Parallel Anisotropic Block-Based Adaptive Mesh Refinement Finite-Volume Scheme

by

Jenmy Zimi Zhang

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

Copyright © 2011 by Jenmy Zimi Zhang
A novel anisotropic adaptive mesh refinement (AMR) technique is proposed and described. A block-based AMR approach is used which permits highly efficient and scalable implementations on parallel computer architectures and the use of multi-block, body-fitted computational grids for the treatment of complex geometries. However, rather than adopting the more usual isotropic approach to the refinement of the grid blocks, the proposed approach uses a binary hierarchical tree data structure that allows for anisotropic refinement of the grid blocks in each of the coordinate directions in an independent fashion. This allows for more efficient and accurate treatment of narrow layers, discontinuities, and/or shocks in the solutions which occur, for example, in the thin boundary and mixing layers of high-Reynolds-number viscous flows and in the regions of strong non-linear wave interactions of high-speed compressible flows with shocks. The anisotropic AMR technique is implemented within an existing finite-volume framework, which encompasses both explicit and implicit solution methods, and is capable of performing calculations with both second- and higher-order spatial accuracy. To clearly demonstrate the potential and feasibility of the proposed AMR technique, it is applied to the unsteady and steady-state solutions of both a model system, the advection diffusion equation, as well as the Euler equations governing compressible, inviscid, gaseous flows, both in two space dimensions.
I wish to take this opportunity to thank Professor Clinton P. T. Groth for introducing me to the field of computational fluid dynamics. I am also grateful for his many inspirations, great enthusiasms and insightful ideas, which have guided me through the first three years of my research in CFD.

I would also like to express my deepest gratitude towards my family and friends for their continuous support and encouragements. I would like to thank especially my mother, who has made a lot of sacrifices in helping me accomplish my academic and personal goals.

Furthermore, I want to thank all my colleagues in the computational fluid dynamics lab for their assistance throughout my research, especially Dr. Lucian Ivan for his help with the high-order code, Scott Northrup for his help with NKS and Dr. Pradeep Jha for answering even my smallest questions about LaTeX and git. I have greatly enjoyed the friendship with everyone in the lab, which has made my master’s studies a memorable experience.

Lastly, computational resources for performing all of the calculations reported herein were provided by the SciNet High Performance Computing Consortium at the University of Toronto and Computer/Calcul Canada through funding from the Canada Foundation for Innovation (CFI) and the Province of Ontario, Canada.

Jenny Zimi Zhang
Toronto, July 2011
Contents

1 Introduction 1
   1.1 Efficient and Accurate Analysis of Complex Flows . . . . . . . . . . . . . 1
   1.2 Parallel Adaptive Algorithms . . . . . . . . . . . . . . . . . . . . . . . 2
      1.2.1 Adaptive Mesh Refinement Techniques . . . . . . . . . . . . . . . 2
      1.2.2 Existing Parallel Adaptive Algorithms . . . . . . . . . . . . . . . 6
   1.3 Motivation and Thesis Objective . . . . . . . . . . . . . . . . . . . . . 7
   1.4 Scope of Research . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8

2 High-Order CENO Finite-Volume Method 10
   2.1 Godunov-Type Finite-Volume Method . . . . . . . . . . . . . . . . . . . 10
      2.1.1 Overview . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10
      2.1.2 Second-Order Scheme and Slope Limiting . . . . . . . . . . . . . . 12
   2.2 Applications to Conservation Equations . . . . . . . . . . . . . . . . . . . 13
      2.2.1 Advection-Diffusion Equation . . . . . . . . . . . . . . . . . . . . 15
      2.2.2 Euler Equations for Inviscid Compressible Gasdynamics . . . . . . 15
      2.2.3 Semi-Discrete Form . . . . . . . . . . . . . . . . . . . . . . . . . . 16
      2.2.4 Flux Evaluation for Advection-Diffusion Equation . . . . . . . . . 16
      2.2.5 Flux Evaluation for Euler Equations . . . . . . . . . . . . . . . . . 19
   2.3 CENO High-Order Reconstruction . . . . . . . . . . . . . . . . . . . . . 22
      2.3.1 $k$-Exact Reconstruction . . . . . . . . . . . . . . . . . . . . . . . 24
      2.3.2 Smoothness Indicator . . . . . . . . . . . . . . . . . . . . . . . . . . 25
      2.3.3 Reconstruction at the Boundaries . . . . . . . . . . . . . . . . . . 26
      2.3.4 High-Order Flux Evaluation . . . . . . . . . . . . . . . . . . . . . . 26
2.3.5 Solution of the Semi-Discrete Form ........................................ 27
2.4 Newton-Krylov-Schwarz Method ............................................... 27
2.5 Parallel Implementation ......................................................... 29

3 Anisotropic AMR for Body-Fitted Multi-Block Meshes .................. 31
3.1 Overview of Adaptive Refinement Scheme ............................... 31
3.1.1 Anisotropic Adaptive Mesh Refinement ................................. 35
3.2 Solution Block Connectivity .................................................. 35
3.2.1 Hierarchical Tree Data-Structure ....................................... 35
3.2.2 Connectivity Rearrangement ............................................. 36
3.2.3 Computation of Solution Block Connectivity ....................... 38
3.3 Anisotropic Mesh Refinement Procedures ............................... 41
3.3.1 Refinement Criteria ....................................................... 43
3.3.2 Conflict Checking ......................................................... 44
3.4 Grid Generation and Solution Transfer .................................... 48
3.5 Message Passing ............................................................... 51
3.5.1 High-Order Solution Prolongation ..................................... 51
3.5.2 Message Passing for Anisotropic AMR ............................... 53

4 Numerical Results ....................................................................... 57
4.1 Advection Diffusion in Rectangular Channel ............................ 57
4.1.1 Unsteady Results ............................................................. 58
4.1.2 Implicit Results ............................................................... 61
4.1.3 Accuracy Analysis .......................................................... 63
4.2 Circular Advection in Rectangular Box ................................... 64
4.2.1 Unsteady Results ............................................................. 65
4.2.2 Implicit Results ............................................................... 69
4.2.3 Accuracy Analysis .......................................................... 69
4.3 Shock-Box Simulation ........................................................... 70
4.4 Supersonic Flow Past a Cylinder .......................................... 73
4.5 Ringleb Flow ........................................................................ 77
4.6 Complex Shock Interaction ........................................... 79
4.7 Remarks on Computational Savings ................................. 83

5 Conclusions and Future Research ................................. 86
  5.1 Summary of Contributions .......................................... 87
  5.2 Future Considerations ............................................ 88

Bibliography .................................................................. 90
List of Tables

3.1 Pseudo-code for the recursive algorithm used in connectivity rearrangement 37
3.2 Conflicts-Checking Scenarios 45
3.3 Values in a ghost cell during an iterated message passing process 54
4.1 Accuracy Comparison for Circular Advection Simulation 70
List of Figures

1.1 Comparison between the patch-based, cell-based and block-based AMR techniques as illustrated on a Cartesian mesh [10] ........................................ 3

1.2 Illustration of an overlapping grid consisting of two structured curvilinear component grids [10] ................................................................. 5

1.3 Illustration of block-based adaptive mesh refinement on a body-fitted grid [10] 5

2.1 Illustration of the closed-path integral involved in deriving the Godunov’s method ................................................................. 11

2.2 Illustration of slope-limiting through a shock tube simulation at $t = 7$ ms with initial conditions (air on both sides): $\rho_l = 4.696 \text{ kg/m}^3$, $u_l = 0 \text{ m/s}$, 

$\rho_l = 404.4 \text{ kPa}; \rho_r = 1.408 \text{ kg/m}^3$, $u_r = 0 \text{ m/s}$, $p_r = 101.1 \text{ kPa}$. CFL number is 0.7 ................................................................. 14

2.3 Riemann problem solution to a scalar advection problem in one dimension ............................................................................. 17

2.4 1D advection diffusion problem at $t = 5$ for a range of Péclet numbers $(Pe = \frac{aL}{\kappa})$. $u(x, t=0)$=0, $u(x=0, t)=1$, $\frac{\partial u}{\partial x}(x=10, t)=0$. $a=0.5$, CFL=0.45, 

200 cells used ......................................................................................... 17

2.5 The four types of solutions to a one-dimensional Riemann problem for the Euler equations ................................................................. 18

2.6 Solution to the linearized Riemann problem as proposed by Roe, where the flux at $x/t = 0$ could be computed from changes of the left- and right-state fluxes, $F_L$ and $F_R$, as a result of the passage of characteristics ........................................................................... 20
2.7 Solution to the Riemann problem by approximating the wave speeds in
the HLLE and HLLL flux evaluations, around which the path integral for
conservation is applied to obtain the interface flux. .................. 20
2.8 Application of the approximate Riemann flux evaluation methods. .... 23
2.9 The graph of $f(\alpha) = \frac{\alpha}{(1-\alpha)}$. .......................... 25

3.1 Multi-block quadrilateral mesh in block-based AMR with layers of over-
lapping ghost cells to facilitate inter-block communication. .......... 32
3.2 Solution blocks of a computational mesh after four levels of AMR on an
initial block, shown with the corresponding quadtree data structure [28]. 33
3.3 Illustration of refinement and coarsening of an $8 \times 8$ block, during (i)
anisotropic AMR in $\xi$, (ii) anisotropic AMR in $\zeta$ and (iii) isotropic AMR
cell division. Their geometrical relationship are also represented. .... 34
3.4 Computational mesh with 6 anisotropic refinement levels originating from
one initial block. .......................................................... 36
3.5 Two identical meshes resulting from anisotropic mesh refinement in differ-
ent orders, and therefore with different different tree structures. .... 37
3.6 The same mesh as shown in Figure 3.4 with an initial tree structure and
the new tree structure after connectivity rearrangement such that blocks
B and D are siblings in $\zeta$. Branches which have been altered are in red. 38
3.7 Examples illustrating the binary tree traversal during a face neighbor
search and a corner neighbor search. The search paths are indicated by
arrows, where a red arrow would indicate a bridge connection. Mesh block
on the left represent blocks at the corresponding binary tree level, and the
block boundaries are co lour-coded to correspond to the binary tree node
visited................................................................. 39
3.8 Illustration of the deadlock situation during coarsening. ................. 42
3.9 Examples of conflicting scenarios during $\xi$-refinement. The block with bold
boundaries represent the block being checked in relation to its neighbors,
while changes resulted from conflict resolution are marked in red. .... 45
3.10 Examples of permitted and non-permitted coarsening scenarios. Flags for the $\xi$-direction are in blue, and flags for the $\zeta$-direction are in red. Boundary between two sibling blocks are represented by dashed lines. (a-e) $\xi$-coarsening is permitted without modification of flags; (f, g) $\xi$-coarsening is permitted by forcing sibling to refine in $\zeta$; (h, i) $\xi$-coarsening is not permitted due to conflicts in $\zeta$; (j) $\xi$-coarsening is permitted after conflict in $\zeta$ is resolved; (k) $\xi$-coarsening is not permitted because siblings are too fine; (l) Coarsening is not permitted because direction for coarsening is ambiguous.

3.11 Addition of new nodes during anisotropic cell division using metrics.

3.12 (a) Illustration of the “true” domain and its division into multiple blocks; and (b) the low- and (c) the high-order mesh representations for the corner block. Straight line segments are sufficient for representing interior block boundaries. The extension (ghost) boundaries provides complete information about the properties of the neighboring splines [83].

3.13 Comparison of low- and high-order boundary representations during anisotropic mesh refinement. High-order splines allow the boundary of the coarse cell to overlap with those of the fine cells and the physical boundary, unlike the case with the low-order representation shown on the left [83].

3.14 Summary of the different message passing scenarios in anisotropic AMR with graphical illustration of the prolongation/restriction required.

3.15 The values recorded in Table 3.3 is of cell $(0, 17)$ in the northwest corner of the block highlighted in red, which has a north neighbor that is coarser in $\xi$, but finer in $\zeta$.

4.1 Comparison of isotropic and anisotropic AMR in simulating advection and diffusion in a rectangular channel as a time-variant problem, with CENO $k = 4$ reconstruction.

4.2 Convergence histories for explicit unsteady simulation with both AMR methods.
4.3 Comparison of isotropic and anisotropic AMR in simulating advection and diffusion in a rectangular channel as a steady-state problem, with CENO $k = 4$ reconstruction.

4.4 Convergence comparison for anisotropic AMR against isotropic AMR and uniform refinements in simulating advection-diffusion in a rectangular channel using $k = 4$ CENO reconstruction.

4.5 Simulation of circular advection in a rectangular box on a coarse mesh without AMR, with CENO $k = 3$ reconstruction.

4.6 Comparison of isotropic and anisotropic AMR in simulating circular advection in a rectangular box as a time-variant problem, with CENO $k = 3$ reconstruction.

4.7 Comparison of isotropic and anisotropic AMR in simulating circular advection in a rectangular box as a steady-state problem, with CENO $k = 3$ reconstruction.

4.8 Simulation of an unsteady shock-box problem on a coarse mesh without AMR, with $k = 3$ CENO reconstruction.

4.9 Comparison of isotropic and anisotropic AMR in simulating the unsteady shock-box problem, with $k = 3$ CENO reconstruction.

4.10 Steady-state simulation of Mach 2 flow past circular cylinder with radius of 1m on a coarse mesh without AMR, with CENO $k = 3$ reconstruction.

4.11 Comparison of isotropic and anisotropic AMR in simulating Mach 2 flow past circular cylinder with radius of 1m as a steady-state problem, with CENO $k = 3$ reconstruction.

4.12 Convergence comparison for anisotropic AMR against isotropic AMR and a uniformly refined mesh in simulating Ringleb’s flow using $k = 3$ CENO reconstruction.

4.13 Mach number distribution for a transonic Ringleb’s flow, shown with the finest grids from isotropic and anisotropic AMR as used in the convergence studies.
4.14 Coarse initial mesh and the steady-state solution for shock-interaction simulation in a diamond-shaped inlet. ................................. 80

4.15 Density and Mach number distribution for the complex shock interaction simulation in a diamond-shaped inlet, with limited second-order least-squares reconstruction. ........................................ 81

4.16 Comparison of convergence history with both isotropic and anisotropic AMR in simulating complex shock interaction simulation in a diamond-shaped inlet, with CENO $k = 3$ reconstruction. .............................. 82
Chapter 1

Introduction

1.1 Efficient and Accurate Analysis of Complex Flows

Computational fluid dynamics (CFD) allows complex flows to be conveniently analyzed, visualized and predicted by providing numerical solutions to partial differential equations (PDE) which govern the physics of the flows of interest. In such ways, CFD has proved to be an important enabling technology in many areas of science and engineering. In spite of its relative maturity and widespread successes in aerospace engineering, there remain a variety of physically-complex flows, which are still not well understood and have proved to be very challenging to predict by numerical methods. Such flows would include but are not limited to: (i) multiphase, turbulent, and combusting flows encountered in propulsion systems (e.g., gas turbine engines and solid propellant rocket motors); (ii) compressible flows of conducting fluids and plasmas; and (iii) micro-scale non-equilibrium flows, such as those encountered in micro-electromechanical systems. These flows present numerical challenges for they generally involve a wide range of complicated physical/chemical phenomena, exhibit strong anisotropic solution features, as well as involve complex flow geometries.

To overcome these challenges, new CFD algorithm development today must be able to fully harness the potentials of the high-performance computing systems, which have truly come to the forefront and become essential tools in the past 10–15 years. High-performance platforms today are generally distributed-memory parallel clusters and have
achieved petascale performance. The parallel clusters range in size from a few hundreds to thousands of cores, and are capable of up to more than one petaflops, or one quadrillion floating point operations, per second. A recent assessment of the needs for large-scale and high-performance scientific computing indicates that a number of fundamental issues in discretization design must be addressed [1]. The identified issues and challenges included: (i) greater automation of mesh generation via adaptive mesh refinement (AMR) to reduce the time to generate high-quality meshes and for the treatment of complex geometries; (ii) efficient parallel implementations of fully implicit time-marching methods for use in combination with AMR; (iii) efficient high-order temporal and spatial discretizations for reduced computational cost for a specified level of accuracy; and (iv) accurate and robust treatments of multi-scale anisotropic physics. The aforementioned issues have provided the primary motivation for the development of the new anisotropic AMR technique that is described in this thesis.

1.2 Parallel Adaptive Algorithms

1.2.1 Adaptive Mesh Refinement Techniques

In a typical CFD solution process, numerical solutions are obtained by discretizing the PDE of interest according to computational grids which subdivide and discretely represent the physical domain. To obtain numerical solution with a given level of accuracy, a minimum mesh resolution is required which depends on the scheme used and behavior of the solution in the region [2,3]. In general, a uniformly-spaced computational grid is very inefficient because it assumes that the required grid resolution is the same throughout the domain [4,5]. However, this is often untrue, especially for the more complex flows commonly encountered in real-life engineering applications. One approach to reducing the computational costs of physically-complex flow simulations is to use AMR [4,6,2,7,8,9]. Computational grids that automatically adapt to the solution are very effective in treating problems with disparate length scales, providing the required spatial resolution while minimizing memory and storage requirements. The use of AMR in conjunction with
finite-volume schemes has produced some very powerful methods for the treatment of a wide variety of physically-complex flows with complex and/or moving geometries.

Currently there exists several approaches to AMR, with different levels of flexibilities and computational requirements. They can be placed into four general categories, namely “cell-based” approaches, “patch-based” approaches, “block-based” approaches, and lastly “hybrid block-based” approaches. The Cartesian meshes resulting from each AMR approach are illustrated in Figure 1.1. A more detailed discussion of each is now described in the remainder of this section.
The patch-based AMR method was first proposed by Berger, Oliger and Collela as an algorithm for dynamic griding [4,6]. In this approach, the smallest rectangular patches of cells containing cells to be refined are formed through a sophisticated algorithm proposed by Berger [11]. All cells within the patch are then refined. In cell-based AMR, as proposed and developed by, for example, Powell and co-workers [12, 2, 13], as well as by Berger and Aftomis [14, 7, 9], cells are refined individually. The resulting cell hierarchy is stored in a flexible tree data-structure, which tracks the connectivity as new grid points are generated, thereby readily allowing local refinements. The cut-cell method could be used with cell-based AMR to effectively treat complex geometrical boundaries. The application of this method, however, is generally restricted to hyperbolic systems [9]. An alternative approach is illustrated in Figure 1.2 which involves overlapping curvilinear grids conforming to the boundaries, with Cartesian grids filling the interior of the domain. Boden and Toro [15], Brislawn [16], Chessire [17] and Henshaw [18] have shown that the combination of a Chimera overlapping grid [19] with AMR is an effective way to handling problems with both disparate scales and complex geometries. However, this method requires re-gridding after each refinement, and the global conservation properties of the method becomes more difficult to enforce [10].

In block-based AMR, cells are organized into blocks, where they are refined as a unit if at least one of the cells has been flagged for refinement. Each block, regardless of its respective resolution, contains a fixed number of cells. Assuming a doubling of mesh resolution during each refinement, each “parent” block is then divided into four “children” blocks in two dimensions, or eight “children” blocks in three dimensions [10]. A tree-like data-structure is similarly required by block-based AMR as in its cell-based counter-part. In general, block-based AMR leads to larger numbers of refined cells. Therefore, it may appear at first less flexible and efficient than a cell-based approach. However, as the connectivity information is stored per block as opposed to per cell, block-based AMR requires a much simpler quadtree (in two dimensions) or octree (in three dimensions) data-structure [10]. The block-based AMR approach also amortizes the total overhead cost associated with communication between individual cells, over blocks of cells [10]. Applications of block-based AMR on Cartesian meshes have been described by Quirk [20],
Figure 1.2: Illustration of an overlapping grid consisting of two structured curvilinear component grids [10]

Figure 1.3: Illustration of block-based adaptive mesh refinement on a body-fitted grid [10]

Berger [21], Gombosi and co-workers [22, 23, 24, 25, 26], Groth and co-workers [27, 28] have since extended the approach by Groth et al. for computational magnetohydrodynamics [8, 29, 30], and developed a flexible block-based hierarchical data-structure to
facilitate automatic solution-directed mesh adaptation on multi-block curvilinear meshes for complex flow geometries. The use of body-fitted meshes with AMR is illustrated in Figure 1.3. It permits more accurate solution near boundaries although it does introduce added complexities. Furthermore, unlike the overlapping grid technique, conservation properties of the solution scheme is readily enforced discretely [10].

Lastly, hybrid block-based AMR approaches are also possible as shown by Holst and Keppens [31], who applied the technique to general curvilinear coordinate systems. Their hybrid method incorporates the ideas of patch-based AMR into the block-based approach, so that incomplete block families may be created if necessary. To obtain the the grid hierarchy, a doubly linked list for grid points per level is required in addition to the full tree data-structure, hence further complicating the neighbor-searching algorithm in three-dimensions. The study by Holst and Keppens [31] compared the efficiency of the three AMR strategies, including the patch-based, the block-based and the hybrid block-based technique. The comparison was done based on a smooth two-dimensional periodic advection test problem solved using a second-order numerical scheme, for which block-based AMR achieved the same level of accuracy using the least amount of CPU time. However, it should be noted that the conclusion reached by Holst and Keppens applies mainly to two-dimensional problems, and were restricted to classical and relativistic Magnetohydrodynamics (MHD) simulations.

1.2.2 Existing Parallel Adaptive Algorithms

The AMR technique discussed in this thesis belongs to the block-based variants of the aforementioned AMR categories. The block-based approach to AMR is favored for various reasons. First of all, it is perhaps the simplest. Although it may be less flexible in terms of adaptation comparing to, for example, cell-based AMR, it would seem to be a lot more computationally efficient in terms of speed and memory requirements when implemented on a computer [10]. Block-based AMR also lends itself easily to parallel implementation, as it has already been proposed [21] [8] [30], thereby taking full advantage of the high-performance distributed memory clusters available today. This research builds on an existing finite-volume framework for block-based parallel AMR for body-
fitted multi-block mesh developed by Groth and co-researchers [27, 32, 33, 34, 35, 36, 37], which has been shown to enable efficient and scalable parallel implementations for a variety of complex flows, as well as allow for local refinement of body-fitted mesh with anisotropic stretching. Groth and Northrup [28] have later proposed an efficient parallel implementation of a fully implicit Newton-Krylov method, which exploits the block structure of the grid through the use of Schwarz-type preconditioning. More advancements have taken place recently as Ivan and Groth [38, 39, 40] extended the finite-volume framework to incorporate high-order solution reconstruction, further reducing the computational cost needed to achieve a desired level of accuracy. Revisiting the challenges outlined by Keyes [1] in Section 1.1, it appears that the existing numerical framework by Groth et al. as described above has issues (i)–(iii) at least partially taken care of, while the challenge relating to accurate and robust treatments of multi-scale anisotropic physics must still be tackled.

1.3 Motivation and Thesis Objective

With the preceding viewpoints in mind, this thesis proposes a somewhat novel parallel block-based anisotropic AMR technique. Most AMR methods in the past have been isotropic, where the mesh resolution is doubled in all directions upon each refinement. However, for flows with strong anisotropic features like shocks or viscous layers, all dimensions of an isotropic cell are constrained by the required resolution imposed by the rapid solution variation in a single direction. Comparing to a highly anisotropic mesh whose cells have dimensions aligned with the flow features, an isotropic mesh unnecessarily increases the number of computational cells used by many folds. The proposed technique considered herein is unlike anisotropic extensions to AMR procedures considered by other researchers in the past, see, for example, the paper by Ham et al. [41] which described a cell-based anisotropic mesh refinement technique with cell connectivity stored in an unstructured manner [41]. The proposed anisotropic AMR approach in this thesis readily permits local block-based refinement in a preferred direction, while still preserving the parallel efficiencies of block-based isotropic methods.
Development of the proposed block-based anisotropic AMR procedure specifically targets physically complex flow problems having disparate spatial and temporal scales and strong anisotropic features. The aim here is to utilize the flexibility of the new refinement strategy to further reduce the number of computational cells required. This would hopefully allow for more efficient and accurate treatment of narrow layers, discontinuities, and/or shocks in the solutions which occur, for example, in the thin boundary and mixing layers of high-Reynolds-number viscous flows and in the regions of strong non-linear wave interactions of high-speed compressible flows with shocks. While this computational challenge identified by Keyes [1] certainly needs to be addressed, the new anisotropic AMR procedure should be applicable to multi-block body-fitted computational grids for the treatment of complex geometries, and it should also be compatible with the high-order solution reconstruction and implicit Newton-Krylov-Schwarz solver that are already in place.

1.4 Scope of Research

This thesis focuses mainly on the implementation and validation of the described block-based anisotropic AMR framework, as the first step towards a more mature and robust AMR method. Therefore, applications of the proposed technique are limited to relatively simple problems in two-dimensions (2D). If successful, this will lay the foundation for future extensions to multi-dimensions and applications in simulating more complex flows, possibly in combination with more sophisticated error-based refinement criteria/indicators.

Several components of this research have contributed to the development of a functioning anisotropic AMR framework. First of all, the implementation of a binary hierarchical tree data-structure allows for anisotropic refinement and coarsening of the grid blocks in each of the coordinate directions in an independent fashion. The same tree structure is then also used to compute block connectivity on the physical domain using a neighbor searching algorithm. Secondly, a conflict checking procedure has been devised, so that the mesh resulting from AMR does not violate any restrictions imposed by accurate solu-
tion treatment at block interfaces. Lastly, the exchange of solution and flux information between neighboring blocks at different resolution is also necessary and has been considered. The initial design of the framework assumed that anisotropic AMR could be carried out in a similar fashion as isotropic AMR. In practice, however, the added flexibilities of anisotropic AMR gave rise to many issues which could hinder the robustness of the technique. Upon realizing some of these issues, the initial framework underwent several revisions to increase the flexibility and robustness of anisotropic AMR. These revisions included the addition of a tree connectivity rearrangement algorithm and a modified refinement and coarsening procedure. Both of these procedures are aimed at improving the effectiveness of anisotropic AMR particularly in coarsening during unsteady simulations.

The proposed anisotropic AMR technique is applied to the solution of both a model system, the standard advection-diffusion equation, as well as the Euler equations governing compressible, inviscid, gaseous flows in two space dimensions for validation. Performance of anisotropic AMR will be examined in the context of both unsteady and steady simulations, which are carried out in combination with second- or high-order spatial reconstructions. Unsteady results are obtained using explicit time-marching of the appropriate temporal accuracy, such that spatial-accuracy could be preserved. Steady-state results are obtained using parallel implicit methods unless otherwise stated. In addition, simulations of all test cases have been performed on parallel computing platforms. Therefore, the same studies could also serve to demonstrate the compatibility of anisotropic AMR with the existing parallel finite-volume framework.
Chapter 2

High-Order CENO Finite-Volume Method

2.1 Godunov-Type Finite-Volume Method

2.1.1 Overview

The block-based anisotropic mesh adaptation technique considered in this research is implemented and applied as part of a Godunov-type finite-volume method, which will be discussed in details in this section. Finite-volume methods are among the important methods in CFD used to obtain numerical solutions for partial differential equations (PDE) of the following form:

\[
\frac{\partial U}{\partial t} + \vec{\nabla} \cdot \vec{F} = \Phi(\vec{X}, U),
\]  

(2.1)

where \( U \) and \( \vec{F} \) represent the solution and flux respectively, and \( \Phi \) is the source term. All terms in Equation (2.1) are in bold as Equation (2.1) could represent either a scalar or a vector equation. In general, the flux, \( \vec{F} \) consists of a hyperbolic and an elliptic component, and the source term, \( \Phi \), is non-zero. Upon discretizing the domain, the finite-volume method treats each sub-domain or cell as a control volume, over which the differential form of the conservation equations is integrated over the arbitrary control
Figure 2.1: Illustration of the closed-path integral involved in deriving the Godunov’s method.

Finite-volume methods discretely balance the fluxes in and out of the control volume with the time evolution of the cell-averaged solution content located at the centroid of each control volume. Therefore, conservation is always satisfied. Finite-volume methods can also be more physically intuitive and versatile with different types of grids comparing to other numerical methods like finite-difference or finite-element methods.

For hyperbolic conservation laws, coping in a reliable and robust fashion with discontinuities and shocks has been the main challenge. To address this challenge, in 1959, Godunov proposed a first-order accurate finite-volume-type conservative numerical scheme for hyperbolic systems, which involves solutions to Riemann problems in flux evaluations [42]. The original scheme proposed by Godunov assumed a piece-wise constant solution representation within each cell. It then relates the solution change in the control volume to the fluxes through a closed path integral, which is illustrated in Figure 2.1 for one dimension. This leads to the following ordinary differential equation for the time rate
of change of the cell average solution in the cell $i$ of the one-dimensional domain $[42]$: \[ \frac{dU_i}{dt} = -\frac{1}{\Delta x} \left[ F_{i+\frac{1}{2}} - F_{i-\frac{1}{2}} \right]. \] (2.4)

Hyperbolic fluxes through the left and right boundaries of the cell are represented by $F_{i-\frac{1}{2}}$ and $F_{i+\frac{1}{2}}$ respectively. Evaluation of the fluxes requires the exact solution to a Riemann problem with initial conditions corresponding to the left and right states at every cell interface. Once the intermediate solution state along $x/t = 0$ is obtained, the fluxes can be evaluated as illustrated in Figure 2.1. Godunov’s method leads to natural upwinding for the hyperbolic part of the PDE of interest, which provides the least amount of artificial dissipation required to ensure monotonicity for a first-order scheme $[42]$. However, Godunov’s scheme must be modified and/or added to when handling systems that are not strictly hyperbolic and contain additional fluxes of an elliptic nature. Detailed discussion of both the hyperbolic and elliptic flux evaluation will be included in Section 2.2.4 and Section 2.2.5 in the context of the advection diffusion equation and Euler equations.

### 2.1.2 Second-Order Scheme and Slope Limiting

The original, first-order, Godunov scheme described above is overly dissipative for most problems, which largely increases the required spatial resolution to accurately capture solution features. However, Godunov’s theorem has stated that no linear scheme more accurate than first-order is oscillation-free near shocks $[42]$. Oscillations in the solution may lead to unphysical situations like negative density and pressure. To make the finite-volume scheme second-order accurate, a linear reconstruction in one dimension (1D) can be applied in conjunction with slope-limiting:

\[ U_i(x) = \bar{U}_i + \psi_i \left[ \frac{\partial U}{\partial x} \bigg|_{i} (x - x_i) \right], \] (2.5)

where $U_i$ is the piece-wise linear solution reconstruction in cell $i$. The second-order scheme in this thesis approximates the first derivative, $\frac{\partial U}{\partial x} \bigg|_{i}$, using least-squares reconstruction with a central stencil. Least-squares reconstruction assumes a total reconstruc-
tion error with the following form:

\[
\epsilon^2 = \sum_{n=1}^{N} \epsilon^2_n = \sum_{n=1}^{N} \left[ \bar{U}_n - \sum_{k=1}^{\text{dim}(\vec{X})} \left( x_{k,n} - x_{k,i} \right) \right]^2,
\]

for \( N \) neighbors. By seeking to minimize \( \epsilon^2 \) with respect to the first derivative in each dimension, the derivative values could be solved for. In 1D, the final expression for the first-derivative reduces to a second-order central difference approximation:

\[
\left. \frac{\partial U}{\partial x} \right|_i = \frac{1}{2\Delta x} (\bar{U}_{i+1} - \bar{U}_{i-1}).
\]

The slope limiter is represented by \( \psi \) in Equation (2.5) and takes on values between \([0, 1]\). The slope-limiters are designed to clip any new local extrema that are created by the reconstruction. As the slope-limiter values depend on the cell-centered solution average values, the scheme becomes non-linear and Godunov’s theorem no longer applies.

The current finite-volume framework uses slope-limiters by Barth-Jesperson [43] and Venkatakrishnan [44] because they are more easily extendable to multi-dimensions. For applications in two or higher dimensions, slope-limiter values are computed in terms of both the cell-averaged solution and the solution at each flux evaluation quadrature point, and the minimum slope-limiter value for the cell is used.

Application of the Venkatakrishnan slope-limiter to a 1D shock-tube problem is illustrated in Figure 2.2. As observed in Figure 2.2a, application of the slope-limiter clips any overshoots resulting from discontinuities, but has little impact on the solution otherwise. Figure 2.2b compares the first- and the second-order solutions. It is evident that the second-order scheme results in a better-resolved solution more closely resembling the exact analytical solution obtained from solving the Riemann problem.

### 2.2 Applications to Conservation Equations

Evaluation of the proposed anisotropic AMR method will focus on application to the solution of two sets of conservation equations: (i) the advection-diffusion equation, and
(a) The effect of slope-limiting

(b) Comparison of first- and second-order schemes

**Figure 2.2:** Illustration of slope-limiting through a shock tube simulation at $t = 7$ ms with initial conditions (air on both sides): $\rho_l = 4.696 \text{ kg/m}^3$, $u_l = 0 \text{ m/s}$, $p_l = 404.4 \text{ kPa}$; $\rho_r = 1.408 \text{ kg/m}^3$, $u_r = 0 \text{ m/s}$, $p_r = 101.1 \text{ kPa}$. CFL number is 0.7.

(ii) the 2D Euler equations. Discretization of these equations of interest will be briefly discussed in this section, specifically in the context of the Godunov-type finite-volume scheme.
2.2.1 Advection-Diffusion Equation

A model advection-diffusion equation with a scalar solution variable, \( u \), is first considered for the evaluation of the proposed anisotropic AMR scheme. This model equation is given by

\[
\frac{\partial u}{\partial t} + \vec{\nabla} \cdot (\vec{V} u) = \vec{\nabla} \cdot (\kappa \vec{\nabla} u) + \phi (x, y, u) .
\] (2.8)

The first term of the equation is the time rate of change of \( u \); the advective flux is represented by \( \vec{\nabla} \cdot (\vec{V} u) \), where \( \vec{V} \) is the prescribed advective velocity field; the diffusive flux is represented by \( \vec{\nabla} \cdot (\kappa \vec{\nabla} u) \), where \( \kappa \) is the diffusion coefficient. The Péclet number, which is defined as \( \text{Pe} = aL/\kappa \), offers an indication of whether a given problem is advection-dominated or diffusion-dominated. The last term of Equation (2.8) is a source term, which may be non-linear if dependent on \( u \).

2.2.2 Euler Equations for Inviscid Compressible Gasdynamics

Application of the proposed numerical framework in solving the Euler equations allows the scheme to be evaluated in the context of more interesting and realistic problems. The strong conservation form of the Euler equations is given as follows:

\[
\frac{\partial U}{\partial t} + \vec{\nabla} \cdot \vec{F} = 0 ,
\] (2.9)

which, in two-dimensions (2D), can be written as

\[
\frac{\partial U}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = 0 .
\] (2.10)

The vector of conserved solution variables, \( U \), is given by

\[
U = [ \rho , \rho u , \rho v , \rho e ]^T ,
\] (2.11)

where \( \rho \) is the gas density, \( u \) and \( v \) are \( x \) and \( y \) components of the velocity vector, \( e = p/(\rho(\gamma - 1)) + u^2/2 \) is the specific total energy, \( p = \rho RT \) is the pressure, \( T \) is the gas temperature, \( R \) is the specific gas constant and \( \gamma \) is the specific heat ratio. Furthermore,
the flux vectors $\mathbf{F}$ and $\mathbf{G}$, associated with the $x$- and $y$-direction respectively, are each defined as

$$
\mathbf{F} = \left[ \rho u, \rho u^2 + p, \rho uv, u (\rho e + p) \right]^T,
$$

(2.12)

and

$$
\mathbf{G} = \left[ \rho v, \rho uv, \rho v^2 + p, v (\rho e + p) \right]^T.
$$

(2.13)

### 2.2.3 Semi-Discrete Form

After applying a finite-volume spatial discretization to either Equation (2.8) and Equation (2.10) for a 2D quadrilateral cell indexed $(i,j)$, the following semi-discrete form can be obtained:

$$
\frac{d \bar{\mathbf{U}}_{i,j}}{dt} = -\frac{1}{A_{i,j}} \sum_{l=1}^{N_f} \sum_{m=1}^{N_G} \left( \omega \mathbf{F} \cdot \bar{n} \Delta l \right)_{i,j,l,m} + \mathbf{S}_{i,j} = \mathbf{R}_{i,j},
$$

(2.14)

where

$$
\bar{\mathbf{U}}_{i,j} = \frac{1}{A_{i,j}} \int_A \mathbf{U}_{i,j} dA.
$$

(2.15)

This is achieved by integrating each term in the PDEs over the control volume of the cell, which is its area $A_{i,j}$, and subsequently applying the divergence theorem. The cell solution average is denoted $\bar{\mathbf{U}}_{i,j}$. Solution state obtained by evaluating the solution reconstruction polynomial, $\mathbf{U}_{i,j}$, is used in calculating the interface flux. The total flux through each face, $\bar{\mathbf{F}}$, is computed using an $N_G$-point Gauss Quadrature numerical integration procedure, after which the fluxes through $N_f = 4$ faces of the cell are summed. A sufficient number of Gauss points are chosen per face such that the desired global accuracy of the scheme is preserved. Finally, $\mathbf{R}_{i,j}$ is the cell residual.

### 2.2.4 Flux Evaluation for Advection-Diffusion Equation

The hyperbolic, or advective, flux of the advection diffusion equation is defined as $\bar{\mathbf{F}}_a \cdot \bar{n} = u \bar{V} \cdot \bar{n}$. For a scalar advection problem, the solution to the Riemann problem involves one characteristic speed, given by the advection field $\bar{V}$. Therefore, the interface solution used in the flux evaluation is simply the upwind solution, or $u(x/t = 0)$ in one dimension as illustrated in Figure 2.3. For two dimensions, the direction of upwinding is determined...
Figure 2.3: Riemann problem solution to a scalar advection problem in one dimension.

Figure 2.4: 1D advection diffusion problem at $t = 5$ for a range of Péclet numbers ($Pe = \frac{aL}{\kappa}$). $u(x, t=0)=0$, $u(x=0, t)=1$, $\frac{\partial u}{\partial x}(x=10, t)=0$. $a=0.5$, CFL=0.45, 200 cells used.

by the sign of the dot product between $\vec{V}$ and the normal to the cell $\vec{n}$. That leads to the following expression for the hyperbolic flux

$$\vec{F}_a \cdot \vec{n} = \begin{cases} 
  u_l(\vec{V} \cdot \vec{n}) & \text{if } \vec{V} \cdot \vec{n} \geq 0, \\
  u_r(\vec{V} \cdot \vec{n}) & \text{if } \vec{V} \cdot \vec{n} < 0.
\end{cases}$$  \hspace{1cm} (2.16)

The left state, $u_l$, refers to the solution state according to cell $(i, j)$, for which the flux is evaluated; the right state, $u_r$, refers to the solution state in a neighbor of cell $(i, j)$. The left and right states are obtained by evaluating the respective solution reconstruction at each Gauss Quadrature point along the cell interface.

Elliptic flux evaluation can be incorporated within or added to the Godunov finite-
volume formulation using a centrally-weighted approach. For the advection-diffusion equation, the elliptic, or diffusive, flux is defined as $\vec{F}_d \cdot \vec{n} = -\kappa \nabla u \cdot \vec{n}$, which is numerically integrated over the boundaries of the control volume. Using first-order forward-difference in one dimension, the left and right interface fluxes for cell $i$ becomes

$$F_{d,l} = \kappa (u_{\text{face}}) \frac{u_i - u_{i-1}}{\Delta x}$$

and

$$F_{d,r} = -\kappa (u_{\text{face}}) \frac{u_{i+1} - u_i}{\Delta x}$$

respectively, where $u_{\text{face}}$ represents the corresponding interface solution obtained using interpolation. Upon summing $F_{d,l}$ and $F_{d,r}$ according to Equation (2.14) and assuming a constant $\kappa$, the total elliptic flux reduces to the second-order central difference approximation to $\kappa \frac{\partial^2 u}{\partial x^2}$, or

$$F_{d,\text{tot}} = \kappa \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2}.$$  

(2.18)

In two dimensions, the average gradient at every interface is approximated through diamond-path reconstruction [45], which, when summed up according to Equation (2.14)
also reduces to a second-order central difference approximation to \( \kappa \frac{d^2 u}{dx^2} \) for constant \( \kappa \). Treatment for the elliptic flux is slightly different for the high-order scheme, which will be discussed separately in Section 2.3.

A sample advection diffusion problem solved using the extended Godunov’s approach is presented in Figure 2.4. The solution has been advanced in time using the RK2 explicit time-marching method. To ensure stability, the time step used to update the solution in time satisfied both of \( \frac{a \Delta x}{\Delta t} < 1 \) for advection and \( \frac{\kappa \Delta x^2}{\Delta t} < \frac{1}{2} \) for diffusion.

2.2.5 Flux Evaluation for Euler Equations

The flux term \( \vec{F} \) for Euler equations is given by \( \vec{F} = \vec{F}_i + \vec{G}_j \), where \( \vec{F} \) and \( \vec{G} \) are the flux vectors defined previously. The numerical flux could be written as follows in terms of the left and right states:

\[
\vec{F} \cdot \vec{n} = \vec{F} \left( \text{int}(\vec{U}_l, \vec{U}_r, \vec{n}) \right) \cdot \vec{n}.
\] (2.19)

The interface solution at every Gauss Quadrature point of all cell faces, \( \text{int} \), is obtained by solving a Riemann problem in the direction indicated by the face normal \( \vec{n} \), and with initial states defined by the reconstructed solution values, \( \vec{U}_l \) and \( \vec{U}_r \), on either side of the cell interface. Depending on the left and right states, one of the four wave patterns from Figure 2.5 could emerge. The different Riemann problem solutions are distinguished by the nature of the three characteristic waves, which may each represent a shock wave, a contact wave, or a rarefaction wave, with an addition of a shear wave in two dimensions. Upon obtaining the wave patterns, the interface flux is evaluated using the time-invariant solution along \( x/t = 0 \).

Although an efficient exact Riemann solver has been proposed by Gottlieb and Groth [46], the same may not be done for all systems, where the exact solution to a Riemann problem either does not exist or is difficult to obtain. This is where the benefits of approximate Riemann problem solutions arise. In addition, computation of the exact flux can be somewhat more expensive, although not significantly so for the Euler system. It is, however, not always necessary due to the presence of discretization error,
Figure 2.6: Solution to the linearized Riemann problem as proposed by Roe, where the flux at \( x/t = 0 \) could be computed from changes of the left- and right-state fluxes, \( F_L \) and \( F_R \), as a result of the passage of characteristics.

Figure 2.7: Solution to the Riemann problem by approximating the wave speeds in the HLLE and HLLL flux evaluations, around which the path integral for conservation is applied to obtain the interface flux.

in which case approximate solutions to the Riemann problem would often suffice.

One alternative to the exact Riemann solver was proposed by Roe \[47\], which is based on an alternative form of Equation (2.10) written as:

\[
\frac{\partial U}{\partial t} + A \frac{\partial U}{\partial x} + B \frac{\partial U}{\partial y} = 0, \quad \text{where} \quad A = \frac{\partial F}{\partial U} \quad \text{and} \quad B = \frac{\partial G}{\partial U}. \tag{2.20}
\]

For a linear system in the above form where the flux Jacobians \( A \) and \( B \) are independent of \( U \), the changes in the characteristic variables could be determined from the passage of the characteristic waves, which in turn could be used to compute the conservative fluxes. Although the Euler equations are non-linear, Roe has proposed to locally linearize the system by computing \( \bar{A} = \bar{A}^*(U^*) \) and \( \bar{B} = \bar{B}^*(U^*) \) from an appropriate Roe-averaged
Subsequently, the eigenstructure of the Jacobian matrices can be used to obtain the interface flux without solving the Riemann problem. To illustrate this in one dimension, the conserved flux could be determined using the following expression:

$$\mathcal{F} = \begin{cases} F(U_L) + \sum_{\lambda_k^* < 0} \alpha_k^* \lambda_k^* r_{ck}^* & \text{where } \alpha_k^* = l_{pk}^* \cdot (W_R - W_L). \\ F(U_R) - \sum_{\lambda_k^* > 0} \alpha_k^* \lambda_k^* r_{ck}^* & \end{cases} (2.21)$$

The right eigenvectors of the conservative flux Jacobian matrix are presented by \( r_{ck}^* \); the left eigenvectors of the primitive flux Jacobian matrix are represented by \( l_{pk}^* \); the change in primitive solution variables across the interface is represented by \( W_R - W_L \) and the characteristic wave strength by \( \alpha_k^* \). As shown in Figure 2.8a, the Roe solver allows for expansion shocks as the upwind dissipation for rarefaction waves vanishes at sonic point. This is fixed by adding Harten’s entropy correction [48].

Another alternative is the HLLE flux functions by Harten, Lax, Van Leer and Einfeldt [49], in which the approximate solution to the Riemann problem assumes three resultant states separated by two waves, one with a "maximum" speed and one with a "minimum" speed. The wave patterns assumed are illustrated in Figure 2.7a. The conservation law is then applied by integrating around the adjacent control volume to obtain an approximation to intermediate solution, \( U^* \). The flux is then computed from one of the three states depending on the direction of the wave motion. The approach was improved by Linde in the HLLL formulation to account for a third wave (e.g., a shear wave) or contact surface, which is represented in Figure 2.7b. For a one-dimensional problem, the HLLL approximate flux function has the following form [50]:

$$\mathcal{F} = \begin{cases} F_L = F(U_L) & \text{if } \lambda^- \geq 0 \\ F_R = F(U_R) & \text{if } \lambda^+ \leq 0 \\ F^* = \frac{\lambda^+ F_L - \lambda^- F_R}{\lambda^+ - \lambda^-} + \frac{\lambda^+ \lambda^-}{\lambda^+ - \lambda^-} \left[ 1 - \left( 1 - \max\left( \frac{u_{roe}}{c}, \frac{u_{roe}}{c} \right) \right) \alpha \right] (U_R - U_L) & \text{otherwise} \quad (2.22) \end{cases}$$

where \( \lambda^+ \) and \( \lambda^- \) are the maximum and minimum wave speeds in both HLLE and HLLL; \( U_R - U_L \) is the change in conserved solution variable. The correction factor added by
Linde is given by \[ 1 - \left( 1 - \max\left( \frac{u_{\text{roe}}}{\lambda}, \frac{u_{\text{roe}}}{\lambda} \right) \right) \alpha \], where \( \alpha \) is an indicator for the presence of a contact surface in the Riemann problem solution.

First-order accurate solutions produced by the exact Riemann solver and the approximate flux functions are compared in Figure 2.8b. Results from the corrected Roe solver is comparable to the exact approach. The HLLL flux provides better resolution of the contact surface than HLLE flux, but it is still more dissipative than Roe’s method and the exact Riemann solver.

2.3 CENO High-Order Reconstruction

Methods up to second-order can still exhibit excessive numerical dissipation and are often not practical for physically-complex flows. Hence improved numerical efficiency may be achieved by further raising the order of accuracy of the spatial discretization. In the essentially non-oscillatory (ENO) high-order finite-volumes schemes proposed by Harten et al. [51], computational stencils containing discontinuities are avoided to achieve solution monotonicity. Although a weighted ENO (WENO) scheme attempts to simplify the ENO procedure by adopting a stencil-weighting approach, both ENO and WENO variants encounter difficulties when selecting appropriate stencils for general multi-dimensional unstructured meshes. [52, 53, 54, 55], and they can result in poor conditioning of the linear systems during reconstruction [54, 55]. These, along with the associated computational cost and complexities involved in the ENO and WENO finite-volume schemes, have limited the range of applications where such methods may be applied.

As an alternative, the finite-volume framework described in this thesis makes use of a high-order central ENO (CENO) method recently proposed by Ivan and Groth [39, 40]. As a variant of the ENO scheme, the CENO scheme uses a fixed central stencil, which in general provides the most accurate reconstruction due to cancellation of truncation errors. This hybrid scheme first performs an unlimited \( k \)-exact reconstruction, and then automatically reverts to a monotonicity preserving limited piecewise linear reconstruction in cells near shocks or with under-resolved solution content. The switching is controlled by a solution smoothness indicator. The CENO reconstruction is effective in eliminating the
Chapter 2. High-Order CENO Finite-Volume Method

(a) Rarefaction wave simulation with air: $\rho_l = 1.598 \text{ kg/m}^3$, $u_l = -383.84 \text{ m/s}$, $p_l = 91.88 \text{ kPa}$; $\rho_r = 2.787 \text{ kg/m}^3$, $u_r = -216.97 \text{ m/s}$, $p_r = 200 \text{ kPa}$. $t = 35 \text{ ms}$

(b) Shock tube simulation with air: $\rho_l = 4.696 \text{ kg/m}^3$, $u_l = 0 \text{ m/s}$, $p_l = 404.4 \text{ kPa}$; $\rho_r = 1.408 \text{ kg/m}^3$, $u_r = 0 \text{ m/s}$, $p_r = 101.1 \text{ kPa}$

Figure 2.8: Application of the approximate Riemann flux evaluation methods.
appearance of $O(1)$ numerical oscillations in under-resolved regions and in solutions that contain strong discontinuities and/or shocks. Although uniform accuracy is not achieved for non-smooth solutions, the method is easily extendable to both multi-dimensions and unstructured mesh.

### 2.3.1 $k$-Exact Reconstruction

The $k$-exact reconstructed solution of variable $u$ in cell $(i, j)$ assumes a polynomial form given by [39]

$$u_{i,j}^k(\vec{r}) = \sum_{p_1=0}^{N_1} \sum_{p_2=0}^{N_2} (x - \bar{x}_{i,j})^{p_1} (y - \bar{y}_{i,j})^{p_2} D_{p_1p_2}^k,$$

(2.23)

where $N_1$ and $N_2$ satisfy $N_1 + N_2 \leq k$; $\bar{x}_{i,j}$ and $\bar{y}_{i,j}$ are coordinates of the cell centroid; and $D_{p_1p_2}^k$ are the coefficients for each polynomial term [39]. To determine the values of $D_{p_1p_2}^k$, constraints are imposed requiring that the average solutions of cell $(i, j)$ and cells in its reconstruction stencil are conserved, or that $\bar{u}_{i,j} = (1/A_{i,j}) \int_{A_{i,j}} u_{i,j}^k(\vec{r}) dA$. Additional neighboring cells are included in the reconstruction stencil beyond what is required to uniquely determine all coefficients, so that reconstruction is more robust for stretched meshes or meshes that are not aligned with the solution gradients. The current reconstruction scheme uses a fixed central stencil with 8 neighboring cells for $k=1$ and 24 neighboring cells for $k=2$, $k=3$ and $k=4$ [39].

The above formulation results in an overdetermined system of linear equations $Ax = B$, which is solved using orthogonal decomposition by Single Value Decomposition (SVD) method [56]. This method involves the computation of a pseudo-inverse matrix $A^{-1}$, where $A$ is unchanged for a fixed stencil in CENO [39]. Therefore, the pseudo-inverse matrix could be stored and reused to speed up the computation. This is another advantage of the CENO method, in addition to the avoidance of reconstruction on multiple stencils and poorly conditioned coefficient matrices, $A$, which may occur in the other Essentially Non-Oscillatory (ENO) and Weighted Essentially Non-Oscillatory (WENO) schemes. Preservation of solution average in cell $(i, j)$ is explicitly enforced by expressing $D_{00}^k$ as a function of the other unknowns [39].
2.3.2 Smoothness Indicator

The CENO scheme preserves solution monotonicity in regions of large gradients or discontinuities by reverting the high-order $k$-exact reconstruction to a limited piecewise linear ($k = 1$) reconstruction. The slope limiters of Barth-Jespersen and Venkatakrishnan could be used [57, 44]. The switching is triggered by a smoothness indicator, which identifies under-resolved regions that may require a drop in order. The smoothness indicator, $S$, is calculated in terms of a smoothness parameter $\alpha$ in the following manner [39]:

$$S = \frac{\alpha}{\max((1 - \alpha), \varepsilon)} \frac{(\text{SOS} - \text{DOF})}{\text{DOF} - 1},$$

(2.24)

where

$$\alpha = \frac{\sum_{\gamma} \sum_{\delta} (u_{i,j}^k (\bar{r}_{\gamma,\delta}) - u_{i,j}^k (\bar{r}_{\gamma,\delta}))^2}{\sum_{\gamma} \sum_{\delta} (u_{i,j}^k (\bar{r}_{\gamma,\delta}) - \bar{u}_{i,j})^2}.$$

(2.25)

In the above equation, SOS and DOF refer to the reconstruction stencil size and number of unknowns respectively, whereas $\varepsilon$ is a tolerance added to prevent division by zero. The computation of $\alpha$ involves comparing values of reconstructed solution at the centroids, $\bar{r}_{\gamma,\delta}$, of all stencil cells indexed by $\gamma$ and $\delta$ [39]. Figure 2.9 illustrates the behavior of

Figure 2.9: The graph of $f(\alpha) = \frac{\alpha}{(1 - \alpha)}$.
\[ \frac{\alpha}{(1-\alpha)} \], which rapidly increases to \( \infty \) as \( \alpha \) approaches 1, indicating a smooth and well-reconstructed solution [39]. Finally, a smoothness cutoff value \( S_c \) is specified, so that if \( S > S_c \), the reconstruction is considered smooth, otherwise the order of reconstruction is dropped locally.

### 2.3.3 Reconstruction at the Boundaries

Correct high-order treatment of boundary conditions is crucial in maintaining the accuracy of the scheme. In the current approach, extra rows of ghost cells are added beyond the geometric boundary of the computational domain to impose high-order boundary conditions. Least-squares reconstruction in control volumes adjacent to the boundary are constrained as described by Olivier-Gooch and Van Altena [58]. Furthermore, geometric data are computed to the same order of accuracy as that of the interior scheme. The Boundary constraints in addition to the reconstruction conditions lead to an enlarged system of equations, which is solved using Gauss elimination with pivoting followed by a Householder QR factorization for the remaining least-squares problem.

### 2.3.4 High-Order Flux Evaluation

Accuracy of the hyperbolic flux evaluation is automatically improved with a higher-order reconstruction, which offers better estimates of cell interface solution values. For a \( k \)-exact solution reconstruction, hyperbolic flux evaluations are \( k+1 \)-order accurate. Treatment of the elliptic operator involves taking the arithmetic average of the solution gradient from the left and right states. Therefore, the elliptic flux value for the advection-diffusion equation at every cell interface is expressed as

\[
\vec{F}_d \cdot \vec{n} = -\kappa \left[ \frac{1}{2} \left( \hat{\nabla} u^k_l(\vec{r}) + \hat{\nabla} u^k_r(\vec{r}) \right) \right] \cdot \vec{n}. \tag{2.26}
\]

The diffusion coefficient, \( \kappa \), is evaluated using an interpolated interface solution. The position vector to the Gauss quadrature point is denoted as \( \vec{r} \). The solution gradient, \( \hat{\nabla} u^k_l(\vec{r}) \) and \( \hat{\nabla} u^k_r(\vec{r}) \), are obtained from analytically differentiating the \( k \)-exact solution reconstruction. Gradient evaluation in the elliptic flux evaluation using a \( k \)-order CENO
reconstruction is $k$-order accurate, which makes the scheme globally $k$-order accurate in the presence of an elliptic term.

### 2.3.5 Solution of the Semi-Discrete Form

For time-accurate simulation of unsteady problems, multi-stage explicit time-marching schemes have been used to advance the ordinary differential equation described by Equation (2.14) in time. A fourth-order Runge-Kutta scheme is used for $k > 2$ to preserve the global accuracy of the scheme, whereas a second-order predictor-corrector method or explicit Euler method may be used for time-integration for $k \leq 2$. For most steady-state problems, an implicit Newton-Krylov-Schwarz method is used to speed up the solution process, which will be described in the next section.

### 2.4 Newton-Krylov-Schwarz Method

Steady-state problems can be very effectively solved by application of Newton’s method to the resulting nonlinear system of algebraic equations, and subsequently a Krylov subspace method, such as the generalized minimal residual (GMRES) algorithm proposed by Saad [59], in combination with a domain-based additive Schwarz preconditioning technique [59] to solve the large, sparse, system of linear equations at each Newton step. In addition, this Newton-Krylov-Schwarz (NKS) approach is well suited to exploiting the potential of distributed-memory multi-processor machines because the Schwarz preconditioner breaks the problem into a set of sub-problems. Rather efficient parallel implementations of implicit algorithms via Schwarz preconditioning have been developed by Keyes and co-researchers and successfully applied to a range of flow problems [60, 61, 62, 63]. Following these ideas, Groth and co-researchers [28, 64, 65] have developed a rather effective and scalable parallel implicit method based on a Jacobian-free inexact NKS approach with additive Schwarz preconditioning and domain partitioning following from the multi-block AMR mesh.

For all steady-state problems of interest, the described NKS method seeks numerical
solution which satisfies
\[ \mathbf{R}(\mathbf{U}) = 0. \] (2.27)

This often leads to large coupled non-linear systems of algebraic equations, to which
Newton’s method offers a robust and efficient iterative technique to compute the solution.
Starting with an initial estimate, \( \mathbf{U}^0 \), successively improved estimates of the solution are
obtained by solving
\[ \left( \frac{\partial \mathbf{R}}{\partial \mathbf{U}} \right)^n \Delta \mathbf{U}^n = \mathbf{J}^n \Delta \mathbf{U}^n = -\mathbf{R}(\mathbf{U}^n), \] (2.28)
at each step, \( n \), of the Newton method, and an improved approximation of the solution
is given by
\[ \mathbf{U}^{n+1} = \mathbf{U}^n + \Delta \mathbf{U}^n, \] (2.29)
where \( \mathbf{J} = \frac{\partial \mathbf{R}}{\partial \mathbf{U}} \) is the residual Jacobian. The iterative procedure is repeated un-
til an appropriate norm of the solution residual is sufficiently small, i.e., \( \|\mathbf{R}(\mathbf{U}^n)\|_2 < \epsilon \|\mathbf{R}(\mathbf{U}^0)\|_2 \) where \( \epsilon \) is some small tolerance value typically around approximately \( 10^{-12} \).

Each step of the Newton iterations requires the solution of a system of linear equa-
tions given by [Equation (2.28)]. For most practical flow computations, this system is large,
 sparse, and non-symmetric. Therefore, the implicit algorithm involved in this paper uses
a class of Krylov subspace iterative methods known as GMRES methods, developed by
Saad and co-workers [66, 67, 68, 59], which are applied extensively in obtaining the so-
lutions to such large sparse non-symmetric linear equations [69, 70, 71, 72, 73, 63]. The
GMRES method is used in conjunction with Schwarz preconditioning as the Jacobian
matrix is often ill-conditioned. A combination of global and local preconditioning tech-
niques is used, where the global additive Schwarz preconditioner for \( N \) solution blocks is
defined as follows:
\[ \mathbf{M}^{-1} = \sum_{k=1}^{N_{\text{blocks}}} \mathbf{B}_k^T \mathbf{M}_k^{-1} \mathbf{B}_k. \] (2.30)
The gather operator or matrix, \( \mathbf{B}_k \), gathers the solution unknowns for the \( k^{th} \) domain
from the global solution vector, and \( \mathbf{M}_k^{-1} \) is the local block preconditioner for the \( k^{th} \)
domain. The described block-based anisotropic AMR fits well with the use of Schwarz
preconditioning in NKS, and is therefore able to readily enable parallel implementation of the overall Newton method.

Application of GMRES leads to an overall solution algorithm with iterations within iterations: an “inner loop” of iterations to obtain the solution of the linear system, within an “outer loop” of iterations associated with the solution of the nonlinear problem. An inexact Newton method is used in this approach, in which the inner iterations are carried out only until \( \|R^n + J^n \Delta U^n\|_2 \leq \zeta \|R^n\|_2 \), where \( \zeta \) is typically in the range 0.1-0.5.

### 2.5 Parallel Implementation

The multi-block quadrilateral mesh and tree data structure lends itself naturally to domain decomposition and enables efficient and scalable implementations of both explicit and implicit solution algorithms on distributed-memory multi-processor architectures [27, 28]. A parallel implementation of the block-based anisotropic AMR scheme has been developed and is used to generate numerical results for both the 2D advection-diffusion problem and the 2D Euler Equations in this paper. The test cases do not benefit most significantly from the use of distributed-memory architectures due to the relatively small problem size in two dimensions, as well as the small range of scales involved. However, they can demonstrate the feasibility of combining the high-order and anisotropic AMR framework with parallel implementation, which can be equally applied to complex flow simulations in three dimensions.

The existing parallel implementation uses the C++ programming language and the MPI (message passing interface) library [74]. Domain decomposition is carried out by farming the solution blocks out to the separate processors, with more than one block permitted on each processor. For homogeneous architectures with multiple processors all of equal speed, an effective load balancing is achieved by exploiting the self-similar nature of the solution blocks and simply distributing the blocks equally among the processors. For heterogeneous parallel machines, such as a network of workstations and computational grids, a weighted distribution of the blocks can be adopted to preferentially place more blocks on the faster processors and less blocks on the slower processors.
Inter-processor communication is mainly associated with block interfaces and involves the exchange of ghost-cell solution values and conservative flux corrections at every stage of the multi-stage time integration procedure. Message passing of the ghost-cell values and flux corrections is performed in an asynchronous fashion with gathered wait states and message consolidation.
Chapter 3

Anisotropic AMR for Body-Fitted Multi-Block Meshes

Adaptive mesh refinement (AMR) techniques ensure all regions on the domain are sufficiently resolved without over-resolving low-gradient regions, and is therefore very effective in treating problems with disparate spatial scales. Anisotropic variants of these techniques offer greater flexibility by adding directionality to the refinement process, thereby more effectively treating problems with anisotropic flow features. This section describes in detail the implementation of the proposed new block-based anisotropic AMR method.

3.1 Overview of Adaptive Refinement Scheme

The AMR technique proposed in this thesis follows the approach by Groth et al. for computational magnetohydrodynamics [8,30]. A block-based hierarchical data structure is used in conjunction with the finite-volume scheme described above to facilitate automatic solution-directed mesh adaptation on multi-block mesh according to physics-based refinement criteria. “Blocks” here refer to patches of adjacent cells that are refined or coarsened as a group during AMR. The proposed AMR formulation borrows from previous work by Berger and co-workers [4,7,21,6], Quirk [20,73], and De Zeeuw and Powell [2] for Cartesian mesh. It is similar to the block-based approaches described by Quirk and Hanebutte [20], and Berger and Saltzman [21]. Cell-based Cartesian mesh
Chapter 3. Anisotropic AMR for Body-Fitted Multi-Block Meshes

Figure 3.1: Multi-block quadrilateral mesh in block-based AMR with layers of overlapping ghost cells to facilitate inter-block communication.

adaptation procedures with more arbitrary quadrilateral and hexagonal mesh have been considered by Davis and Dannenhoffer [76], and Sun and Takayama [77]. In comparison to a cell-based approach, block-based AMR may at times be less flexible and efficient. However, a block-based approach involves communications overhead required per block of cells instead of per cell, and less connectivity also results in a simpler data structure. More importantly, the block-based approach most easily lends itself to parallel implementation, and may thus be the preferred alternative especially for large-scale problems.

In the current approach, every block consists of a fixed number of $N_i \times N_j$ cells. Both $N_i$ and $N_j$ must be even, but not necessarily equal. Cell-averaged solution states within each block are stored in an indexed array which corresponds to the physical arrangement of the cells. In addition to these interior cells, there are layers of “ghost” cells which store solution data of interior cells from neighboring blocks, as shown in Figure 3.1. These overlapping data storage allow the finite-volume calculations on each block to be carried out in a more independent manner. A message passing routine updates the solution in ghost cells as solution in the corresponding interior cells evolve. This may or may not involve restriction or prolongation of the solution depending on the resolution.
change between adjacent blocks. Message passing procedures will be discussed in more details later in this section. Additional inter-block communication is required to correct interface fluxes computed on coarser neighbors using fluxes from their finer neighbors, so that the flux conservation properties of the finite-volume scheme is strictly enforced across the interface [4, 6].

The adaptive procedure is directed by one or more refinement criteria for each block. Percentage thresholds are specified by the user, such that blocks with refinement criteria greater than the refinement threshold are flagged to refine, while blocks with criteria lower than the coarsening threshold are flagged to coarsen. To ensure the accuracy of the scheme, mesh refinement is constrained such that the grid resolution changes only by a factor of 2 between any two adjacent blocks, and the minimum resolution is no less than that of the initial mesh. Therefore, a check must be performed prior to refinement to eliminate any cases that violate such constraints. During mesh refinement, each “parent” block is divided into multiple “children” blocks with the same number of cells, thereby increasing the spatial resolution in the region of interest. This process is reversed for over-resolved regions, where multiple “sibling” blocks are coarsened into their parent. Standard multigrid-type restriction and prolongation operators are used to evaluate solution on all blocks resulting from the coarsening or division processes.

Lastly, a hierarchical tree-like data structure is used to keep track of mesh refinement
Figure 3.3: Illustration of refinement and coarsening of an $8 \times 8$ block, during (i) anisotropic AMR in $\xi$, (ii) anisotropic AMR in $\zeta$ and (iii) isotropic AMR cell division. Their geometrical relationship are also represented.

and connectivity between solution blocks. Example of such tree structure is depicted in Figure 3.2 for isotropic mesh refinement. Leaves on the trees represent all blocks on the current mesh, while the ancestor nodes and the branches represent the division processes or refinement histories which lead to the current mesh. The root of each tree, therefore, represents a block on the initial mesh. The data structure may contain multiple trees, whose roots are stored in an indexed array data structure. Recursive traversal of the multi-tree structure can be used to determine block connectivity on the physical domain. However, in order to reduce overhead associated with accessing solution information from adjacent blocks, the neighbors of each block are computed and stored after each AMR so that direct access is possible. The hierarchical tree data structure has the advantages in readily permitting local mesh refinements. Local modifications to the multi-block mesh can be performed without re-gridding the entire mesh and re-calculating all solution block connectivities.
3.1.1 Anisotropic Adaptive Mesh Refinement

The block-based AMR technique described above readily permits anisotropic mesh refinement, yet most AMR procedures in the past were carried out in an isotropic manner as it is more straight-forward in its implementation. The main distinction between isotropic AMR and anisotropic AMR lies in the block division process. For a 2D Cartesian mesh, isotropic AMR divides each parent block into four children, which is equivalent to doubling the mesh resolution in both $\xi$ and $\zeta$, where $\xi$ and $\zeta$ are the computational coordinates of the body-fitted computational block or domain of interest, assuming a standard curvilinear coordinate transformation between the physical coordinates of the body-fitted mesh and the transformed coordinates [78]. However, this is less flexible, and certainly not the most efficient for flows where solution varies much rapidly in one or more directions than the others. In comparison, the anisotropic AMR technique proposed in this thesis allows each parent block to be divided into two children by splitting the parent blocks in either the $\xi$ or $\zeta$, as illustrated in Figure 3.3. As a result, cell resolution doubles in a coordinate direction of choice, but remains the same in the other direction. Implementation of anisotropic AMR follows the general block-based AMR framework discussed above, with some additional new procedures and data structures to handle the added complexities of the anisotropic grid and block connectivity.

3.2 Solution Block Connectivity

3.2.1 Hierarchical Tree Data-Structure

A flexible binary tree data structure is used to keep track of block connectivity during anisotropic mesh refinement, as compared to a quadtree data structure used in isotropic mesh refinement. Figure 3.4 provides an example of such binary tree structure. Each branch in the tree may indicate a split in either $\xi$ or $\zeta$. Therefore, each tree node are associated with two physical levels in $\xi$ and $\zeta$ respectively, which could vary with the same binary tree level. A level difference of 1 in either direction corresponds to a resolution change of a factor of 2 in the same direction. Furthermore, the sector information of each
Figure 3.4: Computational mesh with 6 anisotropic refinement levels originating from one initial block.

block must be specified for both $\xi$ and $\zeta$ to uniquely identify a block in relation to its neighbors and ancestors. The split, level and sector information are stored as additional attributes in an adaptive block object for every node in the binary tree. Lastly, the efficiency $\eta$ for both isotropic and anisotropic AMR are defined as $1 - \frac{N_{\text{cells}}}{N_{\text{uniform}}}$. For anisotropic AMR, $N_{\text{uniform}}$ is the total number of cells on an isotropic uniform mesh consisting of cells at the same level as the higher of the maximum $\xi$- or $\zeta$-level on the current mesh.

3.2.2 Connectivity Rearrangement

The number of $\xi$- and $\zeta$-splits leading to specific $\xi$ and $\zeta$ levels is unique, but the sequence is not. This property of the binary tree structure is exploited in a connectivity rearrangement algorithm which allows the proposed technique to be more robust. Ham et al. [41] pointed out a disadvantage in using a structured hierarchical data-structure for anisotropic AMR, in that coarsening is not always permitted unless it is the exact reverse of the last refinement. This is illustrated in Figure 3.5, where blocks B and D in the first tree structure could not be coarsened in $\xi$, while blocks A and B in the second tree structure could not be coarsened in $\zeta$, although neither of these two scenarios introduce any violations physically. This greatly reduces the flexibility and efficiency of
anisotropic AMR especially during unsteady simulations. To overcome this limitation, a connectivity rearrangement algorithm has been developed and described herein.

Consider the first tree structure in Figure 3.5 to allow the coarsening of blocks B and D in $\xi$, the $\xi$ refinement must take place after the $\zeta$ refinement. That is effectively accomplished by switching the $\xi$ branch at the first level with the $\zeta$ branch at the second level, so that the first tree structure is changed into the second tree structure without altering the current mesh. The actual switching is rather straightforward, but some neighboring blocks can never be made siblings regardless of the orders of anisotropic refinements. Cases in which connectivity rearrangement can be performed satisfy the following:

1. the two blocks must share the same root;
2. the two blocks must be at the same $\xi$ level and $\zeta$ level;

Table 3.1: Pseudo-code for the recursive algorithm used in connectivity rearrangement

<table>
<thead>
<tr>
<th>rearrangeConnectivity(block_ptr1, block_ptr2, desired_split)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Do block_ptr1 and block_ptr2 satisfy conditions 1-3?</td>
</tr>
<tr>
<td>Yes: Continue</td>
</tr>
<tr>
<td>No: Abort</td>
</tr>
<tr>
<td>Do block_ptr1 and block_ptr2 have a common grandparent?</td>
</tr>
<tr>
<td>Yes: Continue</td>
</tr>
<tr>
<td>No: rearrangeConnectivity(block_ptr1.parent, block_ptr2.parent)</td>
</tr>
<tr>
<td>Is split_of_grandparent the desired_split?</td>
</tr>
<tr>
<td>Yes: Switch split directions between the two levels and Return</td>
</tr>
<tr>
<td>No: Abort</td>
</tr>
</tbody>
</table>
Chapter 3. Anisotropic AMR for Body-Fitted Multi-Block Meshes

Figure 3.6: The same mesh as shown in Figure 3.4 with an initial tree structure and the new tree structure after connectivity rearrangement such that blocks B and D are siblings in $\zeta$. Branches which have been altered are in red.

3. the sectors of the blocks must correspond to their spatial relations to each other (e.g., for blocks to be coarsened in the $\xi$ direction, the east neighbor must be an east sector, while the west neighbor must be a west sector).

Examples shown in Figure 3.5 represent the simplest of such cases, and the neighboring blocks of interest have a common grandparent block. Hence connectivity rearrangement involves simply switching the “splits” of the current branch with the parents’ branch. For cases which are more complicated, the algorithm ascends the tree until the first common ancestor node is found, referred to as the “bridge”. Subsequently, the “split” directions are switched two levels at a time, until the split direction originally at the bridge becomes the immediate linkage between the two blocks of interest. This is carried out in a recursive procedure, which is outlined in the pseudo-code of Table 3.1. Figure 3.6 illustrates the tree data-structure is modified after connectivity rearrangement is performed involving more than 2 binary tree levels.

3.2.3 Computation of Solution Block Connectivity

A new neighbor searching algorithm has also been developed in conjunction with the binary tree structure to compute connectivity of blocks in the physical space. The algorithm an approach similar to those described in other literature [45, 13] and for isotropic
AMR [10]. A common ancestor between the current block and the neighbor is first identified as the “bridge”, which the algorithm crosses to reach the branch containing the neighbor, and subsequently descends to the appropriate leaf node. Each neighbor is associated with a search direction vector (e.g., (1, 0) for east, (−1, 1) for north-west etc.). This search direction, along with the sector information of the neighbor blocks, are used to compute the path of ascent and descent within the binary tree. The search algorithm does not assume an a priori knowledge of the connectivity information before each refinement, so that problems would not be introduced when coarsening is involved. Furthermore, recursion is used to simplify and generalize the search implementation.

The main search function identifies the bridge as a sibling in the search direction. If no sibling is found before reaching the root and if there is a root in the search direction,
then the neighboring root is identified as the bridge. The root block connectivity are computed and stored at the start of the simulation using an algorithm which compares the vertexes of all blocks (see Gao [10]). However, all siblings in anisotropic AMR are face neighbors. Hence for corner neighbors, whose search directions are represented as \((di, dj)\), the search identifies the first bridge encountered in either \((di, 0)\) or \((0, dj)\), after which a new search commences for a face neighbor in the remaining direction. Examples of two search paths are depicted in Figure 3.7. In the second example illustrating the search of a northwest neighbor, the north neighbor is first identified and shaded as the intermediate block. A new search then identifies the west neighbor of the intermediate block as the bridge to the northwest neighbor of the original block.

Once the bridge is found, the next step is to descend to the node corresponding to the neighbor block. This node is not necessarily a leaf, as in Figure 3.7a, where the block returned is two levels below the leaf. In fact, a node is returned by the search function if there is insufficient information to identify the next child to visit. In the case of Figure 3.7a, either of the immediate children of the returned node could be the sought west neighbor because they are finer in the \(\zeta\)-direction. This is similarly the case if the search direction is in \(\zeta\) (i.e. north and south), and the current neighbor node have children which are finer in \(\xi\). This holds during the second stage of a corner neighbor search, as in the case of Figure 3.7b when the west neighbor of the intermediate block is returned. However, the ambiguity no longer exists upon returning from the second stage. Therefore, the search function always returns a leaf node at the end of a corner neighbor search.

The recursive search function is called by another top level function for each of the used blocks in the binary tree. If the returned neighbor node is a leaf, then no further work is required besides storing the connectivity information. If a block has two finer face neighbors, further descent may be required for each neighbor for they may still be split numerous times along the search direction, as in the case with Figure 3.7a.
3.3 Anisotropic Mesh Refinement Procedures

Anisotropic division and coarsening of blocks in the $\xi$ and $\zeta$ directions may be carried out sequentially, where AMR in each direction follows the same procedures as those in isotropic AMR. However, such an approach proved to lead to a “deadlock” situation during unsteady anisotropic AMR for the 2D shock-box simulations, as shown in Figure 3.8a. Figure 3.8b illustrates the formation of this deadlock situation. While blocks A1, A2 and A3 are meant to be coarsened into one block, and blocks B1, B2 and B3 into another, the final resulting mesh does not cause any conflicts. However, when blocks A1 and A2 are coarsened in $\xi$ as an intermediate step, it causes a resolution conflict with B2. Similarly, a conflict in $\zeta$ results if B1 and B2 are coarsened first. Hence the six blocks were locked in a position where coarsening could not take place. This in turn prevents surrounding blocks in Figure 3.8a from coarsening. The deadlock situation is a by-product of connectivity rearrangement, in which the refinement history is re-written in such a way that it could not be reversed without exceeding the maximum resolution difference between two neighbors in the intermediate step.

This deadlock situation can be avoided by eliminating the intermediate step and simultaneously coarsening in both $\xi$ and $\zeta$. Hence the following anisotropic AMR procedure is proposed and used here:

1. Directional refinement criteria in both $\xi$ and $\zeta$ are calculated for all blocks, according to which appropriate blocks are flagged to either refine or coarsen in the respective direction;

2. Conflict-checking procedure is carried out to ensure refinement and coarsening for all flagged blocks are permissible;

3. Connectivity rearrangement is performed to ensure parents of all blocks flagged coarsen are split in the appropriate direction;

4. Reduce all blocks flagged coarsen into their parent blocks;

5. Split all blocks flagged refine in $\xi$ into their west and east children;
Chapter 3. Anisotropic AMR for Body-Fitted Multi-Block Meshes

(a) Example of a deadlock situation occurred during an unsteady shock-box simulation. Shown above is the density distribution at $t = 0.75$ s with initial conditions $p_l = 101.325$ kPa, $\rho_l = 1.225$ kg/m$^3$ and $p_r = 405.300$ kPa, $\rho_r = 49$ kg/m$^3$, where the grid appears out-of-line with the solution.

(b) An illustration of how the above deadlock situation arises due to connectivity rearrangement. Block boundaries of siblings are represented by dashed lines.

Figure 3.8: Illustration of the deadlock situation during coarsening.

6. Split all blocks flagged refine in $\zeta$ into their south and north children;

7. Compute and store the new neighbor information;

8. Update geometry and solution information in ghost cells through message passing.

During the refinement and coarsening of a block, the tree structure is first updated, before new grids are generated and solution are transferred to the new block(s). Note that the constraint of maximum resolution change across block interfaces may be violated before all flagged blocks are refined or coarsened. However, this constraint only needs to be reinforced during message passing at the end. Connectivity rearrangement relies on the use of correct neighbor information, hence it must be performed before refinement.
or coarsening of any block. Note also that anisotropic AMR could lead to isotropic refinement if a block is flagged to refine in both $\xi$ and $\zeta$.

### 3.3.1 Refinement Criteria

During anisotropic AMR, each block will be assigned two separate refinement flags for $\xi$ and $\zeta$ respectively, which indicate whether the block should be refined, coarsened or remain unchanged in each direction. To determine these flags, one or more refinement criteria are required for each block. The most effective refinement criteria are based on solution error measures, which do not only provide direct indications to whether the solution features in a local region are sufficiently resolved, but also the coordinate direction in which the solution is under-resolved. Error-based refinement criteria could be obtained via approximation by either $p$-refinement or $h$-refinement \cite{79, 80, 81}. However, refinement criteria of this type still require further research. Hence a less sophisticated refinement criterion based on the physics of the flow is considered for the purpose of this thesis.

For Euler equations, isotropic AMR has been directed by refinement criteria based on the magnitude of the density gradient, compressibility and vorticity, which would provide indications of any shock waves, contact surfaces or shear waves present. These measures are scaled by the area of the cell and normalized by the magnitude of the solution variables involved. The refinement criteria for anisotropic AMR must provide a separate indicator for the solution behavior in each direction. Partial derivatives of the primitive solution variables with respect to each coordinate direction would be an intuitive choice. Following a similar scaling and normalization in calculating the isotropic refinement criteria, expressions for the directional refinement criteria would be as follows:

\[
\epsilon_\xi = \frac{1}{|u|} \left( \nabla \cdot \Delta \vec{X} \right), \quad \epsilon_\zeta = \frac{1}{|u|} \left( \nabla \cdot \Delta \vec{Y} \right). \tag{3.1}
\]

In Equation (3.1), $u$ represents any solution variable in general, while $\Delta \vec{X}$ is the vector difference between the mid-points of the east and west faces of the cell, and $\Delta \vec{Y}$ is the vector difference between the mid-points of the north and south faces of the cell.
For body-fitted meshes, the gradient vector, $\vec{\nabla}u$, is then projected onto the average dimensions of the cell in curvilinear coordinates to indicate the normalized solution change across the cell along each coordinate direction. Therefore, $\epsilon_\xi$ and $\epsilon_\zeta$ provide a good indication to the needs of refinement by relating the physics of the solution to spatial resolution in the direction of interest. For the single-variable advection-diffusion equation, Equation (3.1) simply involves the gradient of the solution $u$. For the Euler equations, the gradient of the density is used as the refinement criteria by default unless otherwise specified. Approximations to the gradient vectors are obtained via a second-order unlimited least-squares reconstruction.

### 3.3.2 Conflict Checking

The conflict checking procedure essentially revises the list of refinement flags to eliminate any cases of refinement or coarsening that is non-permissible. Refinement or coarsening is not permissible for two reasons: i) they introduce resolution changes of more than a factor of two between adjacent blocks; ii) a block flagged to coarsen does not have a sibling that is also flagged to coarsen. This section will refer to checking of the first type of conflicts as “level checks”, and checking of the second type as “sibling checks”.

The original conflict-checking algorithm for isotropic AMR examines all possible refinement and coarsening scenarios, and resolves each scenario individually. However, application of the same procedure in anisotropic AMR is significantly more complex as blocks could have different resolutions in $\xi$ and $\zeta$. The total number of resolution conflicts has been summarized in Table 3.2 according to a comprehensive list compiled previously by Chu [82]. The number of neighbors per face is summarized under $N_{\text{neighbors}}$, the possible combination of refinement flags in each direction (i.e. refine, coarsen or do nothing) under $N_{\text{flags}}$, and the possible combination of resolution changes before refinement under $N_{\text{resolutions}}$.

Note that the tally in Table 3.2 assumed that AMR in $\xi$ and $\zeta$ are carried out separately. In the present implementation, coarsening in $\xi$ and $\zeta$ simultaneously removes the limitation associated with the sequence of AMR in different directions, which makes the algorithm more robust, but also increases the number of conflicting scenarios for
Figure 3.9: Examples of conflicting scenarios during $\xi$-refinement. The block with bold boundaries represent the block being checked in relation to its neighbors, while changes resulted from conflict resolution are marked in red.

Sibling checks that are not accounted for in Table 3.2. The total number of scenarios will increase dramatically in three dimensions, which renders this hard-coding approach impractical. It is then apparent that a more general conflict checking routine is necessary.

Level Checks

Level checks for $\xi$ and $\zeta$ refinement and coarsening are independent, but are each carried out in a similar manner. By assigning values of “+1”, “0” and “-1” to denote “refine (R)”, “no change (N)” and “coarsen (C)”, level difference resulting from AMR could then be calculated for each direction, such that if the resultant level difference is greater than 1, there is a resolution conflict. There are situations where the conflict cannot be resolved unless the flags for both blocks are changed to “N”. They are when the level difference resulting from AMR is 3 as in Figure 3.9a. This type of conflict is resolved first before any others. Otherwise, when a block flagged “C” conflicts with one of its neighbors as in Figure 3.9c and Figure 3.9d, it would be flagged “N” instead; when a block flagged “N”

Table 3.2: Conflicts-Checking Scenarios

<table>
<thead>
<tr>
<th>Face</th>
<th>$N_{\text{neighbors}}$</th>
<th>$x$ refinement or coarsening</th>
<th>$y$ refinement or coarsening</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$N_{\text{flags}}$</td>
<td>$N_{\text{resolutions}}$</td>
<td>total</td>
</tr>
<tr>
<td>East, West</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>East, West</td>
<td>2</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>South, North</td>
<td>1</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>South, North</td>
<td>2</td>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>Corners</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>
conflicts with a neighbor flagged “R”, the block will be automatically forced to refine as in Figure 3.9b. This level checking algorithm leads to a more general implementation which avoids counting all possible combinations of resolution changes and refinement flag values.

Sibling Checks

When AMR in $\xi$ and $\zeta$ are carried out simultaneously, additional scenarios emerge as siblings of blocks may appear or disappear due to refinement or coarsening in the other direction. Some of the representative scenarios for sibling checks are depicted in Figure 3.10. Cases (a)-(c) in Figure 3.10 present the most straightforward scenarios. Cases (d) and (e) are scenarios where coarsening in $\xi$ is enabled by the mesh change in $\zeta$.

Therefore, in the generalized conflict-checking procedure, “sibling(s)” of a block can simply be one or more neighbors in the appropriate direction. However, it should be noted that since a block can only be coarsened or refined once in each direction, the appropriate sibling(s) of a block must be at the same level as the block along the coarsening direction regardless of the siblings’ flags. If this is not true, as in Case (k) of Figure 3.10, or if the sibling(s) are not flagged to coarsen, then coarsening is not permitted. These scenarios are the first to be checked for.

Furthermore, the block to be coarsened and its siblings must be at the same level in the direction other than the direction of coarsening at the end of AMR. This guarantees that a sibling block with the same dimensions as the block to be coarsened will result by refining or coarsening in the other direction first, as in Cases (d) and (e) in Figure 3.10. In cases like (f) and (g) in Figure 3.10, blocks flagged “N” are forced to refine to meet this requirement. However, for Cases (h) and (i) of Figure 3.10 where refinement could not be forced, coarsening is not permitted, because it does not make sense to coarsen two siblings when they need to satisfy different resolution requirements in the other direction.

It was observed in some test cases that the outcome of the second sibling checks, which involve the resolution change in the direction other than the direction of coarsening, is very sensitive to flags of neighboring blocks. Therefore, ordering of the checking procedures becomes important. In the current implementation, the first sibling checks
Figure 3.10: Examples of permitted and non-permitted coarsening scenarios. Flags for the $\xi$-direction are in blue, and flags for the $\zeta$-direction are in red. Boundary between two sibling blocks are represented by dashed lines. (a-c) $\xi$-coarsening is permitted without modification of flags; (f, g) $\xi$-coarsening is permitted by forcing sibling to refine in $\zeta$; (h, i) $\xi$-coarsening is not permitted due to conflicts in $\zeta$; (j) $\xi$-coarsening is permitted after conflict in $\zeta$ is resolved; (k) $\xi$-coarsening is not permitted because siblings are too fine; (l) Coarsening is not permitted because direction for coarsening is ambiguous.
and the level checks are placed in an inner loop which iterate until the flags no longer change, before the second sibling checks are carried out. The entire conflict-checking procedure is placed inside an outer iteration which exits when no more conflicts are detected. Connectivity of blocks or their parents are rearranged in the end to ensure all “siblings” are properly connected within the tree.

### 3.4 Grid Generation and Solution Transfer

\[
\begin{align*}
\bar{x}(\xi + \Delta \xi, \zeta + \Delta \zeta) &= \bar{x}(\xi, \zeta) + \frac{\partial \bar{x}}{\partial \xi} \bigg|_{\xi, \zeta} \Delta \xi + \frac{\partial \bar{x}}{\partial \zeta} \bigg|_{\xi, \zeta} \Delta \zeta \\
&\quad + \frac{1}{2} \left( \frac{\partial^2 \bar{x}}{\partial \xi^2} \right)_{\xi, \zeta} (\Delta \xi)^2 + 2 \left( \frac{\partial^2 \bar{x}}{\partial \xi \partial \zeta} \right)_{\xi, \zeta} \Delta \xi \Delta \zeta + \frac{\partial^2 \bar{x}}{\partial \zeta^2} \bigg|_{\xi, \zeta} (\Delta \zeta)^2 \\
&\quad + \mathcal{O}\left((\Delta \xi)^3, (\Delta \zeta)^3\right). \tag{3.2}
\end{align*}
\]

Figure 3.11 illustrates the grid generation process at the cell level, where a new node is added between each pair of adjacent nodes on the vertical or horizontal faces in the coarse cell. For cells lying on the physical boundaries, locations of new nodes are determined using path-length splines which describe the geometrical shapes of the boundaries. The boundary splines, as depicted in Figure 3.12a, are created on the initial mesh at start-up, and they must be copied from the parent blocks to the children blocks during refinement. Splines for interior block boundaries are set to zero by default,
Figure 3.12: (a) Illustration of the “true” domain and its division into multiple blocks; and (b) the low- and (c) the high-order mesh representations for the corner block. Straight line segments are sufficient for representing interior block boundaries. The extension (ghost) boundaries provide complete information about the properties of the neighboring splines [83].
representing a straight line. Hence for body-fitted meshes, new interior nodes could be approximated either by the mid-points between existing nodes on the coarse mesh, or by using the grid metrics. The grid metrics, which maps between physical domain \( \vec{x} = (x, y) \) and a uniformly spaced Cartesian computational grid, \((\xi, \zeta)\), could be approximated by Taylor series expansion in Equation (3.2). Values of the derivative terms are computed using second-order central difference approximation, and third-order forward/backward difference at block boundaries. New node locations are then computed by combining the Taylor series expansion from existing nodes on the coarse mesh using a second-order averaging procedure. The mid-point approach is used by default in grid-generation, while the metrics is made available if more accurate grid representation is necessary during convergence studies. Lastly, mesh stretching parameters in the children blocks are directly duplicated from the parent blocks.

Apart from the above procedures which are common between second- and higher-order schemes, an additional step is necessary for higher-order schemes to store the extended spline information during mesh refinements. For the second-order schemes, a linear spline is sufficient to provide an accurate representation of the physical boundaries as shown in Figure 3.12b. For higher-order schemes, however, higher-order boundary elements are required as depicted in Figure 3.12c, which cannot be obtained through nodal values of boundary geometry stored only in the interior cells. As a result, extended splines representing shapes of the ghost cell edges are required on all external block boundaries [83]. The extended spline information from the parent block must be stored for the appropriate boundaries of the children block, in a similar manner as the regular splines. Additional consistency checks are necessary to ensure the ghost cell geometry from, especially those from a coarser neighbor, agrees with those specified by the corresponding extended splines. Special boundary treatments during reconstruction also need to be considered for corner blocks, so that the chosen reconstruction supporting stencils obey the conditions imposed by all splines and extended splines [83]. Since these procedures do not differ between anisotropic AMR and the original isotropic approach, further details are not discussed here (please refer to [83]).

Solution prolongation refers to the process of solution transfer from the coarser parent
block onto its finer children blocks during anisotropic refinement. The most straightforward way to carry out this procedure is via direct injection, in which case the cell-averaged solutions in the coarse cells are directly assigned to the corresponding finer cells in the children blocks. Accuracy information is lost during direct injection as it is a first-order accurate operation. However, since mesh refinement is followed by explicit time-marching or implicit Newton iterations, the lost accuracy will be recovered.

Coarsening essentially reverts the cell division process, during which the coarse mesh is generated by removing every other nodes from the fine mesh, and area-weighted sum of the finer cells solution averages is assigned to the coarse cell. The boundary splines and mesh stretching parameters are again copied from the children blocks into the parent block.

3.5 Message Passing

3.5.1 High-Order Solution Prolongation

At interfaces between blocks of equal resolutions, solution values are directly assigned from cells to cells. Otherwise, restriction and prolongation procedures are required to evaluate the ghost cell solution values. The restriction procedure used during message passing is identical to what is done during interior cell solution transfer after coarsening.
However, the prolongation procedure in message passing requires more accuracy, because the error introduced cannot be reduced via time-integration or implicit iterations. For a second-order scheme, prolongation involves evaluating the solution at centroids of the corresponding ghost cells using the piece-wise linear reconstruction. To preserve the accuracy for higher-order schemes, reconstruction in the coarse interior cell, $W^k_c$, is integrated according to

$$
\bar{W}_f = \frac{1}{A_f} \int_{A_f} W^k_c(\vec{X})dA
$$

to obtain appropriate solution averages, $\bar{W}_f$, for the ghost cells, where $A_f$ is the ghost cell area. To achieve more efficient evaluation of Equation (3.3), Ivan [83] proposed to convert the area integral into a contour integration in the counter-clock-wise direction using the following method:

$$
(x_\text{p}^{1},y_\text{p}^{2}) = \frac{1}{(p_1 + 1) A_f} \int_{A_f} (x - \bar{x}_{i,j})^{p_1+1} (y - \bar{y}_{i,j})^{p_2} dx dy
$$

$$
= \frac{1}{(p_1 + 1) A_f} \int_{A_f} (x - \bar{x}_{i,j})^{p_1+1} (y - \bar{y}_{i,j})^{p_2} dy
$$

$$
= \frac{1}{(p_1 + 1) A_f} \sum_{l=1}^{N_f} \oint_{S_l} (x - \bar{x}_{i,j})^{p_1+1} (y - \bar{y}_{i,j})^{p_2} dy.
$$

The above formulation may be applied to each term in the reconstruction polynomial individually. Subsequently, Equation (3.6) could in turn be evaluated using Gauss quadrature as such [83]:

$$
\mathcal{I} = \oint_{S_l} (x(s) - \bar{x}_{i,j})^{p_1+1} (y(s) - \bar{y}_{i,j})^{p_2} \frac{dy(s)}{ds} ds = \oint_{S_l} F(s) ds = L_l \sum_{m=1}^{N_G} (\omega F)_m,
$$

where the function $F(s) = (x(s) - \bar{x}_{i,j})^{p_1+1} (y(s) - \bar{y}_{i,j})^{p_2} \frac{dy(s)}{ds}$ is evaluated at each of the $N_G$-Gauss quadrature points and and weighted by the quadrature weights $\omega$. For integration involving general curvilinear boundaries as depicted in Figure 3.13, $L_l$ is the total arc length and the locations of the Gauss integration points are computed by taking the corresponding abscissas along the arc length of the integration path, $\frac{dy(s)}{ds}$ is evaluated either numerically or analytically depending on whether an analytical expression for
$y = y(s)$ is available \[83\]. The framework for this contour integration method has already been implemented for the CENO finite-volume scheme for isotropic meshes, and is directly extended to anisotropic mesh adaptation without significant modifications.

In order to preserve conservation, the prolonged solution average in the fine cells must satisfy the following property:

$$
\bar{U}_c A_c = \bar{U}_{f1} A_{f1} + \bar{U}_{f2} A_{f2}.
$$

(3.8)

This relationship is automatically satisfied for second-order accurate reconstruction, as the solution average over any sector of the cell is simply the linear solution reconstruction evaluated at the centroid of the sector, such that no integration is required. This is not the case for high-order schemes, which further explains why more accurate spline representation of the physical boundaries is required. As Figure 3.13 shows, the more accurate spline boundary representation in the high-order scheme ensures that areas of the fine cells, $A_{f1}$ and $A_{f2}$, sum up to the area of the coarse cell. This is to ensure that the error introduced by the geometric representation of the cells, if there is any, shall be beyond the accuracy of the reconstruction and its integration. However, as the integration was performed using the primitive solution variables, an additional step is taken to distribute error in the resulted conserved solution evenly among the fine cell solution average. As a result, the average conserved solution in a fine cell is given by \[83\]

$$
\bar{U}_f = U(\bar{W}_f) + \frac{1}{A_c} \left[ \bar{U}_c A_c - (U(\bar{W}_{f1}) A_{f1} + U(\bar{W}_{f2}) A_{f2}) \right].
$$

(3.9)

3.5.2 Message Passing for Anisotropic AMR

For anisotropic AMR, block resolutions in the $\xi$ and $\zeta$ directions are independent, hence a block may be finer than its neighbor in $\xi$, yet be of the same resolution or coarser in $\zeta$, and vice versa. This results in 9 different types of resolution changes across block interfaces, which are summarized in Figure 3.14. 3 out of these 9 scenarios are isotropic. Furthermore, Cases (a), (b), (d) and (e) involve prolongation or restriction in one of the two directions, hence the procedures are almost identical to those in isotropic AMR with
Figure 3.14: Summary of the different message passing scenarios in anisotropic AMR with graphical illustration of the prolongation/restriction required.

Slight modifications. The two remaining cases involve message passing from blocks which are finer in one direction, but coarser in the other. Hence message passing for these two cases would consist of prolongation in one direction, followed by restriction in the other direction. Take Case (c) for example, the north sub-cell values of both cell \((i, j)\) and cell \((i + 1, j)\) are first obtained through prolongation, following which solution in north sectors of both cells are recombined via restriction.

When high-order prolongation is carried out, accurate ghost cell information from the same block are needed to accurately reconstruct the interior cell solutions. Hence the ghost cell solutions should be up-to-date before the reconstruction. Message passing for isotropic AMR is always carried out first between blocks with no resolution change and from finer blocks to coarser blocks, before solution from coarser blocks are message-passed to finer blocks via prolongation. However, an equivalent sequence of operation

Table 3.3: Values in a ghost cell during an iterated message passing process.

<table>
<thead>
<tr>
<th>Number of Iterations</th>
<th>Ghost cell value</th>
<th>% Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.26080029835558076</td>
<td>(3.05 \times 10^{-6})</td>
</tr>
<tr>
<td>2</td>
<td>0.260799503976414</td>
<td>(6.61 \times 10^{-8})</td>
</tr>
<tr>
<td>3</td>
<td>0.2607995212053638</td>
<td>(1.47 \times 10^{-9})</td>
</tr>
<tr>
<td>4</td>
<td>0.2607995208225586</td>
<td>(2.84 \times 10^{-11})</td>
</tr>
<tr>
<td>5</td>
<td>0.2607995208299719</td>
<td></td>
</tr>
</tbody>
</table>
Figure 3.15: The values recorded in Table 3.3 is of cell (0,17) in the northwest corner of the block highlighted in red, which has a north neighbor that is coarser in $\xi$, but finer in $\zeta$.

could not be determined for anisotropic AMR, because Cases (c) and (f) in Figure 3.14 require both prolongation and restriction. As a result, ghost cell values after one step or operation of message passing are inevitably incorrect, but may be corrected by repeating the message passing procedure in an iterative manner. Table 3.3 provides a record of the value of a ghost cell, which belongs to Case (c) in Figure 3.14 after a given number of message passing iterations. The grid and solution distribution for which the values are recorded is shown in Figure 3.15. By comparing the ghost cell value between the first and second iteration, the inconsistency is quite evident. The change in the ghost cell value seems to be reduced to a more acceptable level after about 5 iterations. Based on this result and other similar tests, 5 iterations of message passing is currently put in place as a temporary solution to overcome this difficulty in evaluating the ghost cell solution values when message passing.

The error in the ghost cell values is not significant, and it has not introduced any noticeable problems in unsteady simulation via explicit time-marching. Accuracy assessments and convergence studies are only performed with steady-state simulations in this thesis, in which case the ghost cell values converge after sufficient number of time steps. An iterative message passing procedure is therefore not essential. However, convergence of implicit methods can be significantly affected by the presence of this error. In an implicit time-accurate simulation using NKS with dual-time-stepping, during which the data in Table 3.3 was recorded, a total of 183 GMRES iterations were required per implicit time step if only one iteration of message passing was used, comparing to a total
of 18 GMRES iterations if 5 iterations of message passing were used. In some cases, it is also possible that the solution would not converge at all without the additional iterations during message passing.

The iterative message passing procedure described above only serves as a temporary remedy to the ghost-cell evaluation problem, as the iterative process is also associated with a significant computational overhead cost, and the method in itself is clearly not elegant. Therefore, a more permanent revision to the message passing routine should be considered in the near future.
Chapter 4

Numerical Results

In this chapter, the proposed block-based anisotropic AMR scheme is applied to the solution of a number of test cases governed by the advection-diffusion equation and the Euler equations in two dimensions. The implemented numerical framework is tested thoroughly by examining both unsteady and steady-state problems, using solution methods of explicit time-marching and implicit Newton-Krylov-Schwarz respectively. Most of the simulations have been carried out using high-order CENO reconstruction to show that anisotropic AMR can indeed be applied with $k$-exact and second-order least squares reconstructions. Lastly, performance of anisotropic AMR is evaluated by comparing the simulation results against those with isotropic AMR. The effectiveness of anisotropic AMR is further evaluated by assessing solution accuracy on the resulting mesh for selected problems.

4.1 Advection Diffusion in Rectangular Channel

A problem involving both advection and diffusion in a rectangular channel is considered as the first validation case for the proposed anisotropic AMR solution scheme. The rectangular channel has an inflow at $x = 0$ described by $\sin(\pi y)$. A constant diffusion field with $\kappa(x, y) = 0.01$ smears the solution as a uniform velocity field $\vec{V}(x, y) = (0.1, 0)$ carries the inflow towards the end of the channel. A uniform initial condition of $u = 0$ is imposed at $t = 0$, which allows this case to be examined both as an unsteady and
a steady-state problem. The unsteady evolution of the solution will be obtained using explicit time-marching; whereas the steady-state solution alone is separately obtained through the implicit Newton-Krylov-Schwarz method. Both simulations consist of $12 \times 12$ blocks, as well as a fourth-order accurate spatial reconstruction (i.e. $k = 4$).

### 4.1.1 Unsteady Results

The mesh for this simulation begins with four blocks, which undergo five levels of initial AMR before the simulation starts. Successive refinements are carried out every 200 time steps, although a higher refinement frequency of 50 is used before $t = 1.5$ s for anisotropic AMR for its lower mesh density. The CFL number is 0.5. The solution near $t = 0$ s is strongly anisotropic as the inflow creates a sharp gradient in the $x$-direction, but the variation becomes more gradual as $t$ increases. This change of the solution distribution is well captured by anisotropic AMR as observed in snapshots of the solution at $t = 0.75$ s, $t = 1.5$ s, $t = 5.0$ s, $t = 10.0$ and $t = \infty$ in Figure 4.1. The refinement and coarsening thresholds are 0.80 and 0.30 for anisotropic AMR, and 0.80 and 0.15 for isotropic AMR.

As expected, the advantage of anisotropic AMR is the most noticeable at $t = 0.75$ s. Isotropic AMR constrains the aspect ratios of the blocks according to those on the initial mesh, hence the mesh resolution in $y$ is forced to increased with that in $x$, resulting in 32 blocks near the west boundary of the channel at $x = 0$. In the mesh generated by anisotropic AMR, the same resolution in $x$ is achieved with only 12 blocks in the $y$ direction, whose dimensions in $y$ are flexibly determined by the variation of the solution along the $y$-axis. Overall, a total of 42 blocks with 6048 cells were used in the anisotropic mesh with a refinement efficiency of 0.96, while a total of 178 blocks with 25632 cells were used in the isotropic mesh with a refinement efficiency of 0.83. Hence anisotropic AMR is able to provide a saving of 76% in terms of computational cost, assuming the cost scales directly with the mesh size.

As the solution evolves in time, it diffuses and becomes less anisotropic overall. However, some regions have anisotropic variations locally when the solution change in $y$ becomes more dominant, as seen near the north and south boundaries of the rectangular channel near $x = 0$. These local anisotropic variations are very well captured by
Chapter 4. Numerical Results

Figure 4.1: Comparison of isotropic and anisotropic AMR in simulating advection and diffusion in a rectangular channel as a time-variant problem, with CENO $k = 4$ reconstruction.
(a) Convergence history for isotropic AMR

(b) Convergence history for anisotropic AMR

**Figure 4.2:** Convergence histories for explicit unsteady simulation with both AMR methods.

anisotropic AMR, as seen in the snapshots for $t = 1.5$ s, $t = 5.0$ s, $t = 10.0$ s and at steady-state. The refinement criteria used in isotropic AMR involves the norm of the gradient, which is a general indicator for both the $x$ and $y$ components of the gradient. Hence for regions in the domain where the gradient in one direction is not so strong, it may reduce the effect of the gradient in the other direction and prevent the block
from refining. It is noticed in the later snapshots, that isotropic AMR appears to favor refinements of regions where the components of the gradient are more similar in magnitude, over regions that have somewhat stronger gradients in one direction but very weak gradients in the other. Therefore, anisotropic AMR is observed to better conform to the variation of the solution in cases where the solution is not strongly anisotropic. Upon reaching steady-state, the percentage savings introduced by anisotropic AMR is 46% when compared to the results obtained using isotropic AMR. Lastly, convergence history for the entire unsteady simulation is included in Figure 4.2, where the jump in residual value after each unsteady AMR is represented by a spike in the convergence history.

4.1.2 Implicit Results

The implicit simulation uses the same initial mesh as the unsteady case, but without the initial AMR. A steady-state solution is first obtained on the coarse mesh using inexact Newton’s method, which is then improved by increasing the mesh resolution through AMR and repeating the NKS-AMR cycles until the mesh ceases to change. The refinement and coarsening thresholds for both isotropic and anisotropic AMR are the same as those used for explicit time-marching. The convergence of Newton’s method speeds up as the solution approaches the steady-state solution. Hence the solution process is sped up significantly by first eliminating the transient part of the solution on a coarse mesh before the solution quality is improved via AMR. The steady-state solution and history of convergence for both isotropic AMR and anisotropic AMR are shown in Figure 4.3. The resulting meshes for both AMR are slightly different from those in the explicit simulation, with anisotropic AMR introducing a saving of 48%, because the refinement histories for explicit and implicit solution procedures are quite different. The NKS method appears to work quite well for this problem, where the residual drops at least 12 orders of magnitude in mostly 8–10 Newton steps for both AMR methods. Furthermore, for the fully converged solution in Figure 4.3, a total of 470 GMRES iterations and an average of 47 iterations per Newton step are required using isotropic AMR, whereas a total of 404 GMRES iterations and an average of 45 iterations per Newton step are required using
Chapter 4. Numerical Results

(a) Steady-state solution with isotropic AMR
\( N = 37728, \eta = 0.74 \)

(b) Steady-state solution with anisotropic AMR
\( N = 19584, \eta = 0.87 \)

(c) Convergence history for isotropic AMR

(d) Convergence history for anisotropic AMR

Figure 4.3: Comparison of isotropic and anisotropic AMR in simulating advection and diffusion in a rectangular channel as a steady-state problem, with CENO \( k = 4 \) reconstruction.
Chapter 4. Numerical Results

4.1.3 Accuracy Analysis

An accuracy analysis for the advection diffusion problem in a rectangular channel has been included to more accurately reflect the effectiveness of anisotropic AMR as compared to isotropic AMR and uniform refinements. Constrained reconstructions are enforced on all boundaries for this study to maintain high-order accuracy. Results of the analysis are plotted in Figure 4.4. Accuracy is first assessed on uniform multi-block meshes obtained via 2, 3 and 4 levels of uniform refinements, before driving the solution to steady-state using implicit method. Data points for isotropic and anisotropic AMR are obtained using the implicit approach described in Section 4.1.2, but with 2, 3, 4, or 6 maximum levels of refinement respectively. Refinement thresholds for both AMR methods are 0.6; coarsening thresholds are 0.2 when maximum levels of refinements are 2 and 3, and 0.15 when maximum levels of refinements are 4 and 5. Lastly, from Figure 4.3c and Figure 4.3d, it can be noticed that the final residual upon convergence is not consistent...
Chapter 4. Numerical Results

(a) Solution at $t = 0.225$ s  
(b) Solution at $t = 0.628$ s  
(c) Solution at $t = \infty$

Figure 4.5: Simulation of circular advection in a rectangular box on a coarse mesh without AMR, with CENO $k = 3$ reconstruction.

because the NKS tolerance is a relative residual measure. To ensure the residual is consistently reduced to machine precision for all accuracy studies, Newton’s method is required to iterate for 50 steps unless the residual drops 16 orders of magnitude.

Convergence with uniform refinements in Figure 4.4 is observed to have a slope of 4 for fourth-order spatial accuracy as expected. However, both AMR methods as observed in this study do not introduce any improvements to the convergence of the uniform mesh. On the contrary, they appear to increase the number of cells required without improving the solution accuracy, with anisotropic AMR appearing even more inferior than isotropic AMR. This is largely related to the fact that the advection-diffusion in a rectangular channel is relatively smooth and lacking in scale disparity, hence the error distribution is likely uniform throughout the domain. Effectiveness of AMR is further hampered by the usage of a physics-based refinement criterion, which does not provide a direct indication of solution error in each block. When regions with the greatest error cannot be clearly identified, over-refinement becomes more favorable.

4.2 Circular Advection in Rectangular Box

The second problem considered for the 2D advection-diffusion equation involves pure advection by a constant angular velocity of 5 rad/s. The circular advection takes place within a square box with initially four $12 \times 12$ blocks, where the block interface along
$y = 0$ and $0 \leq x \leq 1$ represents an inflow/outflow boundary. The inflow variation is defined by $\sin^2(4\pi(r - 0.375))$ for $r \in [0.375, 0.625]$ and 0 otherwise. A uniform initial condition of $u = 0$ is again imposed, which creates a discontinuous flow front starting at $\theta = 0$. The flow front makes an angular displacement of $2\pi$ before the solution connects at the inflow/outflow boundary to form a continuous ring of circular flow. Anisotropic AMR is expected to work best when the direction of solution variation is aligned with the coordinate directions. In the extreme case when the solution varies gradient is exactly 45 degrees from the main coordinate axis, anisotropic AMR simply reverts to isotropic AMR. It should be noted that the ideal grid set up for this problem should be a body-fitted mesh that is wrapped in a circle tangential to the flow direction, in which case the flow variation will always be in one of the coordinate directions. Nevertheless, by simulating a circular flow on a Cartesian grid, it becomes interesting to observe how anisotropic AMR attempts to capture the solution for this problem as it goes through all possible angles of variation. A fourth-order accurate $k = 3$ reconstruction is used for both the unsteady and steady-state simulations of this problem. To serve as a reference, coarse-mesh results using 64 blocks are plotted in Figure 4.5. The solution in Figure 4.5 is clearly under-resolved at the flow front, while the solution distribution at steady-state also appears to be diffused with distance due to excessive numerical dissipation.

### 4.2.1 Unsteady Results

Snapshots of the time-variant results have been plotted in Figure 4.6. The CFL number is again 0.5. To capture the discontinuous wave front, both isotropic and anisotropic AMR are allowed a maximum of 8 refinement levels through 2 levels of uniform refinements and 6 levels of initial AMR, followed by subsequent refinements every 10 time steps. Refinement threshold is 0.80 for isotropic AMR, and 0.60 for anisotropic AMR to encourage $x$-refinements near $t = 0$ of the simulation. Coarsening threshold is 0.20 for both AMR methods.

At $t = 0.01$ s, the sharp gradient created by the discontinuous flow front is approximately aligned with the $y$-axis, again allowing anisotropic AMR to be very advantageous. Although both AMR methods have high efficiency of almost 1, anisotropic AMR is able
Chapter 4. Numerical Results

66

(a) Solution at $t = 0.01$ s

(b) Isotropic AMR
$N = 23040, \eta = 0.997$

(c) Anisotropic AMR
$N = 8064, \eta = 0.999$

(d) Solution at $t = 0.157$ s

(e) Isotropic AMR
$N = 35136, \eta = 0.996$

(f) Anisotropic AMR
$N = 29088, \eta = 0.997$

(g) Solution at $t = 0.628$ s

(h) Isotropic AMR
$N = 61488, \eta = 0.993$

(i) Anisotropic AMR
$N = 48240, \eta = 0.995$

(j) Solution at $t = 0.942$ s

(k) Isotropic AMR
$N = 166896,$
$\eta = 0.982$

(l) Anisotropic AMR
$N = 80496, \eta = 0.993$
to introduce an additional 65% saving when compared to isotropic AMR, as shown in Figure 4.6b and Figure 4.6c. The next snapshot occurs at $t = 0.157$ s, which is when the wave front is 45 degrees from both axis. As anticipated, the mesh produced by anisotropic AMR near the flow front is very similar to that by isotropic AMR. Anisotropic AMR is still observed to offer some advantage in optimizing the number of blocks required in regions away from the wave front, which are refined to satisfy the constraint of maximum resolution change across block boundaries. In the next two snapshots, the wave front wave is again aligned with the coordinate axis. This transition is well captured by anisotropic AMR, which also reflects the effectiveness of mesh coarsening after each AMR. In all snapshots up to $t = 0.942$ s, AMR has allowed the flow front to be much better resolved when compared to Figure 4.5. Meanwhile, the convected solution is also able to maintain the shape of the inflow throughout the distance traveled.

After the wave front has reached the outflow boundary at $t = 1.26$ s, the maximum level of refinement is reduced to 7 to avoid over refinement. The refinement threshold for both isotropic and anisotropic AMR is increased to 0.90. Coarsening threshold increased to 0.30 for anisotropic AMR and reduced to 0.05 for isotropic AMR. AMR frequency is reduced to 100 time steps as the solution is time-marched towards steady-state. In the final mesh depicted in Figure 4.6n and Figure 4.6o, anisotropic AMR places blocks with higher aspect ratios at locations where the solution variation is more aligned with

Figure 4.6: Comparison of isotropic and anisotropic AMR in simulating circular advection in a rectangular box as a time-variant problem, with CENO $k = 3$ reconstruction.
Chapter 4. Numerical Results

Figure 4.7: Comparison of isotropic and anisotropic AMR in simulating circular advection in a rectangular box as a steady-state problem, with CENO $k = 3$ reconstruction.
the coordinate axis, and blocks with lower aspect ratios close to the 45 degree lines, thereby offering a saving of 61% in the number of computational cells. However, mesh resolution produced is for the most part similar to that of isotropic AMR in all directions of variations.

4.2.2 Implicit Results

The steady-state solution for the circular advection problem is obtained following a similar approach as described in Section 4.1.2. The coarse-mesh steady-state solution is obtained on 64 blocks resulting from 2 uniform refinements on the initial mesh. Both isotropic and anisotropic AMR are allowed a maximum refinement level of 7, with a refinement threshold to 0.40 and a coarsening threshold to 0.10. The final steady-state solution is plotted in Figure 4.7 along with the convergence history. Convergence of the implicit method in Figure 4.7 is not as smooth as in those in Section 4.1.2. This is possibly due to discontinuity present at the inflow/outflow boundary and the inflow itself, as well as the non-linearity of the CENO scheme which is not accounted by the NKS solver. The mesh for both AMR methods are denser than in the explicit results, again due to the very different refinement histories from the explicit and implicit procedures. However, distributions of the mesh blocks follow a similar trend and the saving introduced by anisotropic AMR is 40% based on the number of cells used in each computation.

4.2.3 Accuracy Analysis

Accuracy assessments have been performed on the steady-state meshes for both the explicit and the implicit results, where the $L_1$, $L_2$ and $L_{\text{max}}$ norms of the solution error are summarized in Table 4.1. When examining the explicit results alone, anisotropic AMR appears less effective in reducing error in the solution, and the case is similar with the implicit results alone. However, if the anisotropic mesh from the implicit result is compared with the isotropic mesh from the explicit result, then it becomes evident that anisotropic AMR does in fact improve the solution accuracy while still offering a reduction of 24% in the required cell-count. It is important to note that this problem is not at
Table 4.1: Accuracy Comparison for Circular Advection Simulation

<table>
<thead>
<tr>
<th>AMR</th>
<th>N</th>
<th>$L_1$</th>
<th>$L_2$</th>
<th>$L_{max}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Isotropic</td>
<td>607536</td>
<td>$1.5 \times 10^{-4}$</td>
<td>$7.26 \times 10^{-4}$</td>
<td>$7.81 \times 10^{-3}$</td>
</tr>
<tr>
<td>Anisotropic</td>
<td>236016</td>
<td>$2.66 \times 10^{-4}$</td>
<td>$1.21 \times 10^{-3}$</td>
<td>$1.17 \times 10^{-2}$</td>
</tr>
</tbody>
</table>

Implicit Results in Figure 4.7

<table>
<thead>
<tr>
<th>AMR</th>
<th>N</th>
<th>$L_1$</th>
<th>$L_2$</th>
<th>$L_{max}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Isotropic</td>
<td>769968</td>
<td>$8.14 \times 10^{-6}$</td>
<td>$4.62 \times 10^{-3}$</td>
<td>$6.67 \times 10^{-4}$</td>
</tr>
<tr>
<td>Anisotropic</td>
<td>460080</td>
<td>$5.01 \times 10^{-5}$</td>
<td>$2.6 \times 10^{-4}$</td>
<td>$3.77 \times 10^{-3}$</td>
</tr>
</tbody>
</table>

Figure 4.8: Simulation of an unsteady shock-box problem on a coarse mesh without AMR, with $k = 3$ CENO reconstruction.

all anisotropic, and that anisotropic AMR uses less than half of the computational cells in isotropic AMR with similar refinement and coarsening thresholds specified. Therefore, if the difference in cell count could be used towards increasing both the maximum refinement level and refinement thresholds for anisotropic AMR, it is believed that anisotropic AMR will likely produce more accurate results than isotropic AMR.

4.3 Shock-Box Simulation

The shock-box problem is an important unsteady test case for the 2D Euler Equations for inviscid compressible gases. The shock-box problem considered herein consists of air initially rest. The left state, initially occupying $x < 0$ and $y < 0$, is at standard atmospheric condition (i.e. $p = 101.325$ kPa, $\rho = 1.225$ kg/m$^3$); and the right state, initially occupying $x \geq 0$ and $y \geq 0$, is at four times the standard atmospheric condition
Chapter 4. Numerical Results

(i.e. $p = 405.300$ kPa, $\rho = 4.9$kg/m$^3$). Air at the interface separating the two states begin interacting at $t > 0$, resulting in shocks propagating from high to low pressure. Air behind the shock abruptly increases in density and gains a velocity towards to the lower-pressure region. An intermediate region results where the air pressure is between the higher and lower pressure regions in the initial states. Air in the intermediate region expands smoothly into the low-pressure region through rarefaction waves which move in opposite directions from the shockwaves. Meanwhile, air in the intermediate region continues to be separated by the original boundary between the two initial states, which moves at the same velocity as the surrounding air in the form of a contact surface.

In regions sufficiently far away from the origin, where the waves interact, the shock-box problem is essentially 1D with non-varying solution in one of the coordinate directions. Therefore, this problem is strongly anisotropic, making it an excellent practical application of the proposed finite-volume scheme with anisotropic AMR. The current shock-box simulation is assumed to take place on an infinite domain, but only the region with $-0.5 \leq x \leq 0.5$ and $-0.5 \leq y \leq 0.5$ is considered. Therefore, reflection boundary conditions are imposed on all four sides of the domain, and the simulation is stopped before any waves move within proximity of the boundaries. Time-accurate solution for the unsteady simulation is obtained via the fourth-order Runge-Kutta time-marching method, with a CFL number of 0.25. Flux evaluation at cell boundaries makes use of the Roe approximate Riemann solver, combined with a fourth-order space-accurate reconstruction (i.e. $k = 3$). To fully capture the discontinuity in the initial states, the simulation begins by performing 1 uniform refinements and 7 initial AMR on a single block consisting of $10 \times 10$ cells. The mesh is updated once every 20 time steps using both isotropic and anisotropic AMR.

The solution distribution from an unsteady simulation on the coarse mesh after 1 uniform refinement but with no AMR, is shown in Figure 4.8. Only are the discontinuities poorly resolved, the shock, contact surface, the rarefaction wave and the shock-interaction patterns could not be distinguished until $t = 0.745$ s. In comparison, the results obtained using AMR in Figure 4.9 details of the shock-interaction could be observed as early as at $t = 0.045$ s.
Figure 4.9: Comparison of isotropic and anisotropic AMR in simulating the unsteady shock-box problem, with $k = 3$ CENO reconstruction.
The required mesh resolution for the shock-box problem, especially near the start of the simulation at \( t = 0.045 \) s, is driven by the rapid solution change at the discontinuities, which could be satisfied by refining the mesh only in the direction of wave propagation along either \( x \) or \( y \). As a result, the performance of isotropic AMR becomes relatively poor by unnecessarily increasing the mesh resolution along the discontinuities, where the solution changes are minimal. In comparison, anisotropic AMR effectively achieves the same mesh resolution as isotropic AMR across all discontinuities, while keeping the mesh coarse in the other direction. In doing so, anisotropic AMR is able to introduce a significant 87\% reduction in the number of computational cells.

It should be pointed out that for the later snapshots at \( t = 0.145 \) s, \( t = 0.3 \) s and \( t = 0.745 \) s, anisotropic AMR did not get into the “deadlock” situation again with the new coarsening procedure in place, so that the dense mesh along \( x = 0 \) and \( y = 0 \) at Figure 4.9c is able to coarsen completely when the discontinuities move to a new location. The interference pattern produced by intersecting waves also become more dominant as time increases. Discontinuities in the interference region are not aligned with the \( x \) or \( y \) axis, in a similar manner as in the circular advection problem, where anisotropic AMR is observed to revert to isotropic AMR. Therefore, the computational saving from anisotropic AMR at \( t = 0.745 \) s is 47\%, much lower than that at \( t = 0.045 \) s, but still quite significant. A detailed accuracy analysis could not be performed due to the lack of an exact solution. However, by the observation that anisotropic AMR achieves similar spatial resolution as isotropic AMR in critical regions with most continuities and rapid solution changes, it can be concluded that the accuracy of anisotropic AMR and isotropic AMR are likely on par.

### 4.4 Supersonic Flow Past a Cylinder

The second simulation considered for the Euler equations involves supersonic flow past a bluff body, which causes a bow shock to form at a distance away from the bluff body surface. Directly in front of the bluff body, the supersonic free-stream flow becomes subsonic behind the shock and the flow comes to a full stop at the stagnation point. The
flow behind the shock elsewhere could be supersonic depending on the angle at which the free-stream flow crosses the bow shock. Due to the absence of viscous effects in the Euler equations, the flow velocity along the bluff body surface away from the stagnation point is non-zero. The bluff body simulation examined here involves free-stream flow at standard atmospheric pressure and density (i.e. $p = 101.325$ kPa and $\rho = 1.225$ kg/m$^3$), with a Mach number of 2. The bluff body is a 2D cylinder with radius $r = 1$, and with its axis of symmetry perpendicular to the free-stream flow. The simulation is carried out on a body-fitted mesh with the south boundary representing the surface of the bluff body and a curved north boundary shaped according to an estimated bluff body Mach number of 1.5. A fixed boundary condition was imposed at the north boundary for the free-stream inflow; a reflection boundary condition at the south and west boundaries for the subsonic regions; and a boundary condition with constant extrapolation at the east boundary for the outflow. A $k = 3$ CENO reconstruction is used for this calculation. High-order boundary treatment is reinforced by representing curved boundaries using...
(a) Density, Mach number and pressure distribution with mesh using isotropic AMR: \( N = 242800, \eta = 0.963 \).

(b) Density, Mach number and pressure distribution with mesh using anisotropic AMR: \( N = 74000, \eta = 0.989 \).
Figure 4.11: Comparison of isotropic and anisotropic AMR in simulating Mach 2 flow past circular cylinder with radius of 1m as a steady-state problem, with CENO $k = 3$ reconstruction.
splines, as well as by applying constraint reconstruction at the south boundary. The Rusanov flux evaluation is used for added dissipation.

The initial mesh consists of a single block with $20 \times 20$ cells, which undergoes 1 uniform refinement before the simulation begins. Grid generation during AMR for this problem uses the mid-point approach. An implicit method similar to that used in Section 4.1.2 is applied to obtain the steady-state results shown in Figure 4.11 together with the convergence history. The refinement and coarsening thresholds are 0.4 and 0.05 respectively, and maximum 8 levels of refinement are allowed. Solution distribution on the coarse mesh before any AMR is depicted in Figure 4.10, where the thickness of the bow shock on the coarse mesh can be up to a quarter of the blunt-body radius, and is without question too under-resolved. Fortunately, shape and location of the bow shock is much more accurately described with the assistance of AMR as shown in Figure 4.11. Although accuracy assessment is again not available due to the lack of an analytical solution, the distribution of the density, Mach number and pressure in the subsonic region as shown in Figure 4.11 for both AMR methods appear to agree with the coarse mesh solution plotted in Figure 4.10 and between isotropic and anisotropic AMR. Due to the setup of the initial mesh, the bow shock is not perfectly aligned with the grid lines and anisotropic AMR did not refine solely in one coordinate direction. Nevertheless, by optimizing the number of blocks along the bow shock, the application of anisotropic AMR has prevented coarser blocks in the more uniform regions of the domain from refining due to resolution conflicts. Hence anisotropic AMR still introduces a significant 70% reduction in the number of cells.

4.5 Ringleb Flow

The next numerical example considered for the Euler equations involves the simulation of Ringleb’s flow. Ringleb’s flow consists of isentropic and irrotational flow expanding between two streamlines, where analytical solutions are available. Therefore, this serves as an excellent test case where accuracy of the proposed high-order block-based anisotropic AMR scheme could be assessed and compared. Unlike the other two cases considered
for the Euler equations, Ringleb’s flow is much smoother in the solution distribution. Therefore, a transonic variant of Ringleb’s flow has been chosen for this study, where the supersonic region near the southwest corner of the domain is expected to require higher mesh resolution. The domain for this problem is defined by streamlines corresponding to $k_{\text{min}} = 0.5$ and $k_{\text{max}} = 1.2$, and subsonic inflow at the north boundary corresponding to $q = 0.3$. Fourth-order accurate space reconstruction is used, with exact solution imposed as boundary conditions on all boundaries. Grid generation makes use of the grid
metrics to ensure that the quality of the mesh does not affect accuracy of the solution. Lastly, refinement criteria for this problem involve gradients of the pressure field, such that variations in the supersonic region could be emphasized.

The results of the convergence study are depicted in Figure 4.12. All accuracy assessment studies begin with an initial mesh consisting of one $10 \times 10$ block. Solution accuracy with 2, 3, 4 and 5 levels of uniform refinements are again included as a reference, which show a fourth-order convergence for both density and pressure as anticipated. Accuracy assessments with the use of isotropic and anisotropic AMR are performed on the converged meshes with maximum refinement levels set to 2, 3, 4, 5 and 6 respectively. Steady-state solutions are obtained using a multi-stage method with optimal smoothing. Both AMR methods are observed to improve the convergence quite significantly up to 5 levels of refinement, with anisotropic AMR being more superior. Upon reaching 5 levels of refinements, AMR ceases to provide any advantage as the error becomes more distributed throughout the domain. Meshes resulting from both AMR methods with a maximum refinement level of 6 are shown in Figure 4.13.

4.6 Complex Shock Interaction

All of the previous problems considered for this thesis have used the high-order CENO reconstruction, which incorporates the second-order least-squares reconstruction and is hence a more comprehensive way to demonstrate the compatibility of anisotropic AMR with the finite-volume framework. However, in practice, physically complex flows which involve discontinuities do not benefit significantly from high-order methods due to the need to preserve monotonicity, which is where an effective AMR method becomes particularly valuable. Therefore, the last simulation involves complex shock interaction in a diamond-shaped inlet with supersonic inflow at Mach 4. The geometry of the inlet is shown in Figure 4.14. The top of the inlet begins with an initial rectangular section for $x < 0.6$, before the tunnel begins converging with an angle of 12 degrees. The bottom of the inlet resembles a diamond-shaped airfoil located $x = 1.5$. Leading edge of the diamond-shaped forms a 15 degrees wedge from the $x$-axis, and has a wedge length of
Chapter 4. Numerical Results

Figure 4.14: Coarse initial mesh and the steady-state solution for shock-interaction simulation in a diamond-shaped inlet.

0.5. The initial mesh consists of four 10 × 10 blocks, each representing a distinct section of the inlet geometry. Mesh stretching is turned on with a factor of 1.5, and is preserved through the use of metrics. The abrupt changes in geometry on the upper and lower boundaries of the inlet results in three oblique shocks and a Prandtl-Meyer expansion fan, which will reflect off the boundaries and form some complex shock-interaction patterns. Steady-state solution to this problem consists mainly of either discontinuities or uniform regions, which would render any higher-order schemes ineffective. In addition, it is difficult to obtain a numerical solution for this problem using prescribed meshes or shock-fitting methods. For these reasons, this is an excellent application to demonstrate the benefits of an effective AMR method, and a second-order accurate scheme will be used.

Gasdynamics relationships are available which allow shock angles and Mach numbers in each non-interference region to be determined. The angles of the oblique shocks formed, $\beta$, are related to the upstream Mach number, $M$, and the flow turning angle, $\alpha$, by: $\tan(\beta) = \frac{M \sin(\alpha)}{\cos(\alpha) - M \cos(\alpha)}$. The density distribution on the coarse mesh is shown in Figure 4.14(b).
Chapter 4. Numerical Results

(a) Isotropic AMR: $N = 585700$, $\eta = 0.91$.

(b) Anisotropic AMR: $N = 242800$, $\eta = 0.93$.

Figure 4.15: Density and Mach number distribution for the complex shock interaction simulation in a diamond-shaped inlet, with limited second-order least-squares reconstruction.

through the following expression [84]:

$$\cot \alpha = \tan \beta \left[ \frac{(\gamma + 1)M^2}{2(M^2 \sin^2 \beta - 1)} - 1 \right] \quad (4.1)$$

For $\gamma = 1.4$ for air, the first oblique shock deflected from the upper boundary of the inlet has a shock angle at approximately $-24$ degrees from the free-stream, while the shock formed at the leading edge of the diamond is at approximately $27$ degrees. Furthermore, the Mach number, $M_1$, in the region immediately after the leading edge shock is computed to be 2.93 according to [84]

$$M_1^2 \sin^2(\beta - \alpha) = \frac{(\gamma - 1)M^2 \sin^2 \beta + 2}{2\gamma M^2 \sin^2 \beta - (\gamma - 1)} \quad (4.2)$$
Chapter 4. Numerical Results

Figure 4.16: Comparison of convergence history with both isotropic and anisotropic AMR in simulating complex shock interaction simulation in a diamond-shaped inlet, with CENO \( k = 3 \) reconstruction.
Using, then, the Mach number relation for the Prandtl-Meyer expansion fan [85],

\[
2\alpha = \left| \sqrt{\gamma + 1} \tan^{-1} \sqrt{\frac{\gamma - 1}{\gamma + 1} (M^2 - 1)} - \frac{1}{\sqrt{\gamma - 1}} \tan^{-1} \sqrt{M^2 - 1} \right|_{M_1}^{M_2},
\]

the Mach number for the flow entering the trailing edge shock, \(M_2\) is found to be approximately 5.16. This in turn results in an oblique shock angle of approximately 24 degrees at the trailing edge of the diamond geometry.

Density distribution on the initial is also plotted in Figure 4.14, where the shocks and the interaction patterns are hardly identifiable. To capture all shocks with varying strengths, both AMR methods are applied with refinement and coarsening thresholds set to 0.15 and 0.02 respectively, as well as with a maximum refinement level of 8. An implicit method similar to that described in Section 4.1.2 is used to obtain the solution, with the addition of 100 explicit steps using a multi-stage method with optimal smoothing after each AMR and before the NKS iterations. Limiter-freezing is also turned on to assist in solution convergence. Results of the simulation have been plotted in Figure 4.15 for both isotropic and anisotropic AMR. The density and Mach number distributions in Figure 4.15a and Figure 4.15b appear similar, and it is evident that the numerical results agree well with the theoretical data for shock and rarefaction wave locations and strengths discussed above. In addition, all regions on the domain seem well resolved and clearly represented by both AMR methods. However, anisotropic AMR is able to almost double the efficiency of the method by introducing a 58% savings in the number of computational cells. Lastly, convergence history for this study has been included in Figure 4.16, where the residual is usually easily reduced by 8 orders of magnitude using implicit methods.

### 4.7 Remarks on Computational Savings

The proposed anisotropic AMR technique has been applied to numerous problems governed by the advection-diffusion equation and the Euler equations. Both unsteady and steady-state results have been obtained, where anisotropic AMR is compared with the
original isotropic AMR method. Overall, anisotropic AMR has been observed to provide similar spatial resolutions in regions of more rapid solution changes and discontinuities, using a significantly smaller number of cells, and that is in addition to the savings already introduced by the use of AMR.

If the number of solution blocks required in simulating a problem in a n-dimensional space is denoted as $N_n$, then for problems with solution varying only in one coordinate direction, same number of computational cells is required regardless of whether the problem is simulated in a 1D, 2D or 3D space (i.e., $N_3 = N_2 = N_1$) with anisotropic AMR. However, if isotropic AMR is used, $2^{level-1}$ blocks are added in 2D per block on the corresponding 1D domain, and $2^{2\times level-2}$ are added in 3D. Assuming that the maximum refinement level is high, corresponding savings by anisotropic AMR would be approximately $1 - 1/(2^{max level-1})$ in 2D and $1 - 1/(2^{2\times max level-2})$ in 3D. For problems involving a single planar shock, a spherical shock in spherical coordinates or a thin boundary layer involving more than 6 levels of refinements, the savings could easily be over 95%. However, problems with pure uni-directional variations are not so commonly encountered in physically complex flows. Take the shock-box and supersonic flow past cylinder simulations for example, although computational savings are significant, neither are over 90%.

In the majority of the problems considered for this thesis, anisotropic AMR often reverts to isotropic AMR in some regions because solution variation was not strongly aligned with either of the coordinate directions. Savings in computational cells for these problems have been approximately 50%, which is from optimizing the number of blocks in the more uniform regions of the domain. In 3D applications where the solution distribution is somewhat isotropic on a planar level but non-variant in the third dimension, savings with anisotropic AMR could again be approximated by $1 - N_{2,aniso}/(N_{2,iso}(2^{max level-1}))$. Using this approximation, 3D axisymmetric extensions of supersonic flow past a cylinder and the flow inside a diamond-shaped inlet will have projected savings of over 99%. For 3D problems, where the solution variations are rapid but relatively isotropic in all coordinate directions, a rigorous prediction of the savings becomes more difficult. Nevertheless, judging from the exponential nature of the number
of used cells when another dimension is added, it is estimated that the saving would probably be around 75% or more.
Chapter 5

Conclusions and Future Research

This thesis began by identifying a few physically complex flows, which are difficult to simulate by numerical methods in CFD. To overcome these challenges, some fundamental issues need to be addressed, which include high-order discretization, mesh adaptation techniques, parallel implicit methods and implementations which could take advantage of the advanced parallel computing platforms available today. Adaptive mesh refinement technique has offers an intelligent and robust way to generate high-quality solution grids with minimal supervision. Computational grids that automatically adapt to the solution are very effective in treating problems with disparate length scales, providing the required spatial resolution while minimizing memory and storage requirements. Furthermore, the use of AMR in conjunction with finite-volume schemes has produced some very powerful methods for the treatment of a wide variety of physically-complex flows with complex and/or moving geometries. However, fully reliable and accurate AMR strategies, particularly for high-order schemes and viscous flows, as well as efficient parallel implementations for unsteady flows with dynamic adaptation, have yet to be devised.

Therefore, this thesis has proposed a block-based anisotropic AMR technique for body-fitted multi-block meshes. With the use of a hierarchical binary tree data-structure, the proposed technique readily permit local refinements of the grid blocks in each of the coordinate directions in an independent fashion. Anisotropic AMR aims to improve upon the efficiency and robustness of the traditional isotropic AMR methods, by allowing for the more efficient and accurate treatment of narrow layers, discontinuities, and/or shocks...
in the solutions which occur, for example, in the thin boundary and mixing layers of high-Reynolds-number viscous flows and in the regions of strong non-linear wave interactions of high-speed compressible flows with shocks. The described anisotropic AMR technique has been designed to work with an existing finite-volume framework with parallel second-order and higher-order, explicit and implicit capabilities, to demonstrate its potentials for future extensions to simulating more complex and challenging flow applications.

5.1 Summary of Contributions

In implementing the proposed anisotropic AMR technique for two-dimensional applications, a binary tree data structure has been put in place, with purposes and functions very similar to those of a quadtree data structure. However, the binary tree data structure has the added capability of dynamically rearranging the internal connectivity of the nodes to enable coarsening of non-sibling blocks, without altering the existing mesh. To further alleviate the constraint imposed by the sequence of refinements and coarsening in each coordinate direction, anisotropic AMR procedures for both coordinate directions have been merged into one step while still permitting directional refinements and coarsening. Furthermore, a flexible and robust conflict-checking algorithm has been implemented to accommodate the new anisotropic AMR procedure, and to simplify the extension of the algorithm into three dimensions. Lastly, additional message passing procedures which arise as the results of anisotropic AMR have been considered and implemented.

The current implementation of anisotropic AMR has also been extensively tested through applications in first the advection-diffusion equation, and subsequently the Euler equations, where anisotropic AMR has demonstrated great potentials when compared to the performance of the original isotropic AMR. As expected, anisotropic AMR is most advantageous in simulating flows with rapid solution changes, particularly when the solution gradients are aligned with coordinate axis, although savings are still substantial when anisotropic AMR partially reverts to isotropic AMR in cases where the discontinuities present are not aligned with the coordinate axis. Anisotropic AMR has also been observed to better conform to local variations even for more uniform solution distribu-
tions. Accuracy assessments have been performed on a few simple simulations, although the results appear less satisfactory possibly due to the simplicity of the problems considered (i.e., lack of disparity in the spatial scales) and the lack of sophisticated refinement criteria.

5.2 Future Considerations

There are some technical issues which remain unaddressed in the current implementations of anisotropic AMR due to time limitations. First of all, procedures for message passing need to be reconsidered for accuracy reasons previously discussed in Section 3.5.2. Secondly, interior cell solution transfer upon refinement is done via direct injection, which reduces the accuracy of high-order simulations with AMR. This is related to possible consecutive refinement of a block in each of the coordinate directions. As the solution reconstruction is not updated after each refinement, high-order solution prolongation becomes unavailable. One way to fix this problem is to permit refinements of solution blocks using the isotropic refinement routines. This could possibly be taken a step further to ensure that when anisotropic AMR reverts to isotropic refinement or coarsening, the resulting grids and solutions are consistent with those that would result from application of isotropic AMR.

Physically complex flows in real practice are mostly three-dimensional, where the benefits of anisotropic AMR are expected to increase exponentially. Therefore, implementation of the proposed technique in three dimensions is among the more immediate extensions of this thesis. Unlike a quadtree data structure which requires modifications into an octree, a binary tree data structure is directly applicable in higher dimensions. Although additional complexities will be introduced in operations like connectivity rearrangement, neighbor searching and conflict checking, these procedures have been generalized to hopefully simplify the modifications required. Lastly, an effective AMR technique cannot be without refinement indicators that accurately identify under-resolved regions. Error- and adjoint-based refinement criteria based on $p$- or $h$-refinements have been proposed in the past [79, 80, 81], but they are not yet fully matured, and anisotropic variants
of these criteria remain to be fully formulated. Furthermore, refinement criteria which could possibly facilitate the switching between high-order methods ($p$-refinement) and AMR ($h$-refinements) are worthy of further investigations. It is believed that a sophisticated algorithm to compute effective refinement criteria will allow anisotropic AMR to become a much more powerful method.
Bibliography


