{j,k,l}^{n+1,m+1} - \vec{D}_{j,k,l}^{n+1,m} \right) \right] - \frac{I_m}{\Delta t} \left( 1.5\vec{D}_{p,q,r}^{n+1,m} - 2\vec{D}_{p,q,r}^{n} + 0.5\vec{D}_{p,q,r}^{n-1} \right)
\]

and finally, this equation may be cast into the standard form

\[
\sum_{j,k,l} \left[ \left( \vec{D}_{j,k,l}^{n+1,m+1} - \vec{D}_{j,k,l}^{n+1,m} \right) \left( \vec{D}_{j,k,l}^{n+1,m+1} - \vec{D}_{j,k,l}^{n+1,m} \right) \right] = - \vec{R}_{p,q,r}^{n+1,m} - \frac{I_m}{\Delta t} \left( 1.5\vec{D}_{p,q,r}^{n+1,m} - 2\vec{D}_{p,q,r}^{n} + 0.5\vec{D}_{p,q,r}^{n-1} \right)
\]

Equation (4.26) may also be used to form a linear system of algebraic equations

\[Ax = b\]

where \(A\) is a square matrix of \((JM \times KM \times LM)\) by \((JM \times KM \times LM)\) formed by the array of quantities

\[
\left[ I_{tr} \delta_{(p,q,r),(j,k,l)} + \left( \frac{\partial \vec{R}_{p,q,r}}{\partial \vec{D}_{j,k,l}} \right) \right]^{n+1,m},
\]

\(x\) is the unknown column vector of \((JM \times KM \times LM)\) elements in the array

\[
\left( \vec{D}_{j,k,l}^{n+1,m+1} - \vec{D}_{j,k,l}^{n+1,m} \right),
\]

and \(b\) is the column vector of \((JM \times KM \times LM)\) elements generated from

\[- \vec{R}_{p,q,r}^{n+1,m} - \frac{I_m}{\Delta t} \left( 1.5\vec{D}_{p,q,r}^{n+1,m} - 2\vec{D}_{p,q,r}^{n} + 0.5\vec{D}_{p,q,r}^{n-1} \right).\]

Equation (4.26) is used to march along the pseudo-time axis until the residual vector \(\vec{R}\) converge for that physical time instant.

### 4.3.2 Upwind Differencing

In both Eqs. (4.21) and (4.26), no specific spatial discretization scheme is referred. In INS3D, second-order derivatives that correspond to viscous effects are discretized via central difference approximations. However, for first-order derivatives that correspond to the convective nature of the flows, their discretisation is not such a simple matter.
One distinct feature pertinent to convective terms such as \( u \frac{\partial u}{\partial x} \) is that they consist of first-order derivatives. If a symmetric central-difference approximation (of order 2) is used to represent the first-order derivative term, non-physical oscillations may build up in the solution if the flow field is convection-dominated. This is related to a dispersion-like influence in the truncation error. Furthermore, the nonlinear characteristics of the convective terms enhance these spurious oscillations. As a rule of thumb, the order of the approximating finite-difference expression should not be greater than the order of the original derivatives to be approximated.

On the other hand, if an asymmetric one-sided difference expression is employed in representing the first-order derivative terms (simple form of upwind difference [12]), the accuracy of these solutions is sacrificed, typically one order less than that of the central difference, although the smoothness of the solution is usually improved. In severe cases, the error introduced can be as large in magnitude as the physical term being modeled and, thus, totally ruins the solution.

Schemes that utilize the concept of artificial viscosity have been developed for addressing this problem. If a central difference is used to approximate the first-order derivative term, along with inclusion of proper dissipation terms which have the same order of magnitude as the local truncation error, the resulting algebraic equation may lead to smooth solutions. However, the amount of dissipation that is applied uniformly to all grid points cannot be decided beforehand and, therefore, requires some trial and error on user’s part.

Another remedy to address this problem is the use of the Flux-Splitting form of the Upwind Difference scheme (FSUD) rather than the simple form or "donor cell" form [52] of the scheme. With this scheme, dissipation is added to the system naturally, and second order accuracy may be retained. This dissipation will automatically suppress any oscillation caused by the nonlinear convective flux. In addition, the upwind-differenced flux vector contributes to
terms on the main diagonal of the Jacobian of the residual, whereas central-differenced flux does not. This helps to make the implicit scheme nearly diagonally dominant and, therefore, greatly improves the convergence properties and robustness of the scheme. Hence, although the use of FSUD scheme demands more CPU time than the central-difference scheme, the speed-up in convergence may result in considerable savings in the overall computational resources required.

In the present application, the FSUD scheme is derived using a one-dimensional consideration. For three-dimensional problems, it is applied to each coordinate direction individually. The spirit of the following derivation is to cast the 1-D governing equation into its characteristic form, and then setup the differencing stencil such that it accounts for the direction of the artificial sound wave propagation. The 1-D hyperbolic system of the conservation laws reads

\[
\frac{\partial \tilde{q}}{\partial t} + \frac{\partial \tilde{f}}{\partial x} = 0
\]

where \( \tilde{q} \) is the vector of unknown variables, and \( \tilde{f} \) is the flux vector. The spatial discretization of Eq.(4.27) gives

\[
\left( \frac{\partial \tilde{q}}{\partial t} \right)_j + \left[ \frac{\tilde{f}_{j+\frac{1}{2}} - \tilde{f}_{j-\frac{1}{2}}}{\Delta x} \right] = 0
\]  

(4.28)

where

\[
\tilde{f}_{j+\frac{1}{2}} = \frac{1}{2} \left[ \tilde{f}(\tilde{q}_{j+1}) + \tilde{f}(\tilde{q}_j) - \delta_{j+\frac{1}{2}} \right]
\]

and \( \tilde{f}_{j-\frac{1}{2}} \) may be defined similarly.

The key to the flux-splitting upwind differencing scheme is the term \( \delta_{j+\frac{1}{2}} \). If \( \delta_{j+\frac{1}{2}} = \tilde{0} \)
(and similarly, if $\tilde{\phi}_{j+\frac{1}{2}} = 0$), then Eq.(4.28) is reduced to the symmetric central differencing mentioned above. In first-order FSUD scheme, the "smoothing flux" $\tilde{\phi}$ is defined as

$$\tilde{\phi}_{j+\frac{1}{2}} = \Delta\tilde{f}_{j+\frac{1}{2}}^{+} - \Delta\tilde{f}_{j+\frac{1}{2}}^{-}$$

(4.30)

where

$$\Delta\tilde{f}_{j+\frac{1}{2}}^{\pm} = A^{\pm}(\tilde{q})\Delta\tilde{q}_{j+\frac{1}{2}}$$

(4.31)

are the "flux differences" across the traveling waves. In case of a third-order FSUD, the "smoothing" flux $\phi$ has the form

$$\tilde{\phi}_{j+\frac{1}{2}} = -\frac{1}{3}\left(\Delta\tilde{f}_{j+\frac{1}{2}}^{+} - \Delta\tilde{f}_{j+\frac{1}{2}}^{-} + \Delta\tilde{f}_{j+\frac{1}{2}}^{+} - \Delta\tilde{f}_{j+\frac{1}{2}}^{-}\right).$$

FSUD of even higher order is possible; for example, the fifth-order smoothing flux reads

$$\tilde{\phi}_{j+\frac{1}{2}} = -\frac{1}{30}\left(-2\Delta\tilde{f}_{j+\frac{1}{2}}^{+} + 11\Delta\tilde{f}_{j+\frac{1}{2}}^{-} - 6\Delta\tilde{f}_{j+\frac{1}{2}}^{+} - 3\Delta\tilde{f}_{j+\frac{1}{2}}^{-} + 2\Delta\tilde{f}_{j+\frac{1}{2}}^{+} - 11\Delta\tilde{f}_{j+\frac{1}{2}}^{-} + 6\Delta\tilde{f}_{j+\frac{1}{2}}^{-} + 3\Delta\tilde{f}_{j+\frac{1}{2}}^{-}\right).$$

The "flux difference", as defined by Eq.(4.31), requires computation of the matrices $A^{+}$ and $A^{-}$, where $A$ is the Jacobian of the flux vector $A = \frac{\partial\tilde{f}}{\partial\tilde{q}}$. To accomplish this, the diagonal matrix of positive eigenvalue $A^{+}$ is computed first, then $A^{+}$ is obtained by performing a similarity transformation, $A^{+} = XA^{+}X^{-1}$. Because $A^{+} + A^{-} = A$, $A^{-}$ is simply determined by subtraction, i.e., $A^{-} = A - A^{+}$. Also, in Eq.(4.31), $\tilde{q} = \frac{1}{2}(\tilde{q}_{j+1} + \tilde{q}_{j})$ and $\Delta\tilde{q}_{j+\frac{1}{2}}^{\pm} = \tilde{q}_{j+1} - \tilde{q}_{j}$.

In INS3D, the flux vector $\tilde{f}$ is of the form (with $i$ taking the value 1, 2 and 3, representing application of the FSUD scheme to $\xi$, $\eta$, $\zeta$ direction, respectively)
\[ \tilde{E}_i = \frac{1}{J} \begin{pmatrix} \beta \gamma \\ k, u + k, v + u \gamma \\ k, v + k, u + v \gamma \\ k, w + k, p + w \gamma \end{pmatrix} \]

where

\[ k, x = \frac{\partial \xi_i}{\partial x}, \quad k, y = \frac{\partial \xi_i}{\partial y}, \quad k, z = \frac{\partial \xi_i}{\partial z}, \quad k, t = \frac{\partial \xi_i}{\partial t}, \]

and \( Q = k, u + k, v + k, w \) is the contravariant component of velocity vector.

The Jacobian matrix

\[
A_i = \frac{\partial \tilde{E}_i}{\partial D} = \begin{bmatrix} 0 & k, \beta & k, \beta & k, \beta \\ k, x & k, x u + Q + k, i & k, y u & k, z u \\ k, y & k, y v & k, y v + Q + k, i & k, z v \\ k, z & k, z w & k, z w + Q + k, i & \end{bmatrix}
\]

is diagonalized as \( A_i = X_i \Lambda_i X_i^{-1} \), where

\[ \Lambda_i = \text{diag}[\lambda_1, \lambda_2, \lambda_3, \lambda_4], \]

with the eigenvalues found to be

\[ \lambda_1 = Q + k, i, \quad \lambda_2 = Q + k, i, \quad \lambda_3 = Q + \frac{k, i}{2} + c, \quad \lambda_4 = Q + \frac{k, i}{2} - c, \]

and \( c = \sqrt{\left(Q + \frac{k, i}{2}\right)^2 + \beta(k, x^2 + k, y^2 + k, z^2)} \) is the scaled artificial sound speed.

The following matrix of the right eigenvector \( X_i \) has been used in the similarity transformation.

\[
X_i = \begin{bmatrix} \beta \left(c - \frac{k, i}{2}\right) & -\beta \left(c + \frac{k, i}{2}\right) \\ u \lambda_3 + \beta k, x & u \lambda_4 + \beta k, x \\ v \lambda_3 + \beta k, y & v \lambda_4 + \beta k, y \\ w \lambda_3 + \beta k, z & w \lambda_4 + \beta k, z \end{bmatrix}
\]

and its inverse is given as:
\[ X_i^{-1} = \begin{bmatrix}
  a_{11} & a_{12} & a_{13} & a_{14} \\
  a_{21} & a_{22} & a_{23} & a_{24} \\
  a_{31} & a_{32} & a_{33} & a_{34} \\
  a_{41} & a_{42} & a_{43} & a_{44} 
\end{bmatrix} \]

where

\[ a_{11} = x_k (k_y w - k_y w) + y_k (k_x w - k_x u) + z_k (k_x v - k_x v), \]

\[ a_{21} = x_k (k_y w - k_x v) + y_k (k_x u - k_x w) + z_k (k_x v - k_x u), \]

\[ a_{31} = -\lambda_4 \left( \frac{c + k_t}{2} \right), \quad a_{41} = -\lambda_3 \left( \frac{c - k_t}{2} \right), \]

\[ a_{12} = y_k (\lambda_i w + \beta k_x) - z_k (\lambda_i v + \beta k_y), \]

\[ a_{22} = -y_k (\lambda_i w + \beta k_x) + z_k (\lambda_i v + \beta k_y), \]

\[ a_{32} = \frac{k_x \left( \frac{c + k_t}{2} \right)}{2c}, \quad a_{42} = \frac{k_x \left( \frac{c - k_t}{2} \right)}{2c}, \]

\[ a_{13} = z_k (\lambda_i u + \beta k_x) - x_k (\lambda_i w + \beta k_x), \]

\[ a_{23} = -z_k (\lambda_i u + \beta k_x) + x_k (\lambda_i w + \beta k_x), \]

\[ a_{33} = \frac{k_y \left( \frac{c + k_t}{2} \right)}{2c}, \quad a_{43} = \frac{k_y \left( \frac{c - k_t}{2} \right)}{2c}, \]

\[ a_{14} = x_k (\lambda_i v + \beta k_y) - y_k (\lambda_i u + \beta k_x), \]

\[ a_{24} = -x_k (\lambda_i v + \beta k_y) + y_k (\lambda_i u + \beta k_x), \]

\[ a_{34} = \frac{k_z \left( \frac{c + k_t}{2} \right)}{2c}, \quad a_{44} = \frac{k_z \left( \frac{c - k_t}{2} \right)}{2c} \]

in which
\[ x_k = \frac{\partial x}{\partial \xi_{i+1}} \quad \text{and} \quad x_{kk} = \frac{\partial x}{\partial \xi_{i+2}} \]

\[ \xi_{i+1} = \eta, \zeta \text{ or } \xi \text{ for } i=1,2,3, \text{ respectively,} \]

\[ \xi_{i+2} = \zeta, \xi \text{ or } \eta \text{ for } i=1,2,3, \text{ respectively.} \]

For grid points near the boundary where the full stencil is not available, a reduction in accuracy is inevitable when using the FSUD scheme. However, it is not necessary to reduce the accuracy to first order. If the following flux is employed, near-second-order accuracy is maintained:

\[
\tilde{f}_{j+1/2} = \frac{1}{2} \left[ f(q_{j+1}) + f(q_j) - \varepsilon \phi_{j+1/2} \right]. \tag{4.32}
\]

Note that Eq.(4.32) is a weighted average between the symmetric central-differencing expression (\(\varepsilon=0\)) and the first-order FSUD expression (\(\varepsilon=1\)). In INS3D, a value of \(\varepsilon=0.01\) is used to ensure that the solution is oscillation-free at the boundary.

### 4.3.3 Implicit Scheme

To conduct the numerical solution for the system of Eqs.(4.21) or (4.26), the residual vector and its Jacobian matrix needs to be evaluated first. This is done by performing spatial discretisation of Eqs.(4.21) and (4.26). As described in detail in the previous section, the spatial discretization is accomplished by replacing the second-order derivatives (viscous fluxes) by central difference approximations, and first-order derivatives (convective flux) by the flux-split upwind difference approximation that was shown in the second term in Eq.(4.28). The residual term, as defined by Eq.(4.19) and evaluated at node (j,k,l), becomes

\[
R_{j,k,l} = \frac{\tilde{E}_{j+1/2,k,l} - \tilde{E}_{j-1/2,k,l}}{\Delta \xi} + \frac{\tilde{F}_{j,k+1/2,l} - \tilde{F}_{j,k-1/2,l}}{\Delta \eta} + \frac{\tilde{G}_{j,k,l+1/2} - \tilde{G}_{j,k,l-1/2}}{\Delta \zeta} - \frac{\left( \tilde{E}_v \right)_{j+1,k,l} - \left( \tilde{E}_v \right)_{j-1,k,l}}{2\Delta \xi} - \frac{\left( \tilde{F}_v \right)_{j,k+1,l} - \left( \tilde{F}_v \right)_{j,k-1,l}}{2\Delta \eta} - \frac{\left( \tilde{G}_v \right)_{j,k,l+1} - \left( \tilde{G}_v \right)_{j,k,l-1}}{2\Delta \zeta}. \tag{4.33}
\]
Replacing the convective flux by the first-order FSUD representation similar to Eqs.(4.29) and (4.30) results in

\[
\begin{align*}
R_{j,k,l} &= \frac{1}{2} \left[ \tilde{E}_{j+1,k,l} - \tilde{E}_{j-1,k,l} + \tilde{F}_{j,k+1,l} - \tilde{F}_{j,k+1,l} + \tilde{G}_{j,k+1,l} - \tilde{G}_{j,k-1,l} \right] \\
&+ \frac{1}{2} \left[ -\Delta\tilde{E}^{+}_{j+\frac{1}{2},k,l} + \Delta\tilde{E}^{-}_{j-\frac{1}{2},k,l} + \Delta\tilde{E}^{+}_{j-\frac{1}{2},k,l} - \Delta\tilde{E}^{-}_{j+\frac{1}{2},k,l} \right] \\
&+ \frac{1}{2} \left[ -\Delta\tilde{F}^{+}_{j,k+\frac{1}{2},l} + \Delta\tilde{F}^{-}_{j,k-\frac{1}{2},l} + \Delta\tilde{F}^{+}_{j,k-\frac{1}{2},l} - \Delta\tilde{F}^{-}_{j,k+\frac{1}{2},l} \right] \\
&+ \frac{1}{2} \left[ -\Delta\tilde{G}^{+}_{j,k+\frac{1}{2},l} + \Delta\tilde{G}^{-}_{j,k-\frac{1}{2},l} + \Delta\tilde{G}^{+}_{j,k-\frac{1}{2},l} - \Delta\tilde{G}^{-}_{j,k+\frac{1}{2},l} \right] \\
&+ \frac{1}{2} \left[ (\tilde{E}_v)_{j+1,k,l} + (\tilde{E}_v)_{j-1,k,l} + (\tilde{F}_v)_{j,k+1,l} + (\tilde{F}_v)_{j,k-1,l} + (\tilde{G}_v)_{j,k+1,l} + (\tilde{G}_v)_{j,k-1,l} \right] \\
\end{align*}
\]

in which the grid spacing \(\Delta\xi\), \(\Delta\eta\), \(\Delta\zeta\) in the computational domain is chosen to be unity.

To form the Jacobian matrix with elements \(\frac{\partial R_{p,q,r}}{\partial D_{j,k,l}}\), derivatives are taken on both sides of Eq.(4.34). However, even though the resulting Jacobian is a banded matrix, its numerical evaluation is still too expensive in practice. Therefore, an approximate Jacobian of the residual matrix is used, as originally proposed by Barth [54]. This approximation is derived by retaining only the orthogonal mesh terms in the exact Jacobian, and this greatly simplifies the expression.

With the right hand side determined from Eq.(4.34) and the left hand side approximated as described in the last paragraph, Eq. (4.21) or (4.26) can then be solved. Many schemes are possible for solving the resulting algebraic equations numerically. One of these is the line-relaxation scheme. In this method, a "relaxation line", say a line in j direction for certain k and l values, is selected. All the terms on the left which contains the unknowns on this line (e.g., unknowns with subscript \((j-1,k,l), (j,k,l)\) and \((j+1,k,l)\)) stay on the LHS, whereas those terms that are off this line are moved to the right-hand side of the equation. The right-hand side of the equation is evaluated using the latest-known values for the \(\Delta D\), where \(\Delta D = D^{m+1} - D^m\) for
steady-state equations, or \( \Delta \tilde{D} = \tilde{D}^{n,m+1} - \tilde{D}^{n,m} \) for time-accurate simulations. The resulting equation is a tridiagonal matrix of 4×4 blocks, and can be solved by the block version of the Thomas Algorithm.

The above "relaxation" procedure is repeated for each of the other indeces (e.g., \( k \) and \( l \) index). A systematic way is to "sweep" over the computational domain for, for example, \( k=1 \) through \( k_{\text{max}} \) and for \( l=1 \) through \( l_{\text{max}} \). Because Eq.(4.21) or Eq.(4.26) is used to march along the pseudo-axis until the solution converges, an exact solution for the algebraic equation (4.21) or (4.26) is not necessary. Thus, if more "sweeps" are performed in relaxing the algebraic equations, it is likely to require less marching steps along the pseudo-time axis; conversely, if only a few sweeps are performed for solving the algebraic equation (4.21), then more pseudo-time steps are to be anticipated. Therefore, the optimal number of sweeps for each of the directions that gives the best overall convergence performance requires trial and error runs. Figure 20 illustrates the effects of the number of sweeps in each direction on the number of pseudo-time steps required, for a sample problem of flow in a square duct with a 90° bend after Rogers [55].
This concludes the description of the numerical method used to solve the flow equations.

In this particular example, the setup with njswp=3/nkswp=3/nlswp=3 leads to the least number of iterations (red line). However, this case also gives the second highest CPU time required. The case corresponding to the least CPU time required is the case with njswp=2/nkswp=1/nlswp=1.

This concludes the description of the numerical method used to solve the flow equations.
The next chapter is dedicated to the validation of the overall problem solution procedure - from setting up the grid to obtaining the flow solution.
CHAPTER 5

SIMULATION OF FLOW IN VASCULAR GRAFT ANASTOMOSES

5.1 Introduction

The construction of an end-to-side arterial graft is often used by surgeons to bypass flow over a diseased area. However, there are long-term complications. Intimal hyperplasia and plaque formation are usually observed in the floor and toe area (Figure 21). This is believed to be related to the hemodynamic conditions in the flow field. Use of computational technique can help to quantify these hemodynamic conditions, such as low wall shear, in relation to the origin of the abnormality. Furthermore, the computational technique presented here can also be used in the design of new anastomoses with minimum adverse hemodynamic effect.

Steinman [69] performed two-dimensional simulation of the flow field in an end-to-side anastomosis model, and found that the pathogenesis of distal anastomotic intimal hyperplasia is correlated with the wall shear stress in the flow field. Lei [70], in his three-dimensional simulation, provided evidence that wall shear stress gradient (WSSG) may also be a key factor that triggers abnormal growth of arterial tissue and, hence, intimal hyperplasia.

The present research has studied this flow configuration in detail. Experimental and numerical results are also available for this flow, especially for the corresponding 2-D configuration, and can be used to verify and validate the present research. For example, White [18] provided an in-vitro experimental visualization of the iliofemoral graft, whereas Taylor [26] performed a FEM simulation of the in-vivo flow field in the graft. The purpose of the
verification is two-folded, namely, numerical H-H grid generation scheme and numerical flow solution procedure.

Figure 21 and Table 1 summarize the key geometric parameters used by White [18] and Taylor [26]. Model 1 corresponds to the in-vitro visualization model of White [18] which is in turn a 1:7.5 scale-up model of an in-vivo experimental setup. Model 2 duplicates the key geometric features of an in-vivo model used by Taylor. Note that the terminologies used in the following discussion are also labeled. Since the application of the flow solver INS3D requires the input of a non-dimensional geometric model, the inlet diameter is chosen to be the reference length. The resulting non-dimensional model is presented in Table 2.

![Figure 21. Graft Geometry.](image)

**Table 1 Model Geometry**

<table>
<thead>
<tr>
<th></th>
<th>Model 1 (in vitro)</th>
<th>Model 2 (in vivo)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A (Inlet Diameter)</td>
<td>31.5 mm</td>
<td>5.6 mm</td>
</tr>
<tr>
<td>B (Proximal Outlet Diameter)</td>
<td>31.5 mm</td>
<td>3.5 mm</td>
</tr>
<tr>
<td>C (Distal Diameter)</td>
<td>27.0 mm</td>
<td>3.5 mm</td>
</tr>
<tr>
<td>D (Hood Length)</td>
<td>135.5 mm</td>
<td>17.1 mm</td>
</tr>
<tr>
<td>E (Sinus Diameter)</td>
<td>44.5 mm</td>
<td>3.5 mm</td>
</tr>
</tbody>
</table>
Table 2  Non-Dimensional Model Geometry

<table>
<thead>
<tr>
<th></th>
<th>Model 1 (in vitro)</th>
<th>Model 2 (in vivo)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1.000</td>
<td>1.000</td>
</tr>
<tr>
<td>B</td>
<td>1.000</td>
<td>0.625</td>
</tr>
<tr>
<td>C</td>
<td>0.857</td>
<td>0.625</td>
</tr>
<tr>
<td>D</td>
<td>4.238</td>
<td>3.054</td>
</tr>
<tr>
<td>E</td>
<td>1.413</td>
<td>1.000</td>
</tr>
</tbody>
</table>

The grid generation procedure may be thought of as a mapping which transforms a region with uniform grid in the computational domain onto the original region in the physical domain. The multi-box concept described in Chapter 3 is utilized. For example, corresponding to model 1, the multi-box computational domain is initialized as shown in Figure 22. Then a mapping is established by solving the inverted Poisson equations with Dirichlet boundary conditions. The resulting grid system in the physical domain is the image of the rectangular grid system in the computational domain, and is shown in Figure 23. Note the highlight of this grid system is its H-H topology, as shown in Figs. 24 and 25, respectively.

Figure 22. Computational Domain for Model 1 (Grid Size: 175 x 61 x 125)
Figure 23. Physical Domain (Model 1).

Figure 24. H-H Topology; this view shows that both inlet plane and proximal outlet plane have H grid.

Figure 25. H-H Topology; this view shows how the two ‘blocks’ join in the physical domain.
Likewise, the grid system for model 2 is generated similarly and is shown in Fig. 26.

Figure 26. Model 2.
(Grid Size :175 x 61 x 125 )

For comparison purposes, the flow conditions investigated are specified to correspond to those in the related literature. The fluid employed in White’s experimental work was a mixture of water and glycerine in a volumetric ratio of approximately 42:58. This mixture gives a kinematic viscosity of 0.078 cm²/s. Taylor’s simulation of the in-vivo condition utilized the fluid properties of blood with kinematic viscosity of 0.035 cm²/s. For the present simulation, the fully developed parabolic profile is specified at the graft inlet plane. For model 1, the average velocity is chosen such that the Reynolds numbers are 1000 and 200 for the two cases examined. The Reynolds number for the simulation with Model 2 is 208, corresponding to the flow visualization
by White. The boundary condition at the proximal outlet is a static pressure condition such that the flow division ratio meets the prescribed value. This requires a trial and error routine to determine the proper pressure at the proximal outflow. The flow condition at the distal outlet is assumed to be zero pressure. Table 3 shows the various steady flow conditions simulated in this work.

<table>
<thead>
<tr>
<th>Case</th>
<th>Model</th>
<th>Re</th>
<th>Flow Division (Proximal:Distal)</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1000</td>
<td>0 : 100</td>
<td>White [18], in vitro</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>200</td>
<td>0 : 100</td>
<td>White [18], in vitro</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>200</td>
<td>50 : 50</td>
<td>White [18], in vitro</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>200</td>
<td>100 : 0</td>
<td>White [18], in vitro</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>208</td>
<td>20 : 80</td>
<td>Taylor [26], in vivo</td>
</tr>
</tbody>
</table>

5.2 Results and Discussion

The experimental work of White [18] is used extensively in this section. The investigation examines the effect of different factors such as Reynolds number, flow-division ratio, and hood length, and demonstrates the possible correlation between low wall shear and localization of intimal hyperplasia in a graft anastomosis.

Figures 27a and 27b present the simulated velocity vectors and streamline patterns in the central symmetric plane for Case 1. These results compare well qualitatively with the photographs by White [18] (Fig. 27c). Note the upward velocity components downstream of the graft. From the consideration of conservation of mass, it will then follow that there is a downward velocity component near the side wall of the graft. This indicates that a pair of helical structures is formed, and that is a unique feature of this flow field (Fig. 27d).
Figure 27a. Velocity Vectors for Case 1: Model 1, Re=1000, Flow-Division Ratio=0:100.

Figure 27b. Streamline Pattern for Case 1: Model 1, Re=1000, Flow-Division Ratio=0:100.

Figure 27c. Experimental Results for Case 1: Model 1, Re=1000, Flow-Division Ratio=0:100.
White [18] reports that the measured stagnation point along the floor of the anastomosis occurs at a location 71% of the total hood length for this case. The computed value is 72%, and agrees well with the experimental data.

Besides the velocity components, Figs. 27e and 27f present the surface pressure contours and surface vorticity contours, respectively. There is no flow in the proximal conduit, thus pressure remains at a constant value in this segment of the geometry.
The distinct lines on the hood area and on the distal conduit are where the surface singularity is. The computed vorticity values at these points may not be realistic, and thus should be ignored. Since surface vorticity is proportional to the surface shear stress, the surface vorticity contours also provide information about the surface shear stress. From Fig. 27e, it is clear that the proximal side of the graft experiences less shear stress than that the distal side of the graft.

The goal of studying Case 2 is to investigate the Reynolds number effect. For this case, the results for Re = 200 are compared with those for Case 1 where the Reynolds number is 1000. Since the same geometry is employed and, in both cases, the average inflow velocity was chosen as the reference velocity, varying the Reynolds number is equivalent to varying the viscosity of the fluid. Therefore, the case with Re = 200 represents a more ‘sticky’ fluid than the case with Re = 1000. As expected, the upward velocity component in the central symmetric plane and the downward velocity component on the side wall are diminished for Case 2. Much of the energy associated with these flow components is dissipated by the viscosity effect.
Comparing Figs. 28b and 28c may raise the question that these streamline patterns are not similar. This dilemma can, at least partially, be attributed to the way the photograph is generated in the experiment. The PIV (Particle Image Velocimetry) technique is employed for flow visualization in the experiment. When photographed, the camera sees not just one specific plane,
but also flow activities on planes other than the plane the user requested. This bias is mainly caused by optical scattering. Hence, readers should be cautious when interpreting these images.

The experimental results indicate that the stagnation location is at 63% of the total hood length, while the current research results in 56% for the value. The deviation is within 10% and, therefore, considered acceptable.

For Case 1, the maximum pressure is around 1.5 (dimensionless), whereas it is 4.0 for Case 2. Again, this is due to the higher viscosity, and these results are consistent with the above explanation.

Figure 28d. Surface Pressure Contours for Case 2: Model 1, Re=200, Flow-Division Ratio=0:100.

Figure 28e. Surface Vorticity Contours for Case 2: Model 1, Re=200, Flow-Division Ratio=0:100.
Case 3 depicts the effect of flow-division ratio on the flow field. When the proximal flow ratio is increased from 0% to 50%, the stagnation point moves distally. At 50:50 ratio, the computed stagnation location is 80% of the hood length, whereas the experimental result provides 86%. The deviation is again about 7.5%, and therefore considered acceptable. Note also that both experiment and numerical simulation reveal that, by going as the flow-division ratio increase from 0:100 to 50:50, the stagnation location moves distally by approximately 23%.

Figure 29a. Velocity Vectors for Case 3: Model 1, Re=200, Flow-Division Ratio=50:50.

Figure 29b. Streamline Pattern for Case 3: Model 1, Re=200, Flow-Division Ratio=50:50.

Figure 29c. Experimental Results for Case 3: Model 1, Re=200, Flow-Division Ratio=50:50.
The surface pressure contours (Fig. 29d) indicate that there is non-zero pressure drop in both conduits for Case 3, and the maximum pressure required to ‘pump’ the flow is reduced to 1.2, as compared to 4.0 for Case 2.

![Surface Pressure Contours](image)

**Figure 29d. Surface Pressure Contours for Case 3: Model 1, Re=200, Flow-Division Ratio=50:50.**

It is important to observe that this flow-division ratio (50:50) is the closest to the real physiological situation. For this case, the surface vorticity contours DO indicate that there is a low wall-shear stress area on the hood and toe of the graft – a CFD result that confirms the clinically observed phenomenon.
Figure 30a-e present results for Case 4 which corresponds to an occluded distal outflow condition. At the first glance at Figs. 30b and 30c, it is easy to get the impression that one of the major flow structures is missing in the simulation. However, as stated earlier, the PIV technique may show flow activity on areas other than the plane the user requested. If three-dimensional streamlines, rather than just the two-dimensional streamlines on the central symmetric plane, are plotted from the CFD simulation, one may obtain a totally different view. The situation is confirmed via the simulation result shown in Figs. 30d. The major flow structure in the experimental photograph, thought to be a clockwise vortex, is indeed a counterclockwise ‘cross over’ flow structure, and is a truly three dimensional phenomenon. This view cannot be achieved by simply a slicing plane in the 3-D flow field; rather, it has to be done by projecting all the 3D streamlines onto the central symmetric plane. In this context, the numerical simulation really helped in the interpretation of the experimental data, and in revealing this 3-D phenomenon.
Figure 30a. Velocity Vectors for Case 4: Model 1, Re=200, Flow-Division Ratio=100:0.

Figure 30b. Streamline Pattern for Case 4: Model 1, Re=200, Flow-Division Ratio=100:0.

Figure 30c. Experimental Results for Case 4: Model 1, Re=200, Flow-Division Ratio=100:0.
The flow field in this case is rather complicated. Figure 30e illustrates four different views of the streamlines emitted from a horizontal rake near the center of the inflow conduit. Depending on the distance between the ‘seeding point’ and the central point of the inflow plane, the streamlines may follow a totally different path; some simply make a 180 degree turn, others exhibits a ‘crossover‘ trajectories. Also notice that, in the proximal conduit, a pair of vortices is formed, similar to Case 1 where a pair of vortices was formed in the distal conduit. Recall that, for Case 2, the flow division ratio was 0:100, whereas it is 100:0 for Case 4.

The computed stagnation point is located at 74% of the hood length, whereas the experimental value is 70%, and this deviation is, again, acceptable.
Figs. 30f and 30g show the surface pressure contours and surface vorticity contours, respectively. As expected, the major pressure loss is in the proximal conduit.
Figure 30f. Surface Pressure Contours for Case 4: Model 1, Re=200, Flow-Division Ratio=100:0.

Figure 30g. Surface Vorticity Contours for Case 4: Model 1, Re=200, Flow-Division Ratio=100:0.
Case 5 utilizes model 2 for the simulation. The Reynolds number is 208, and the flow division ratio is 20:80, corresponding to the FEM simulation by Taylor [26]. The major difference between model 1 and 2 is that the diameter of inflow conduit is larger than that of the outflow conduit (both proximal and distal). As a result, the flow accelerates in the graft (Figure 31a).

In Figure 31c, a helical structure near the end of the hood area is apparent, especially for streamlines originating from points located on the upper lateral portion of the inflow plane. Notice that the direction of rotation is different from that of case 1, indicating that the underlying physics is different in these two flows.

Figure 31a. Velocity Vectors for Case 5: Model 2, Re=208, Flow-Division Ratio=20:80.

Figure 31b. Streamline Pattern for Case 5: Model 2, Re=208, Flow-Division Ratio=20:80.

Figure 31c. 3D Streamline Pattern for Case 5.
Figs. 31d and 31e compare the velocity profiles in the symmetry plane and in a plane transverse to it, with the corresponding data from Taylor [26] (FEA) and Loth [71] (LDA). In most cases, the agreement is quite good.

Figure 31d. Velocity Profiles in the Symmetry Plane for Case 5.
The largest deviation is 15% and occurs at section A, Figure 31e, corresponding to the starting portion of the proximal conduit. This deviation may be attributed to the variation in geometry details of the heel area (Fig. 21). In the experiment, the heel is rounded, whereas in the present simulation, the heel is right-angled.

Figure 31e. Velocity Profiles Transverse to the Symmetry Plane for Case 5.
Figs. 31f and 31g show the corresponding surface pressure contours and surface vorticity contours.

Figure 31f. Surface Pressure Contours for Case 5: Model 2, Re=208, Flow-Division Ratio=20:80.

Figure 31g. Surface Vorticity Contours for Case 5: Model 2, Re=208, Flow-Division Ratio=20:80.
This concludes the presentation of the simulation results. In summary, the capability of CFD is demonstrated via the example of the flow field in the graft geometry. The results provide more insight into the flow field (in particular, Case 4), as well as validate a clinic observation (Case 3).

In the next chapter, application to the geometry of a heart chamber is described.
CHAPTER 6

FLOW FIELD IN LEFT ATRIUM

6.1 Review of Cardiac Events

A pump is a device that accepts fluid at low pressure and transfers it to a region where the pressure is high. In fact, what the human heart performs is just that. The cardiac cycle can be visualized in Figure 32 where the ventricular pressure (red line) is plotted versus time. The left ventricle accepts fluid from the left atrium at a low pressure of 10-20 mmHg, and performs work on the fluid to elevate its pressure to as high as 120 mmHg.

Figure 32. Cardiac Cycle.
(Data from http://human.physiol.arizona.edu/TEST/ANSWER/CVSupplements/Wig_PV.GIF)
Figure 33 reveals the relation between pressure and volume in this cycle. Also indicated are the landmarks during the cycle, as explained below.

The cycle can be divided into the following 7 phases, and can be seen from the following images. These images are taken from a trans-esophageal echocardiography examination. The left ventricle chamber is outlined in red. Note the proximity of the mitral valve to the aortic valve.
1. **Isovolumetric Relaxation** (A-B in figure, sustained about 80 ms)

When the LV pressure falls below the pressure in the aorta, the aortic valve closed. While LV pressure is still higher than that of LA, the mitral valve remains closed. Since both valves upstream and downstream of the ventricle are closed, its blood contents remain constant. The ventricular muscles relax during this period, and the LV pressure reduces sharply.

2. **Rapid Filling** (B-B₁ in figure, sustained about 110 ms)

As soon as the LV pressure falls below the LA pressure, the mitral valve opens, and rapid ventricular filling begins. Blood coming from the atrium quickly fills the ventricles, and pressure in both chambers declines sharply. Blood flow from the aorta to the peripheral arteries continues, and thus the aortic pressure decreases gradually.
3. **Diastasis or reduced filling** (B₁-B₂ in figure, sustained about 190 ms)

Flow across the mitral valves is greatly diminished.

The pressure in both LV and LA rises gradually.

![Diastasis Image]

4. **Atrial Contraction** (B₂-C in figure, sustained about 100ms)

At the end of the diastolic phase is the atrial contraction. This increases the pressure gradient between LA and LV by 5 mmHg and, hence, elevates the pressure and volume of the ventricle slightly. Atrial contraction results in the second burst in ventricle filling, and contributes 20-30% of the total filling. At fast heart rates, atrial contraction is very important because the phase of rapid filling and diastasis is reduced.

![Atrial Contraction Image]
5. *Isovolumetric Contraction* (C-D in figure, sustained about 50ms)

As long as the ventricular muscles contract, the generated pressure closes the mitral valve so that both valves upstream and downstream of ventricle are closed. Thus, the LV pressure rises quickly, from about 25mmHg to 80mmHg in 50ms.

6. *Rapid Ejection* (D-D₁ in figure, sustained about 90ms)

As soon as the pressure in the left ventricle exceeds the pressure in the aorta, the aortic valve opens and blood flows rapidly from the ventricle into the aorta. This is associated with a sharp decrease in ventricular volume. The force that the ventricle exerts is so high that the pressure in the ventricle and root of the aorta rises to 120mmHg. The amount of blood ejected depends on contractility and preload. During this period, the pressure in the pulmonary vein also increases, and the filling of atrium begins.

7. *Decreased ejection* (D₁-A in figure, sustained about 130ms)

In this period, the aortic pressure may be slightly greater than the ventricular pressure but the blood flow is still forward. This is due to the momentum of the fluid in balancing the adverse pressure gradient. The atrium is still filling due to the difference between pressure in the pulmonary vein and in the atrium.
6.2 Some Physiological Conditions Associated with Left Atrium

This section will briefly describe some of the physiological conditions associated with atrial function. These disorders or diseases are all related to the dynamics of the prevailing fluid motion, and are of concern to biological fluid dynamicists.

When a blood clot is formed near the mitral valve or the mitral leaflet is thickened due to prior occurrence of rheumatic fever, flow through the mitral valve is restricted. This Mitral Stenosis (MS) may affect the normal diastolic function of the left ventricle and, hence, reduce the stroke volume. In response, the body then generates a natural compensation by increasing left atrium volume, or by producing a higher atrial contraction pressure. However, this compensation may result in other problems upstream of the atrium, i.e., the pulmonary system. For example, the major symptom of mitral stenosis includes dyspnea (shortness of breath) due to the fact that the air passage was congested by the elevated pressure level in the lung. In severe cases, the valve may have to be widened by a procedure called valvotomy, or the valve replaced if repair is not feasible. This condition is illustrated in Figure 34.

![Normal Valve Mechanisms](image1)

![Mitral Valve Stenosis](image2)

**Figure 34. Mitral Valve Stenosis.**
Instead of restricted flow during ventricular diastole, a mitral valve could be leaky during ventricular systole (Mitral Regurgitation). This condition is most often caused by rheumatic heart disease (inflammatory disease), a type of degeneration of the valve, dysfunction of the muscles that control the closing of the valve, or rupture of the valve chords. If the portion of the heart that supports the position of the valve is disrupted, a heart attack may follow as a result. In acute cases, symptoms may be sudden and severe. Patients may go into heart failure, and urgent therapy is necessary. There are no medications that can help to heal the valves; therapy is directed toward relief of dyspnea and other related symptoms. Severe cases are most likely treated by surgical replacement rather than repair.

Yet another leaky valvular condition is known as Mitral Valve Prolapse (MVP). It is a deformity of the mitral valve that may prevent its leaflets from closing properly. One or both leaflets may be bulging, or the entire valve may be out of its normal position. Depending on the degree of the deformity, the prolapse can lead to mitral regurgitation. The disorder is believed to be primary hereditary. It is usually recognized by its characteristic clicks and murmurs that can

![Image of normal valve mechanisms and mitral regurgitation](image)

**Figure 35. Mitral Valve Regurgitation.**
be heard with a stethoscope. In some cases, MVP may lead to mitral insufficiency; so strenuous activities are to be restricted.

Besides valvular disorders, a defect on the atrial wall may cause other conditions called Atrial Septal Defect (ASD). The two upper chambers of the heart, the right atrium and left atrium, are separated by a "wall", called the ATRIAL SEPTUM. Sometimes, this "wall" is not complete. There is a hole in it. This hole is called an Atrial Septal Defect (ASD). In the normal heart, blood flowing in the right side of the heart (atrium and ventricle) is completely separated from the left heart by the atrial septum. When there is a hole in this "wall", blood from the left atrium at higher pressure flows through the hole into the right side where the pressure is low. Beside the normal amount of "impure" blood coming from the veins through the right atrium, the right ventricle (lower chamber) now receives more blood due to the extra blood coming into the right atrium through the hole in the atrial septum. This will increase the loading on the right ventricle and, as years go by, may result in heart failure. The increased volume of blood in the pulmonary circulation system may also result in pulmonary hypertension – an condition unfavorable to the lung. In the case of ASD, increase in blood volume in pulmonary circulation is accompanied by a reduction of blood volume in systemic circulation. The reduction in left ventricular stroke volume often induces a higher heart rate as compensation. This disturbance in normal rhythm of the heart may eventually develop into arrhythmia – an irregularity in the electrical events of the heart. Most doctors suggest a surgical procedure to close the hole to prevent further complications.

A particular form of arrhythmia called Atrial Fibrillation (AF) is characterized by the loss of synchrony between the atria and the ventricles. In general, AF is thought of as a storm of
electrical energy that travels in spinning wavelets across the left and right atria, making these upper chambers quiver or fibrillate at 300 to 600 times per minute, a frequency at least four times higher than the normal value, while the rhythm in other parts of the heart stays, more or less, the same. Many patients describe the irregular, often rapid pulsations of the heart in AF as an uncomfortable flapping sensation inside the chest, with a sudden and keen awareness of every heartbeat. This may be accompanied by shortness of breath, chest pain, profuse sweating, dizziness, syncope (passing out), exercise intolerance and extreme fatigue. During AF, the left atrium does not contract effectively and, hence, is not able to empty its contents efficiently. Sluggish blood flow may come about inside the atrium, and forms clots. One type of stroke (thromboembolic cerebral vascular accident, or CVA) occurs when a blood clot travels to the brain, and lodges in a vessel, causing the normal blood flow to stop, and the brain tissue to die from lack of oxygen. As a matter of fact, atrial fibrillation increases an individual's risk of stroke by 4 to 6 percent, and about 15 percent of stroke patients have atrial fibrillation before they experience a stroke. To prevent this kind of severe complication, an anticoagulant or blood thinner such as Coumadin is usually administered. However, its dosage is highly individualized, and must be carefully monitored to ensure safety. Other non-pharmacological therapy such as electrical cardioversion, ablations, maze procedure etc. are available for different kinds of atrial fibrillation.

In the next section, the model geometry used in the present simulation will be described. Section 6.4 is dedicated for the numerical simulation of the flow field in the left atrium.
6.3 The Atrium Model and Initial/Boundary Conditions

The atrium model employed in the present research is derived from various sources. The difficulty in obtaining the geometry data is noteworthy. For example, while stationary geometry data can be readily obtained via a non-beating heart using invasive technique, it is unfeasible to perform an invasive measurement on a live person to acquire "in vivo" the temporal variation data for chamber geometry. Whenever a non-invasive technique is not available for acquiring this kind of data, invasive measurements on animals instead of human beings is the only alternative, and the animal needs to be sacrificed.

In regard to the atrium geometry, both Lemmon [21] and Hoit [27] employed the approximation of an ellipsoid. The present work duplicates the stationary geometry data used by Lemmon,[21], and the key parameters are listed below:

- Left atrium long axis length = 3.9 cm
- Left atrium short axis length = 3.0 cm
- Mitral orifice diameter = 2.5 cm

Zacek [1] quoted another set of geometry data which is not quite consistent with Lemmon's data. In Zacek's data, the mitral orifice area is 18 cm² while Lemmon's data show it is 4.9 cm² (\(\frac{\pi}{4} \cdot 2.5^2 = 4.9\) cm²). Also presented in Zacek's paper is the lumped flow area of the pulmonary vein (11 cm²), which is not available in Lemmon's work.

The present work employs Lemmon's data but also incorporates the flow area ratio from Zacek's work. Based on these assumptions, the diameter for an individual pulmonary vein is found to be 0.977 cm (\(\sqrt[11]{\frac{4}{18}} \cdot 2.5\)).
With these dimensions available and defining the mitral orifice diameter as the reference length, the following dimensionless geometrical parameters may be computed:

\[ a^* = \frac{3.0}{2.5} = 0.6 \]

\[ b^* = \frac{3.9}{2.5} = 0.78 \]

\[ Dv = \sqrt{\frac{114}{18}} = 0.391 \]

For normal heart at normal condition, the cardiac output (volume of blood per unit of time) is averaged to

\[ Q = 5.6 \text{ liter/min} = 93.3 \text{ cm}^3/\text{sec}, \]

so the average velocity across mitral valve is

\[ U = \frac{Q}{A} = \frac{93.3 \text{ cm}^3/\text{sec}}{4.91 \text{ cm}^2} = 19 \text{ cm/sec}. \]

This velocity is set to be the reference velocity.

The Reynolds number can then be computed as

\[ \frac{\rho UL}{\mu} = \frac{(1 \frac{g}{cm^3})(19 \frac{cm}{sec})(2.5 \text{ cm})}{(3.5 \times 10^{-2} \frac{g}{cm \cdot s})} = 1357 \]

The model geometry is illustrated in Figure 36.
6.4 Flow Field in Left Atrium

In the previous section, the Reynolds number, based on the mitral orifice diameter, is computed to be 1357. However, this Reynolds number will result in unsteady flow in the chamber. To simulate the steady flow field in the atrium, qualitatively resembling the long-term behavior, a Reynold’s number of 500 is employed.

For flow visualization purposes, two cutting planes are used. P1 represents a symmetry plane passing through the long axis of the chamber as well as the central axis of the mitral orifice (refer to Figure 37a). The second plane, P2, also passes through the long axis of the chamber,
but makes a 30 degree angle with P1. Therefore, the “central axis” of two of the inflow conduits lies on P2, as shown in Figure 38a.

Figure 37a. Symmetric Cutting Plane (P1).

The simulated velocity vectors and streamline pattern are displayed in Figure 37b-c and Figure 39b-c. A distinct feature in Figure 37c is the interior stagnation point, where the four ‘jets’ coming from the four inflow conduit meet. Again, a three-dimensional streamline plot greatly helps in interpreting the two-dimensional streamline pattern, as shown in Figure 38.

Figure 37b. Velocity Vectors in Cutting Plane P1.
Figure 37c. 2D Streamlines in Cutting Plane P1.

Figure 38. 3D Streamlines.
On cutting plane P2, the flow pattern is very different from that in P1. In this view, the vortex-like flow structures resulting due to sudden enlargement of the flow passage are clearly seen. These structures are especially susceptible for the low wall shear stress and the consequence of blood clot formation. (Figure 40)
This concludes the flow field simulation in the left atrium. Chapter 7 provides a summary for the overall procedure and the results achieved and recommendations for further work.
CHAPTER 7

CONCLUSION AND RECOMMENDATIONS

7.1 Computational Accomplishments and Conclusion

In this research, a new grid generation technique is developed and implemented in a flow simulation. This technique enables one to perform grid generation for complex geometry using only a single computational zone. By employing a single zone and a blanking array, it is possible to analyze the flow field without zonal iteration, and therefore, with increased efficiency. Furthermore, the proposed scheme lays a foundation for a more general application of the flow adaptive grid generation technique. So far, flow-adaptive grid generation schemes are confined to application to grid system with single zone and simple computational domains only. When a multi-zone grid system (patched or overlaid) is utilized, such as those employing Chimera type schemes, there is a major issue to implement the scheme that allows grid points to move across the different blocks of the grid. The tracking and book-keeping of these grid movements across the artificial zone boundaries then becomes a difficult subject. By using the proposed scheme, the applicability of the flow-adaptive technique is greatly extended to a more general category of complex geometries, as all grid points always remain in a single zone topology.

The scheme is based on the composite transformation of an algebraic mapping and a mapping governed by the Laplace equation. The numerical scheme used for integration of the resulting governing equations is an extension of the traditional three-dimensional Douglass-Gunn scheme. Modifications to this scheme and enhancements are made so as to account for the multi-rectangular or multi-box computational domain. The corresponding numerical scheme to
accommodate this extension is adjusted accordingly, leading to the Thomas Algorithm with blanking.

Grids were generated for two model geometries using the proposed grid generation software. The graft model features one inflow conduit and two outflow conduits, while the left atrium (LA) model has four inflow conduits and one outflow conduit.

Flow simulation was performed using the research code INS3D, which employs the method of artificial compressibility. For the flow simulation inside the graft, the effect of Reynolds number and flow division ratio is examined. The Reynolds number effect is, as expected, demonstrated via the presence of a helical flow structure as well as the overall pressure drop. The flow-division ratio, on the other hand, alters the flow field in a way that moves the stagnation points. In particular, the case with 50:50 flow division ratio closely resemble to those observed clinically, and the highlighted low wall stress area on the hood and toe of the anastomosis strengthen the hypothesis on the formation of intimal hyperplasia. The complicated flow field demonstrated by the case with 100:0 division ratio, corresponding to a occluded distal arteries, demonstrated that three-dimensional numerical simulation of the flow field assisted in interpreting data from a PIV experimental session.

The steady-state simulation of flow field in the left atrium of the heart is yet another subject of interest. Although steady state simulation is not as realistic as time accurate simulation, it nevertheless gives information on the long term performance of the chamber. The simulation shows the existence of low wall shear region. Those low shear stress area in the chamber are area susceptible to blood clot formation. In fact, clinical evidences show that the cause of certain stroke is indeed cause by clot formed in the atrium and traveled through the arterial system and essentially lodged in the brain. Since this phenomenon is geometry-related
and there is no practical way to alter, common therapy for such conditions is to administered certain ‘blood thinner’ (Anticoagulation) to reduce the chance of blood clot formation.

7.2 Recommendations

Throughout this study, it is found that the multi-box scheme is very useful in handling geometries with multiple inflow and/or outflow, such as the graft (one inflow, 2 outflow ) and the left atrium chamber ( 4 inflow, 1 outflow ). The simulated Reynolds number range is fairly low (1-1000), so the effect of grid clustering near the wall is minor. However, for flows with higher Reynolds number, it is mandatory to incorporate greater clustering near the wall region. In fact, as a rule of thumb, one should properly resolve the boundary layer, which is of the order of \( \frac{1}{\sqrt{Re}} \). Past experience suggests that placing at least 5-8 points inside the laminar boundary layer are necessary. Due to the nature of the multi-box scheme, the coordinate surface aligned with one part of the boundary may be an interior coordinate surface in other regions. Therefore grid clustering near one of the solid walls may propagate into the interior of the domains regardless of whether grids clustering is needed, or not, in that interior region. Figure 41 illustrates this situation where the grid clustering near the wall of the inflow conduit essentially becomes internal clustering inside the chamber.
Another issue associated with this grid generation technique concerns the grid orthogonality on boundary. As mentioned in Chapter 3, for H grid topology, the singular points are always on the boundary, where the flow solution is not required. However, in the vicinity of this singular point, the near-singular behavior of the grid may depreciate the accuracy of the flow solution in this vicinity. In other words, it is not always feasible to generate grid system that is orthogonal, or even near orthogonal, to the boundary.

The above mentioned two issues merit further investigation. For the first issue, one of the possible avenues is to devise a way to quickly disperse the grid clustering at the junction between, for example, the inflow conduit and the chamber. This scheme needs to be robust enough to detect any of these junction scenarios, and perform adequate smoothing. Regarding the second issue, one may implement the orthogonality condition for boundary points a certain number of ‘cells’ away from the singular points.
Parallelization of the grid generation software is also a feasible extension, in particular, for the Douglass-Gunn ADI phase. Instead of sequentially ‘sweeping’ through all the directions other than the relaxation direction, simultaneous relaxation will speedup the overall grid generation process.

Finally, time-accurate flow simulation is certainly warranted for a deeper understanding of the flow physics inside the graft and the heart chamber. Higher Reynolds number, boundary movement and pressure-driven unsteadiness are all realistic phenomena in biological flows, and should be examined.
Reference


71. Loth, F., “Velocity and wall shear stress measurements inside a vascular graft model under steady and pulsatile flow conditions” Ph.D. Dissertation, Georgia Institute of Technology, Atlanta, GA, 1993