A Parallel Solution-Adaptive Method for Turbulent Non-Premixed Combusting Flows

by

Xinfeng Gao

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy
Graduate Department of Aerospace Science and Engineering
University of Toronto

Abstract

A Parallel Solution-Adaptive Method for Turbulent Non-Premixed Combusting Flows

Xinfeng Gao

Doctor of Philosophy

Graduate Department of Aerospace Science and Engineering

University of Toronto

2008

A parallel adaptive mesh refinement (AMR) algorithm is proposed and applied to the prediction of steady turbulent non-premixed compressible combusting flows in both two and three space dimensions. The parallel solution-adaptive algorithm solves the system of partial-differential equations governing turbulent compressible flows of reactive thermally perfect gaseous mixtures using a fully coupled finite-volume formulation on body-fitted multi-block quadrilateral and hexahedral meshes. The compressible formulation adopted herein can readily accommodate large density variations and thermo-acoustic phenomena. A preconditioned multigrid algorithm is used to obtain the solution on highly stretched meshes in a more efficient manner. A flexible block-based hierarchical data structure is used to maintain the connectivity of the solution blocks in the multi-block mesh and to facilitate automatic solution-directed mesh adaptation according to physics-based refinement criteria. For calculations of near-wall turbulence, an automatic near-wall treatment readily accommodates situations during adaptive mesh refinement where the mesh resolution may not be sufficient for directly calculating near-wall turbulence using the low-Reynolds-number formulation. Numerical results for turbulent diffusion flames, including cold- and hot-flow predictions for a bluff-body burner, are described and compared to available experimental data. The numerical results demonstrate the validity and potential of the parallel AMR approach for predicting fine-scale features of complex turbulent non-premixed flames.

Acknowledgements

This thesis research has provided me with a rich learning experience full of much excitement and joy. I would like to thank many people who have contributed to this great experience.

I would like to express my gratitude to my supervisor Prof. Clinton P. T. Groth for his excellent guidance and teaching. I've been encouraged and influenced greatly by his enthusiasm and attitude toward research. I would like to thank him for many stimulating suggestions and encouragement that helped me during the time of research and the thesis write-up.

I would like to thank my thesis committee Prof. David W. Zingg and Prof. Ömer L. Gülder for their advice and recommendations.

I am deeply indebted to my husband and my parents for their continuing support, understanding and encouragement to this thesis research over the years, and their patient love enabled me to complete this work. I would like to thank my friend and colleague, Stephen Guzik, for his continuing help, meticulous suggestions and inspirations. I am grateful to him for many enjoyable moments and insightful discussions.

I thank Dr. Jai Sachdev, Dr. Peterson Wong, and Scott Northrup for many valuable discussions and help during my research. I thank Alistair Wood for his early revision on the thesis draft. I thank all my friends and my colleagues from UTIAS.

I am very grateful to Prof. Clinton P. T. Groth and University of Toronto for the financial support provided throughout this research.

Xinfeng Gao

University of Toronto
July 21, 2008

Contents

List of Tables

List of Figures

List of Symbols

Alphanumeric

$A_{i,j}$	area of cell i, j
$\vec{u}$	Favre-averaged mixture velocity
c_{p_n}	species specific heat
c_n	species mass fraction
D_b	bluff-body burner diameter
D_k	diffusion coefficient for turbulent energy
D_{tn}	turbulent diffusivity of species n
e	Favre-averaged total specific mixture energy
E_p	relative parallel efficiency
$\mathbf{F}$	inviscid radial flux vector in two-dimensional (axisymmetric) coordinate system or inviscid x flux vector in three-dimensional coordinate system
$\mathbf{F}_v$	viscous radial flux vector in two-dimensional (axisymmetric) coordinate system, or viscous x flux vector in three-dimensional coordinate system
$\mathbf{G}$	inviscid axial flux vector in two-dimensional (axisymmetric) coordinate system or inviscid y flux vector in three-dimensional coordinate system
$\mathbf{G}_v$	viscous axial flux vector in two-dimensional (axisymmetric) coordinate system or viscous y flux vector in three-dimensional coordinate system
$\mathbf{H}$	inviscid z flux vector in three-dimensional coordinate system
$\mathbf{H}_v$	viscous z flux vector in three-dimensional coordinate system
h_n	absolute internal enthalpy for species n
$\vec{\mathcal{J}}_n$	species molecular diffusive flux
$\vec{\mathcal{J}}_{tn}$	species turbulent diffusive flux
k	specific turbulent kinetic energy
Ma	Mach number
N	total number of species

Alphanumeric

p	time-averaged mixture pressure
Pr_t	turbulent Prandtl number
$\vec{n}$	unit vector normal to the cell face or edge
$\vec{q}$	molecular heat flux vector
$\vec{q}_t$	turbulent heat flux vector
Re	Reynolds number
Sc_t	turbulent Schmidt number
$\mathbf{S}_\mathrm{c}$	chemical source vector
S_{ki}	strain rate tensor
S_p	relative parallel speed-up
$\mathbf{S}_\mathrm{t}$	turbulent source vector
$\mathbf{S}_\mathrm{a}$	inviscid source vector in two-dimensional (axisymmetric) coordinate system
$\mathbf{S}_{\mathrm{a}v}$	viscous source vector in two-dimensional (axisymmetric) coordinate system
u_τ	friction velocity
y	the distance normal from the wall
y^+	dimensionless, normal distance from wall

Greek

$\alpha \quad \beta \quad \beta^* \quad \sigma \quad \sigma^*$	closure coefficients of k-ω two-equation model
$\vec{\vec{\lambda}}$	turbulent Reynolds stress tensor or dyad
μ	mixture molecular viscosity
μ_n	species molecular viscosity
μ_t	eddy viscosity
ρ	time-averaged mixture density
τ_w	wall shear stress
κ	thermal conductivity
κ_n	species thermal conductivity
κ_t	turbulent thermal conductivity
$\vec{\vec{\tau}}$	molecular stress tensor or dyad
ω	specific dissipation rate
Ω_{ij}	vorticity tensor
$\dot{w}_n$	species mean reaction rate

Chapter 1

Introduction

1.1 Turbulent Reactive Flow Simulations

With the recent advances in computational fluid dynamics (CFD) and numerical methods for combusting flows, as well as advances in high-performance-computing hardware, numerical modeling has become an important, powerful, and effective tool for the design of advanced combustion systems. The reliance on numerical modeling has also increased with the increasingly stringent emission legislation imposed by governments worldwide [2], as the latter has made the combustor and engine design process much more challenging.

Virtually all practical combustion systems involve turbulent combustion. Moreover, pollutant and particulate emissions are controlled by the details of the fuel-air mixing and combustion processes. For these reasons, a detailed understanding of the strong nonlinear interaction between the turbulent flow structure, chemical kinetics, and thermodynamic properties of the reactants and products is required to obtain improved low-emission combustor designs. Note that there are a wide range of existing combustion configurations in which the fuel and oxidizer are initially separated and this provides some of the rationale for emphasizing non-premixed combustion process in the present study.

Three primary tools for performing simulations of turbulent combusting flows have emerged: (i) direct numerical simulation (DNS); (ii) large-eddy simulation (LES); (iii)

and Reynolds- or Favre-averaged Navier-Stokes (RANS) simulation techniques, each possessing various advantages and disadvantages [3, 4]. In DNS, all of the turbulent and chemical length and time scales are fully resolved. For this reason, DNS is a powerful tool for studying turbulent flame structure and turbulence/chemistry interactions in detail. It is well suited for providing an understanding of the basic processes of combustion phenomena, such as extinction and re-ignition, flow and flame unsteadiness, and differential diffusion of chemical species. Some recent examples of the application of DNS techniques to problems in combustion are described by Vervisch [5]. However, despite the successes to date, DNS is generally restricted to generic simplified and/or more academic combustor configurations due to the very high computational costs of fully resolving all solution scales, both turbulent and chemical. It will probably not be used to simulate turbulent combustion phenomena in practical combustor configurations with complex geometry any time in the near future.

LES is an alternative to DNS in which the large energy containing structures or eddies are computed directly and the small, generally more universal, dissipative, turbulent scales are modeled, thereby offering potential computational savings [6–9]. Over the last decade, the approach has evolved to become a truly predictive tool for non-reacting flows [6–8, 10, 11] and has been shown in many applications to provide more accurate predictions of the flow fields than the more conventional RANS-based methods for reacting flows [5]. Nevertheless, universal and accurate sub-filter scale models for non-premixed and premixed reacting flows are not currently available and the accurate and reliable numerical solution of the filtered Navier-Stokes equations remains a significant computational challenge for many practical problems.

As LES is still at an early stage of development for combusting flows and due to the still relatively high cost of performing such simulations, RANS-based methods are the predominant approach in engineering CFD applications for combusting flows involving complex flow geometries [12]. Moreover, this situation is not expected to change in the near future. Nevertheless, in spite of simplifications offered by time-averaging approaches (integral length and time scales for the turbulence need only be resolved, rather than all scales down to those of the Kolmogorov and chemical

scales), the system of time-averaged equations governing turbulent combusting flows can be both large and stiff and its solution can still place severe demands on available computational resources. In particular, approaches are required to reduce the computational costs of simulating combusting flows using RANS-based methods, thereby permitting their application on a more routine basis to a wider range of problems.

1.2 Parallel Adaptive Algorithms

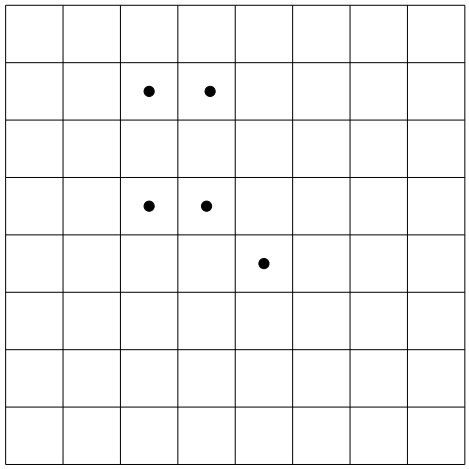
Many approaches have been taken to reduce the computational time required for simulating combusting flows using RANS techniques. One successful approach is to make use of solution-directed mesh adaptation, such as the adaptive mesh refinement (AMR) algorithms. Efficient and accurate simulations of turbulent combusting flows require a careful and selective gridding of the flow field; however, three-dimensional mesh generation for complex geometry currently can require a significant amount of manpower and computational effort. At the present time, a general rule of thumb is that approximately 50% of the time to obtain a CFD flow solution is associated with mesh generation. Given an initially coarse mesh that hopefully can be generated in a relatively short period of time, AMR algorithms can then produce more highly refined mesh with many desirable features while significantly lowering the manpower requirements and computer costs usually associated with the mesh generation and subsequent solution computation. Large massively-parallel distributed-memory computers provide another approach by enabling a many fold increase in processing power and memory resources beyond those of conventional single-processor computers. These parallel computers provide an obvious avenue for greatly reducing the time required to obtain numerical solutions of combusting flows. A combination of these two strategies to produce a parallel AMR method that both reduces the overall problem size and the corresponding time to calculate a solution would seem very desirable.

1.2.1 Overview of Adaptive Mesh Refinement Algorithms

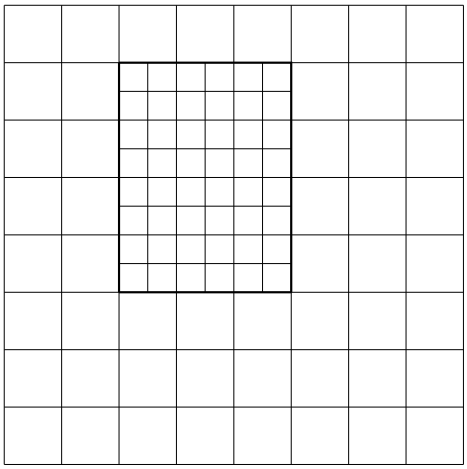
AMR is a powerful tool for computing solutions to partial differential equations (PDEs) whose solutions have disparate spatial scales. Computational grids are automatically adapted to the solution of the governing equations and this can be very effective in treating problems with multiple scales, providing the required spatial resolution while minimizing memory and storage requirements. An adaptive grid technique was originally proposed by Berger and Oliger for computing time-dependent solutions to hyperbolic PDEs in multiple space dimensions [13], and AMR approaches have since been developed for a wide variety of engineering problems [14–32]. Currently, several different AMR strategies have emerged. These approaches can be classified into four broad categories depending on the partitioning algorithm used and/or the data structure that is adopted to keep track of the mesh connectivity. They are as follows: (1) "patch-based", (2) "cell-based", (3) "block-based", and (4) "hybrid block-based" AMR techniques. Figure 1.1(a) depicts a base Cartesian mesh with cells tagged for refinement. Figures 1.1(b)–1.1(d) demonstrate the subsequent refinement of this base mesh resulting from the "patch-based", "cell-based", and "block-based" AMR methods. Each of these strategies is now briefly reviewed in turn below and compared to one another. The advantages and disadvantages of each strategy are also discussed.

Berger, Oliger and Colella [13, 14] developed an algorithm for dynamic gridding, now more generally referred to as patch-based AMR. The algorithm begins with the entire computational domain covered with a coarsely resolved base-level regular Cartesian grid. As the calculation progresses, individual grid cells are tagged for refinement as illustrated in Figure 1.1(b). The patch-based AMR strategy relies on a fairly sophisticated algorithm, laid out by Berger [33], to organize a collection of individual grid cells into properly nested rectangular patches. The mesh within these newly farmed patches can then be further refined, creating additional patches. State-of-the-art packages adhering to this patch-based AMR include Chombo [34] and SAMRAI [35].

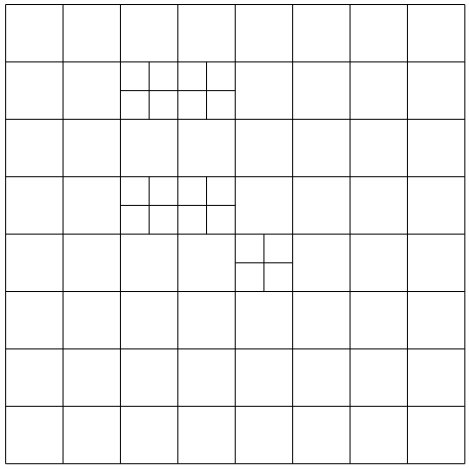
In cell-based AMR, as proposed and developed for example by Powell and coworkers [17, 18, 36] and Berger and Aftomis [24, 37, 38], each cell can be refined indi-



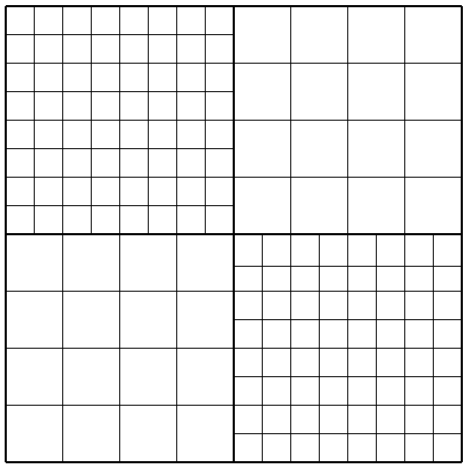
(a) Base Cartesian grid with cells tagged for refinement as indicated by black dots



(b) Refined Cartesian mesh resulting from patch-based AMR



(c) Refined Cartesian mesh resulting from cell-based AMR



(d) Refined Cartesian mesh resulting from block-based AMR

Figure 1.1: Illustration and comparison of the patch-based, cell-based and block-based AMR techniques for Cartesian mesh.

vidually as shown in Figure 1.1(c) and each cell is stored using a tree data structure. This cell-based tree structure is flexible and readily allows for the local refinement of the mesh by keeping track of the computational cell connectivity as new grid points are generated from the refinement process. Virtually all cell-based approaches are based on Cartesian meshes. In many cell-based approaches, cut cells are generally used

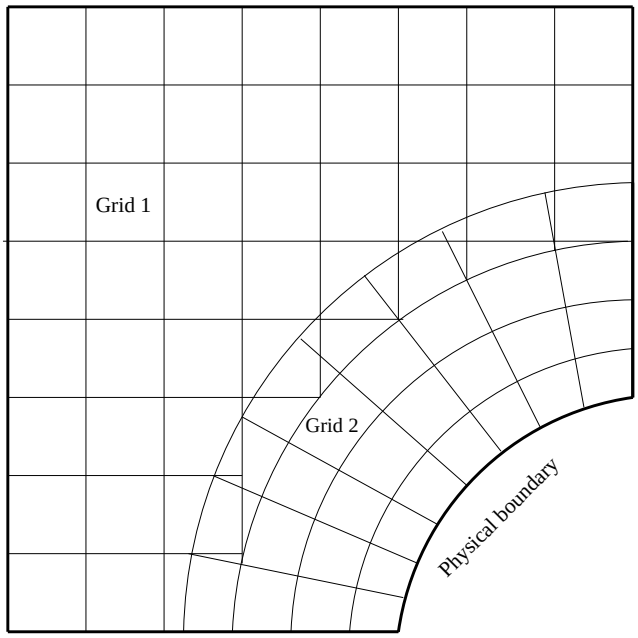
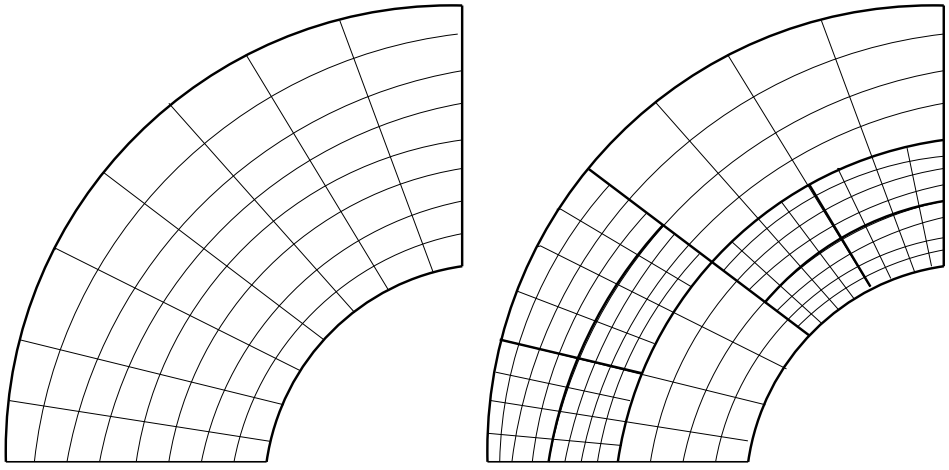


Figure 1.2: Illustration of overlapping grid consisting of two structured curvilinear component grids.



(a) A body-fitted coarse grid

(b) A refined body-fitted grid consisting of ten grid blocks

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

to treat complex geometry and very efficient AMR schemes have been devised using this boundary treatment. However, discretization of elliptic operators on Cartesian cut cells can be challenging [39] and applications are generally restricted to hyperbolic systems. Cart3D is a good example of a high-fidelity inviscid analysis package which allows for automated CFD analysis on complex geometry with the cut-cell technique to handle the curved geometry boundaries [40]. Fully three-dimensional meshes around extremely complex objects can be generated automatically and routinely in a matter of a few hours or less using this technique.

Another AMR approach for treating more complex geometries with curved boundaries is based on composite overlapping grids used together with AMR. In this case, curvilinear grids that conform to the curved boundaries are used together in an overlapping fashion with one or more Cartesian grids that fill the interior of the domain. Figure 1.2 illustrates an overlapping grid consisting of two structured curvilinear component grids in physical space, an annular grid and a background Cartesian grid. In essence, a Chimera overlapping grid [41] technique is combined with AMR. Boden and Toro [42], Brislawn [43], Chesshire [44], and Henshaw [45–49] demonstrated that AMR on overlapping grids can lead to an efficient approach for solving problems with multiple space and time scales for complex geometry. A main challenge of this AMR approach is to determine the physical grid point in terms of mapping when refining body-fitted grids. In other words, each component grid is logically rectangular defined by a smooth mapping from computational space to physical space. The mapping is used to define grid points at any desired resolution as required when a grid is refined. Re-gridding for each base grid is necessary during the adaptation process and the grid information such as connectivity and those grids hidden by refined grids, has to be re-generated and stored. In general, some care is required during the re-gridding for the interpolation between different base grids and/or between grids with different levels to ensure that accurate values are obtained. Global conservation properties are also difficult or impossible to enforce discretely with this overlapping grid approach.

In a block-based AMR strategy, mesh adaptation is accomplished by the dividing and coarsening of appropriate solution blocks. In general, each block also has an equal number of cells shown in Figure 1.1(d). The basic data structure is then a tree

(quadtree for two dimensions and octree for three dimensions), where any block that requires refinement generates a number of equal sized blocks (4 in two dimensions and 8 in three dimensions) when a resolution change of two is assumed. The block-based AMR strategy results in a rather light tree data structure for prescribing the connectivity between blocks as compared to the tree structure generally used for tracking cell connectivity in the cell-based methods. In addition, the block-based data structure naturally lends itself towards an efficient and readily scalable parallel implementation. It amortizes the overhead of communication over entire blocks of cells, instead of over single cells as in cell-based data structures. However, generally larger numbers of refined cells can be created (i.e., typically more than the corresponding number of cells used in cell-based tree data structures) thereby possibly increasing the amount of computational work and storage space needed to solve a given problem.

Applications of the block-based approach on Cartesian mesh are described by Quirk [19], Berger [20], Gombosi and co-workers [50–54]. Groth and co-workers [32,55] have since extended the approach developed by Groth *et al.* for computational magnetohydrodynamics [30,31,56] and developed a flexible block-based hierarchical data structure to facilitate automatic solution-directed mesh adaptation on multiblock body-fitted (curvilinear) meshes for complex flow geometries. While introducing some added complications, the use of body-fitted meshes permits more accurate solutions near boundaries, enables the use of anisotropic grids with grid point clustering and stretching, and allows for better resolution of thin boundary and mixing layers. Furthermore, unlike the overlapping AMR approaches, conservation properties of the solution scheme are readily enforced discretely. Figure 1.3 illustrates the application of block-based AMR technique to a body-fitted mesh.

Finally, hybrid block-based AMR approaches have also been considered. Holst and Keppens [57] applied a hybrid approach to general curvilinear coordinate systems, modifying the full tree data structure to allow for incomplete block families (not all children are created; the usual block-based AMR always has complete families of 2, 4, or 8 children depending on dimensionality) and incorporating the ideas of patch-based strategies. This hybrid AMR strategy requires two means to traverse the grid hierarchy, e.g., there is a doubly linked list of grid pointers per level in addition

to the tree data structure. Thus, the mixed data structure further complicates the neighbour search algorithm in three-dimensions. Holst and Keppens [57] compared the three AMR strategies, i.e., a patch-based, a tree block-based, and a hybrid block-based, for a smooth two-dimensional advection test problem on a doubly periodic domain with a second order numerical scheme, and found that the block-based AMR approach is the most efficient in terms of the execution speed for the same accuracy. However, it should be kept in mind that the applications considered by Holst and Keppens were mainly two-dimensional and were restricted to classical and relativistic MHD simulations.

In this thesis research, a block-based AMR strategy will be developed for combusting flow applications, as this strategy appears to be somewhat more computationally efficient with respect to parallelization aspects and memory requirements than cell-based AMR. In addition, it allows for the use of anisotropic body-fitted grids and is therefore better suited to tackling problems with thin shear and boundary layers and curved boundaries. Further, although not considered here, this approach is also well suited for solving large systems of PDEs, such as those encountered in turbulent combusting flows with a preconditioned Krylov subspace iterative scheme as outlined by Groth and Northrup [55]. See also related work by Keyes and co-workers [58–61]. The Schwarz type preconditioning used in the Newton-Krylov method exploits the block structure of the grid to produce a very efficient parallel implementation of a fully implicit time marching scheme.

1.2.2 Parallel AMR Algorithms for Combusting Flows

Adaptive mesh refinement techniques have been applied previously to both steady [62–66] and unsteady [26, 67] combustion simulations. Bennett *et al.* included local rectangular refinement techniques with a finite-difference discretization for realistic combustion modeling [68,69]. The development of a new adaptive meshing method and its combustion applications were also explored by Bennett and Valdati [68–71]. In addition, the effects of adaptive mesh refinement have also been explored on unstructured meshes. Silva *et al.* [72] demonstrated an adaptive

unstructured-grid calculation of high-speed, compressible flows of inert and reactive gas mixtures. Sullivan *et al.* [73] investigated NO_x formation in laminar, ammonia-seeded, nitrogen-diluted, methane diffusion flame with the use of adaptive mesh refinement to capture fine-scale features of the flame. The latter included a detailed chemical mechanism, differential diffusion, buoyancy, and radiative losses.

The use of massively parallel computing [74,75] has greatly advanced physics-based numerical simulation of aircraft engine and combustion flows. Douglas *et al.* [76] described a parallel algorithm for numerical combustion modeling and carried out simulations of both the flame sheet problem (an axisymmetric laminar diffusion flame) [77] and combustion problems with a finite-rate chemistry [77, 78] on parallel computers. Desprez *et al.* [79] present the parallel computation of a combustion model with a two-step sequential reaction mechanism and the algorithm was implemented with high efficiency on the hypercube iPSC/860 and the Paragon by overlapping communications and computations. Nkonga *et al.* [80] developed a parallel strategy based on a dynamic communication structure for the computation of a dispersed spray in a turbulent flow. Yasar [81] has considered simulations of fusion plasmas, internal combustion engines, particle dynamics, and transport systems, and has also analyzed the performance of this scalable parallel algorithm for numerical simulations of turbulent, radiative, magnetized, reactive fluid and particle systems on parallel distributed-memory computers. Some of Yasar's work involves the study of internal combustion engines and further has been utilized to produce numerical models such as the parallel version of KIVA-3, a block-structured, multidimensional finite-difference combustion code that is applicable to laminar and turbulent flows, subsonic and supersonic flows, and single-phase and dispersed two-phase flows [81–84]. Oran *et al.* [85] conducted two-dimensional computations of the propagation of a detonation in a low-pressure, argon-diluted mixture of hydrogen and oxygen with a detailed chemical reaction mechanism on massively parallel computers. Lepper *et al.* [86] described the parallelization strategy and some details of the implementation of a three-dimensional simulation code for turbulent flow combustion processes in full-scale utility boilers. Huang *et al.* [87] investigated the flame dynamics in a lean-premixed swirl-stabilized combustor. Chen and co-workers have developed a massively parallel DNS code for

simulations of turbulent flames with detailed descriptions of chemistry [88]. Although the DNS simulations are limited to small scale laboratory flames with simple flow geometries, the results can be used to validate mixing and combustion models used by simulations for practical combustion devices and some fundamental insight into complex turbulence/chemistry interactions in flames.

Combined parallel AMR algorithms have also been considered in previous studies. Recent progress in the development and application of parallel AMR algorithms for low-Mach-number reacting flows and premixed turbulent combustion is described by Day and Bell [89–93]. More recently, Northrup and Groth [94] and Gao and Groth [95, 96] have also proposed a parallel block-based AMR method using body-fitted multiblock meshes for application to both laminar and turbulent non-premixed combusting flows. The success of the block-based approach for body-fitted multiblock meshes prompted Gao and Groth [97] to consider the extension of the parallel algorithm for combusting flows to three dimensions. This thesis encompasses this extension and provides a detailed description of the parallel adaptive algorithm for two-dimensional axisymmetric and three-dimensional turbulent non-premixed combusting flows.

1.3 Motivation and Thesis Objectives

As described above, the prediction of turbulent combustion processes by numerical methods remains a very challenging area of active research, due to the fact that turbulent combusting flows involve a wide range of complicated physical and chemical phenomena (flame front behavior is dictated by a strong interaction between the turbulent flow structure, chemical kinetics, and thermodynamic properties of the reactants and products). In addition, the numerical solution of such flows places heavy demands on currently available computing resources.

The thesis research therefore focuses on the development of a new parallel CFD method for more efficiently predicting both two- and three-dimensional turbulent combusting flows. The goal is to devise a robust and efficient computational tool that harnesses the potential of high-end parallel computers and thereby enables the more

routine prediction of combustion processes associated with combustors of gas turbine engines. The resulting numerical tool should enable detailed analysis of turbulent combusting flows and at the same time be very helpful in the design and development of gas turbine combustor systems. The research objectives are as follows:

1. Develop a numerical framework for predicting multi-species turbulent reacting flows for gaseous fuels and oxidizers based on a finite-volume formulation applied to the compressible form of the governing equations.

2. Cope with numerical stiffness associated with disparate spatial scales by developing a parallel solution-adaptive algorithm.

3. Verify the proposed parallel adaptive algorithm for a range of turbulent non-reactive and reactive flows.

The primary goal of this thesis research is to enable numerical solutions of non-premixed combusting flows in a routine, efficient, and accurate manner and not to improve modelling of such flows. Therefore, somewhat simplified models for turbulence, chemical kinetics, and interactions between turbulence and chemistry were employed. In particular, a two-equation, k-ω turbulence model, one-step chemical kinetics for gaseous fuels, and an eddy-dissipation model of turbulence/chemistry interactions are considered and used. The use of these somewhat standard yet simple models of turbulence, chemical kinetics, and turbulence-chemistry interaction has reduced the mathematical complexity of the problem and thereby allows one to concentrate on aspects of the algorithm development. Nevertheless, the modelling incorporates key physical and chemical factors controlling combustion processes in realistic combustors with sufficient accuracy for the predictions to be of engineering value. In some sense, this research provides a foundation for future follow-on work that will involve the use of more detailed chemical kinetic schemes, more sophisticated turbulence chemistry interaction models, soot and radiation models, and multi-phase flow treatment for liquid fuels.

It should be noted that the focus of this thesis research is on the development of a highly scalable parallel finite-volume scheme with AMR for treating numerically

stiff combusting flows having potentially disparate spatial scales. Although a multigrid method is considered as part of this study for accelerating the convergence for steady-state problems, appropriate time-marching and/or iterative methods for solving unsteady and steady reactive flow problems having disparate temporal scales is largely outside the scope of this thesis. It will undoubtedly be the subject of future research efforts.

1.4 Thesis Organization

The thesis is structured as follows. In Chapter 2, the system of governing equations for a compressible thermally perfect reactive mixture of gases is presented. In Chapter 3, the main elements of the finite-volume scheme and the multigrid algorithm are described. Chapter 4 presents details on the proposed parallel adaptive mesh refinement scheme developed herein. A partial numerical verification of the algorithm is carried out in Chapter 5. Components of the proposed algorithm are evaluated separately for several canonical flow problems. In Chapter 6, the solutions of both two-dimensional (axisymmetric) and three-dimensional algorithms for a bluff-body burner are compared to experimental results. Finally, some conclusions are drawn and the main contributions of the thesis are highlighted in Chapter 7.

Chapter 2

Mathematical Modelling

This chapter summarizes the governing equations and mathematical modelling used herein for describing turbulent combusting flows. Section 2.1 introduces the concept of Favre averaging and provides details of the derived Favre-averaged Navier-Stokes equations for a turbulent reactive gaseous mixture. The closure approximations for the unresolved terms resulting from the Favre-averaging process are based on the Boussinesq hypothesis and details are described in Section 2.2. Section 2.3 outlines the thermodynamic relationships and transport coefficients used to close the system of governing equations. Finally, the closures for the time-averaged chemical source terms are reviewed and described in Section 2.4.

2.1 Favre-Averaged Navier-Stokes Equations

2.1.1 Time Averaging Procedure

Many flows of engineering significance are turbulent. Turbulence introduces random-like fluctuations to the flow properties. Most practical theoretical methods and computational tools capable of representing the effects of turbulence are based on statistical approaches, namely averaging concepts. The associated theory provides a means of predicting or determining the averages of solution quantities. Reynolds averaging of solution quantities, introduced by Reynolds in 1895, assumes a variety of

forms involving either an integral or a summation [98]. The three forms of averaging are the time average, spatial average and ensemble average. For the sake of completeness, these forms are now briefly reviewed. The reader is referred to the textbook by Wilcox [98] for further details.

Time averaging is the most commonly used form of averaging and is appropriate for stationary turbulence (i.e., a turbulent flow that does not vary statistically with time). Consider an instantaneous flow variable, such as instantaneous velocity, $\vec{u}$, expressed as $\vec{u}(\vec{x},t)$. The time average of $\vec{u}$ is defined by

$$\overline{\vec{u}_{\mathrm{T}}}(\vec{x}) = \lim_{\mathrm{T}\longrightarrow\infty} \frac{1}{\mathrm{T}} \int_{\mathrm{t}}^{\mathrm{t+T}} \vec{u}(\vec{x},t)dt\,, \tag{2.1}$$

where $\overline{\vec{u}_{\mathrm{T}}}(\vec{x})$ is the mean value of velocity and T is the time interval for the averaging procedure. Note that, although applied to velocity here, the averaging procedure can be applied to any flow variable of interest. An infinite value for T is not realizable in any physical flow; and a practical finite selection of T is sufficient if it is long relative to the maximum period of the turbulent velocity fluctuations [98].

Spatial averaging, which involves an average over the fluid volume and requires integration over all spatial coordinates, is appropriate for spatially homogeneous turbulence. For a turbulent flow that, on the average, is uniform in all directions, spatial averaging can be expressed as

$$\overline{\vec{u}_{\mathrm{V}}}(t) = \lim_{\mathrm{V}\longrightarrow\infty} \frac{1}{V} \iiint_{\mathrm{V}} \vec{u}(\vec{x},t)dV\,, \tag{2.2}$$

where $\overline{\vec{u}_{\mathrm{V}}}(t)$ is again the mean value.

The most general type of Reynolds averaging suitable for turbulent flows that vary in both space and time is ensemble averaging. As an illustration, the ensemble-averaged velocity, $\overline{\vec{u}_{\mathrm{E}}}(\vec{x},t)$, is defined by

$$\overline{\vec{u}_{\mathrm{E}}}(\vec{x},t) = \lim_{\mathrm{N}\longrightarrow\infty} \frac{1}{N} \sum_{\mathrm{n}=1}^{\mathrm{N}} \vec{u}_{\mathrm{n}}(\vec{x},t)\,, \tag{2.3}$$

where N represents the number of identical experiments and $\overline{\vec{u}_{\mathrm{E}}}(\vec{x},t)$ is the ensemble average velocity. Clearly, for both stationary and homogeneous isotropic turbulence

(an ergodic random process), the averages of Equations (2.1), (2.2) and (2.3) are all equal.

Reynolds averaging procedures were introduced for incompressible flows, but they can also be applied to the compressible case. In applying Reynolds averaging to compressible flows, one will encounter some additional complexities when establishing suitable closure approximations. It can be shown (see, for example, Wilcox [98]) that the use of the time-averaged procedure for compressible conservation equations yields additional terms without analogs in the laminar equations and further approximations to the correlations between the density and velocity fluctuations are needed. This complexity can be easily seen from the Reynolds-averaged continuity equation given by

$$\frac{\partial \bar{\rho}}{\partial t} + \vec{\nabla} \cdot \left(\bar{\rho} \bar{\vec{u}} + \overline{\rho' \vec{u}'} \right) = 0 \,, \tag{2.4}$$

where $\overline{\rho' \vec{u}'}$ is the Reynolds-averaged correlation between fluctuations of density and velocity, ρ' and $\vec{u}'$, respectively. This term does not appear in the original continuity equation and requires modelling. Reynolds-averaging applied to the momentum equation introduces even further complications.

In 1965, Favre suggested a density-weighted averaging procedure that results in great mathematical simplification to the time-averaged equations. For turbulent combusting flows considered in this thesis work, there are density and temperature fluctuations in addition to velocity and pressure fluctuations and these fluctuations are not small. Therefore, the Favre averaging procedure is employed in this study. From this point on, Favre averaging shall be considered. It is therefore worthwhile to review the Favre (mass-averaging) procedure and some notational conventions in support of understanding the Favre-averaged equations to be described in the next section.

The mass-averaged velocity vector, $\widetilde{\vec{u}}$, is defined by

$$\widetilde{\vec{u}} = \frac{1}{\bar{\rho}} \lim_{\mathrm{T} \longrightarrow \infty} \frac{1}{\mathrm{T}} \int_{t}^{\mathrm{t+T}} \rho(\vec{x}, \tau) \vec{u}(\vec{x}, \tau) d\tau \,, \tag{2.5}$$

where $\bar{\rho}$ is the conventional Reynolds-averaged density and $\vec{u} = \widetilde{\vec{u}} + \vec{u}''$ with $\vec{u}''$ being the mass-weighted fluctuating velocity. Equation (2.5) can be expressed in terms of

conventional Reynolds averaging as

$$\bar{\rho}\widetilde{\vec{u}} = \overline{\rho\vec{u}} = \bar{\rho}\overline{\vec{u}} + \overline{\rho'\vec{u}'}\,. \tag{2.6}$$

Substituting Equation (2.6) into Equation (2.4) one arrives at the Favre-averaged continuity equation given by

$$\frac{\partial\bar{\rho}}{\partial t} + \vec{\nabla}\cdot\left(\bar{\rho}\widetilde{\vec{u}}\right) = 0\,, \tag{2.7}$$

which, as desired, has an identical form to the original (laminar) form of the equation. For simplicity, throughout the rest of this thesis, for mean (mass-averaged) quantities, the symbol "$\widetilde{\ }$" will be dropped. For example, $\vec{u}$ is used to denote the Favre-averaged mean velocity vector; for the Reynolds-averaged density, $\bar{\rho}$, the symbol "$\bar{\ }$" , will also be dropped for convenience.

2.1.2 Favre-Averaged Equations

A mathematical model based on the Favre-averaged Navier-Stokes equations for a compressible thermally perfect reactive mixture of gases has been formulated and is used herein to describe turbulent non-premixed combustion processes. In this formulation, the continuity, momentum, and energy equations for the reactive mixture of N species are

$$\frac{\partial\rho}{\partial t} + \vec{\nabla}\cdot(\rho\vec{u}) = 0\,, \tag{2.8}$$

$$\frac{\partial}{\partial t}(\rho\vec{u}) + \vec{\nabla}\cdot\left(\rho\vec{u}\vec{u} + p\vec{\vec{I}}\right) = \vec{\nabla}\cdot\left(\vec{\vec{\tau}} + \vec{\vec{\lambda}}\right)\,, \tag{2.9}$$

$$\frac{\partial}{\partial t}(\rho e) + \vec{\nabla}\cdot\left[\rho\vec{u}\left(e + \frac{p}{\rho}\right)\right] = \vec{\nabla}\cdot\left[\left(\vec{\vec{\tau}} + \vec{\vec{\lambda}}\right)\cdot\vec{u}\right] + \vec{\nabla}\cdot\left(D_k\,\vec{\nabla}k\right) - \vec{\nabla}\cdot(\vec{q} + \vec{q}_t)\,, \tag{2.10}$$

where ρ is the time-averaged mixture density, $\vec{u}$ is the Favre-averaged mean velocity of the mixture, p is the time-averaged mixture pressure, $\vec{\vec{I}}$ is the identity tensor, $e = |\vec{u}|^2/2 + \sum_{n=1}^{N} c_{\mathrm{n}}h_{\mathrm{n}} - p/\rho + k$ is the Favre-averaged total specific mixture energy with h_{n} being the species enthalpy, k is the specific turbulent kinetic energy, $k = \overline{\vec{u''}\cdot\vec{u''}}$, and D_{k} is the coefficient for the diffusion of the turbulent energy ($D_{\mathrm{k}} =$

$\mu+\mu_{\rm t}\sigma^*$). In addition, μ is the total molecular viscosity of the mixture, $\mu_{\rm t}$ is turbulent eddy viscosity, σ^* is a turbulence model constant, and both are described later in this chapter), $\vec{\vec{\tau}}$ and $\vec{\vec{\lambda}}$ are the molecular and turbulent Reynolds stress tensors (dyads), respectively, and $\vec{q}$ and $\vec{q}_{\rm t}$ are the molecular and turbulent heat flux vectors, respectively. The mixture pressure is given by the ideal gas law

$$p = \sum_{\rm n=1}^{\rm N} \rho c_{\rm n} R_{\rm n} T \,, \tag{2.11}$$

where $R_{\rm n}$ is the species' gas constant and T is the mixture temperature. The molecular fluid stress tensor, $\vec{\vec{\tau}}$, is defined as

$$\vec{\vec{\tau}} = 2\mu(\vec{\vec{S}} - \tfrac{1}{3}\vec{\vec{I}}\,\vec{\nabla}\cdot\vec{u})$$

where $\vec{\vec{S}}$ is the mean strain rate tensor. The transport equation describing the time evolution of the species mass fraction, $c_{\rm n}$, is given by

$$\frac{\partial}{\partial t}(\rho c_{\rm n}) + \ \vec{\nabla}\cdot(\rho c_{\rm n}\vec{u}) = -\ \vec{\nabla}\cdot\Big(\ \vec{\mathcal{J}}_{\rm n} + \ \vec{\mathcal{J}}_{\rm tn}\Big) + \rho\dot{w}_{\rm n}\,, \tag{2.12}$$

where $\dot{w}_{\rm n}$ is the time-averaged or mean rate of the change of the species mass fraction produced by the chemical reactions and $\vec{\mathcal{J}}_{\rm n}$ and $\vec{\mathcal{J}}_{\rm tn}$ are the molecular and turbulent diffusive fluxes for species n, respectively.

Fourier's law is used to represent the thermal diffusion caused by the random thermal motion and turbulence. The molecular heat flux and the laminar diffusive species flux are modelled using Fourier's and Fick's laws, respectively, and given by

$$\vec{q} = -\left(\kappa\ \vec{\nabla}T - \sum_{n=1}^{N} h_{\rm n}\vec{\mathcal{J}}_{\rm n}\right) \tag{2.13}$$

$$\vec{\mathcal{J}}_{\rm n} = -\rho D_{\rm n}\ \vec{\nabla}c_{\rm n} \tag{2.14}$$

where κ is the thermal conductivity of mixture, $D_{\rm n}$ is the molecular diffusivity of species n relatively to the major species and obtained from the given Schmidt number Sc using the relation of $D_{\rm n}=\mu/\rho Sc$, and $h_{\rm n}$ is the absolute (chemical and sensible) internal enthalpy for species n. It is important to point out that the time-averaged reaction rate is a key problem in turbulent combustion modelling. Details for modelling this source term are given in Section 2.4.2.

Turbulent contributions to thermal conductivity and species diffusivity are modelled by making an analogy between momentum and heat transfer. The turbulent heat flux and the turbulent species flux are modelled in a similar fashion to their molecular counterparts and given by

$$\vec{q_t} = -\left(\kappa_t \vec{\nabla} T - \sum_{\mathrm{n}=1}^{\mathrm{N}} h_{\mathrm{n}} \, \vec{\mathcal{J}}_{\mathrm{tn}} \right), \tag{2.15}$$

$$\vec{\mathcal{J}}_{\mathrm{tn}} = -\rho D_{\mathrm{tn}} \vec{\nabla} c_{\mathrm{n}} \,, \tag{2.16}$$

where κ_{t} is the turbulent thermal conductivity of the mixture and D_{tn} is the turbulent molecular diffusivity of species n. Introducing the turbulent Prandtl and Schmidt numbers, $\mathrm{Pr_t}$ and $\mathrm{Sc_t}$, both of which are taken to be constant ($\mathrm{Pr_t} = 0.9$ and $\mathrm{Sc_t} = 1$), assume $\kappa_{\mathrm{t}} = \mu_{\mathrm{t}} c_{\mathrm{p}} / \mathrm{Pr_t}$ and $D_{\mathrm{tn}} = \mu_{\mathrm{t}} / \rho \mathrm{Sc_t}$.

Note that employing constant values for turbulent Prandtl/Schmidt numbers over the domain of interest assumes that the scalar fluctuations are proportional to the local velocity fluctuations. If a uniform proportionality constant can be applied, then no further equations need to be solved. However, the assumption of constant values of turbulent Prandtl and Schmidt numbers becomes problematic as the flow becomes more complex and, in general, substantive variations of these parameters in different regions of the flow are to be expected. For flows where high Mach number compressibility effects must be considered and variations due to compressibility can occur for both turbulent Prandtl and Schmidt numbers, models that allow for variable turbulent Prandtl- and Schmidt-numbers may be required [99].

The averaging process for the Navier-Stokes equations described above has resulted in some unknown quantities, for example, the turbulent Reynolds stress tensor, $\vec{\vec{\lambda}} = -\overline{\rho \vec{u}'' \vec{u}''}$. The correlations reflect the influence of the turbulence on the mean flow solution. Modelling of these unknowns in terms of the mean or averaged variables is required to close the system of equations. The role of the turbulence modelling is to provide appropriate approximations for these unknown or unclosed terms. The turbulence modelling closure adopted in this thesis work is discussed in the next section to follow.

2.2 Turbulence Model

2.2.1 Introduction

The modified two-equation k-ω model of Wilcox [98] is used in this thesis work to model the unresolved turbulent flow quantities. The brief review of the turbulence modelling provided in what follows has been intentionally kept short, as our primary objective is to present an overview of the selected turbulence model. However, it is extremely important to keep in mind that the quality of a reacting-flow simulation will heavily depend on the performance of the turbulence model. Discussion of issues (such as boundary conditions, Reynolds and Schmidt number effects, applicability of a model to a particular flow) that can greatly affect the performance of a turbulence model are beyond the scope of this thesis. Care must be given when applying these two-equation turbulence models to complex shear flows. Refer to the textbooks by Wilcox [98], Pope [100], and Fox [12] for in-depth discussions of scalar-field evolution turbulence modelling techniques based on the eddy-viscosity concept.

As discussed above, the Reynolds stresses, $-\overline{\rho\vec{u}''\vec{u}''}$, an unclosed term resulting from the Favre-averaging process, must be modelled. As in most conventional turbulence models, the Boussinesq approximation is used to relate the Reynolds stress tensor, $\vec{\vec{\lambda}}$, to the mean flow strain-rate tensor using a turbulent eddy viscosity, μ_{t},

$$\vec{\vec{\lambda}} = -\overline{\rho\vec{u}''\,\vec{u}''} = 2\mu_{\mathrm{t}}(\vec{\vec{S}} - \frac{1}{3}\vec{\vec{I}}\vec{\nabla}\cdot\vec{u}) - \frac{2}{3}\vec{\vec{I}}\rho k\,. \tag{2.17}$$

Equation (2.17) converts the problem of modelling the six components of the Reynolds stress tensor to one of modelling the scalar field μ_{t}.

The simplest models for the turbulent eddy viscosity require no additional transport equations and are classified as algebraic models. For example, the assumption that μ_{t} is constant can be applied to only a small class of turbulent flows. Mixing length models relate the turbulent eddy viscosity to the mean rate of strain by introducing a characteristic mixing length. These models have limited applicability but are rather simple to apply. Readers interested in more background information on algebraic models for the turbulent eddy viscosity should consult Wilcox [98] and Pope [100].

In terms of complexity and predictive capabilities, the next level of turbulence modelling introduces a transport equation to describe the variation of the turbulent eddy viscosity throughout the flow domain and can be categorized as one-equation models. While other approaches are possible [101] in many cases, the turbulent eddy viscosity is modelled by

$$\mu_{\mathrm{t}}(\vec{x},t) = l_{\mathrm{mix}}\sqrt{k(\vec{x},t)}\,, \tag{2.18}$$

where l_{mix} is the mixing length assumed to be known or given by an algebraic relationship. The corresponding velocity scale is determined from the turbulent kinetic energy and the model is closed by solving the transport equation for k of the form

$$\frac{\partial}{\partial t}(\rho k) + \vec{\nabla}\cdot(\rho k\vec{u}) = \vec{\vec{\lambda}} : \vec{\nabla}\vec{u} + \vec{\nabla}\cdot\left[\left(\mu + \frac{\mu_{\mathrm{t}}}{\sigma_{\mathrm{k}}}\right)\vec{\nabla}k\right] - \rho\epsilon\,, \tag{2.19}$$

where ϵ is the dissipation rate of the turbulent kinetic energy and takes on the form $\epsilon \sim k^{\frac{3}{2}}/l_{\mathrm{mix}}$. The principal weakness of the one-equation model is that the mixing length must be specified by the user or determined from an algebraic relation. Indeed, for complex turbulent flows, it is unreasonable to expect that the mixing length can be adequately represented by an algebraic relationship.

In order to effectively remove the need to specify the mixing length, a second transport equation can be introduced. A widely used two-equation model is the k-ϵ model, wherein a transport equation for the turbulent dissipation rate is formulated [98]. The standard k-ϵ model employs the following equation for ϵ:

$$\frac{\partial}{\partial t}(\rho\epsilon) + \vec{\nabla}\cdot(\rho\epsilon\vec{u}) = \mathcal{C}_{\epsilon 1}\frac{\epsilon}{k}\,\vec{\vec{\lambda}} : \vec{\nabla}\vec{u} + \vec{\nabla}\cdot\left[\left(\mu + \frac{\mu_{\mathrm{t}}}{\sigma_{\epsilon}}\right)\vec{\nabla}\epsilon\right] - \mathcal{C}_{\epsilon 2}\frac{\rho\epsilon^2}{k}\,, \tag{2.20}$$

where the model constants have been "tuned" by fitting model predictions to experimental data for canonical flows: $\mathcal{C}_{\epsilon 1} = 1.44$, $\mathcal{C}_{\epsilon 2} = 1.92$, and $\sigma_{\epsilon} = 1.3$. The turbulent eddy viscosity is specified $\nu_{\mathrm{t}} = \mathcal{C}_{\mu}\frac{k^2}{\epsilon}$ and $\mathcal{C}_{\mu} = 0.09$. However, it must be recognized that different values will be needed to model specific flows accurately, and thus the "standard" values represent a compromise chosen to give the "best overall" results [98, 100].

Other two-equation models have been proposed and developed based on the time evolution of alternative scalar fields. One of the more popular, perhaps, is the k-ω

model which is used in this thesis work. One noteworthy advantage of the k-ω model over the k-ϵ model is its treatment of the near-wall region in boundary-layer flows, especially for low-Reynolds-number flows. The k-ω model can be applied well into the viscous sub-layer, while, without modifications in the form of damping functions, the k-ϵ model requires the first grid point away from the wall to lie in the log layer.

2.2.2 k-ω Model

In the k-ω model, the turbulent eddy viscosity is prescribed by $\mu_{\rm t} = \rho k/\omega$. Transport equations are solved for turbulent kinetic energy, k, and the specific dissipation rate, ω, given by

$$\frac{\partial}{\partial t}(\rho k) + \vec{\nabla}\cdot(\rho k \vec{u}) = \vec{\vec{\lambda}} : \vec{\nabla}\vec{u} + \vec{\nabla}\cdot\Big[(\mu + \mu_t \sigma^*)\ \vec{\nabla} k\Big] - \beta^* \rho k \omega\,, \tag{2.21}$$

$$\frac{\partial}{\partial t}(\rho \omega) + \vec{\nabla}\cdot(\rho \omega \vec{u}) = \alpha \frac{\omega}{k}\vec{\vec{\lambda}} : \vec{\nabla}\vec{u} + \vec{\nabla}\cdot\Big[(\mu + \mu_t \sigma)\,\vec{\nabla}\omega\Big] - \beta \rho \omega^2\,, \tag{2.22}$$

where σ^*, β^*, α, σ, and β are closure coefficients for the two-equation model. The latter are given by

$$\alpha = \frac{13}{25}\,, \quad \beta = \beta_\circ f_\beta\,, \quad \beta^* = \beta^*_\circ f_{\beta^*}\,, \quad \sigma = \sigma^* = \frac{1}{2}\,, \tag{2.23}$$

with

$$\beta_\circ = \frac{9}{125}\,, \quad \beta^*_\circ = \frac{9}{100}\,, \tag{2.24}$$

$$f_\beta = \frac{1 + 70\chi_\omega}{1 + 80\chi_\omega}\,, \quad f_{\beta^*} = \begin{cases} 1 & \chi_k \le 0\,, \\ \dfrac{1 + 680\chi_k^2}{1 + 400\chi_k^2} & \chi_k > 0\,, \end{cases}\,, \tag{2.25}$$

and

$$\chi_\omega = \left|\frac{(\vec{\vec{\Omega}} \otimes \vec{\vec{\Omega}}) : \vec{\vec{S}}}{(\beta^*_\circ \omega)^3}\right|\,, \quad \chi_k = \frac{1}{\omega^3}\vec{\nabla} k \cdot \vec{\nabla}\omega\,. \tag{2.26}$$

The tensors $\vec{\vec{\Omega}}$ and $\vec{\vec{S}}$ are the vorticity and strain rate tensors, respectively.

Note that the Morkovin's hypothesis is invoked here in the modelling of turbulent quantities for compressible flows [98]. The effects of compressibility on the turbulence are assumed small and, only the variation of mean density is take into account for in the turbulence model formulation. In particular, no modifications are made to the k-ω turbulence model coefficients in terms of compressibility-related corrections.

2.2.3 Near-Wall Turbulence Treatment

Both low-Reynolds-number and wall-function formulations of the k-ω model are used for the treatment of near-wall turbulent flows, with a procedure for automatically switching from one to the other, depending on mesh resolution. In the case of the low-Reynolds-number formulation, it can be shown that

$$\lim_{y\to 0}\omega = \frac{6\nu}{\beta y^2}\,, \tag{2.27}$$

where y is the distance normal from the wall [98]. Rather than attempting to solve the ω-equation directly, the preceding expression is used to specify ω for all values of $y^+ \leq 2.5$, where $y^+ \equiv u_\tau y/\nu$. Note that u_τ is the friction velocity defined by $u_\tau^2 = \tau_{\rm w}/\rho$ where $\tau_{\rm w}$ is the wall shear stress. The procedure for obtaining this value is given below. Provided there are 3-5 computational cells inside $y^+ = 2.5$, this procedure reduces numerical stiffness, guarantees numerical accuracy, and permits the k-ω model to be solved directly in the near-wall region without resorting to wall functions. In the case of the wall-function formulation, the expressions

$$k = \frac{u_\tau^2}{\sqrt{\beta_{\rm o}^*}}\,, \qquad \omega = \frac{u_\tau}{\sqrt{\beta_{\rm o}^*}\kappa y}\,, \tag{2.28}$$

are used to fully specify k and ω for $y^+ \leq$ 30-250, where κ is the von Kármán constant, 0.41.

In this thesis research, a procedure has also been developed to automatically switch between these two approaches, depending on the near-wall mesh resolution. In this procedure, the values of k and ω are approximated by

$$k = \frac{u_\tau^2}{\sqrt{\beta_{\rm o}^\star}}\left(\frac{\min(\mathrm{y}^+, 30)}{30}\right)^2\,, \qquad \omega = \omega_{\rm o}\sqrt{1+\left(\frac{\omega_{\rm wall}}{\omega_{\rm o}}\right)^2}\,, \tag{2.29}$$

where $\omega_{\rm o} = \frac{6\nu}{\beta y^2}$ and $\omega_{\rm wall} = \frac{u_\tau}{\sqrt{\beta^*}\kappa y}$. This automatic near-wall treatment readily accommodates situations during adaptive mesh refinement (as described in Chapter 4) where the mesh resolution may not be sufficient for directly calculating near-wall turbulence using the low-Reynolds-number formulation.

The quantity, y^+, is the dimensionless distance from the wall surface and is defined by $y^+ \equiv u_\tau y/\nu$. This term must be evaluated in order to apply above boundary

conditions for turbulent kinetic energy and specific dissipation rate during solution procedures. The distance normal from the wall surface, y, must be computed after the grid is created and/or is refined (coarsened), and the minimum value of y is determined for each cell after searching all the possible values computed from all the solid wall surfaces. The friction velocity, u_τ, is determined either directly or iteratively using Newton's method by using the relations below depending on the flow regime:

$$u^+ = \begin{cases} y^+ & \text{for viscous sublayer}\,, \\ \frac{1}{\kappa}\ln(y^+) + C & \text{for log layer} \quad \kappa \approx 0.41\,, C = 5.0\,. \end{cases} \tag{2.30}$$

Equation (2.31) below, derived by substituting $u^+ \equiv u/u_\tau$ and $y^+ \equiv u_\tau y/\nu$ into Equation (3.30), is used to determine the friction velocity

$$f(u_\tau) = \begin{cases} u_\tau - \sqrt{\frac{u\nu}{y}} & \text{for viscous sublayer}\,, \\ u_\tau - \frac{\kappa u}{\ln(E\ u_\tau\ y/\nu)} & \text{for log layer} \quad E = e^{\kappa C}\,. \end{cases} \tag{2.31}$$

2.3 Thermodynamic and Transport Properties

In addition to the turbulence quantities, thermodynamic relationships and transport coefficients are also required to close the system of equations given above. In this study, the compressible reactive gaseous mixture is assumed to be thermally perfect, i.e., a gaseous mixture in which the specific heats are only functions of temperature [102].

Thermodynamic and molecular transport properties of each gaseous species are prescribed using the empirical database compiled by Gordon and McBride [103, 104], which provides curve fits for the species enthalpy, h_n; specific heat, c_{p_n}; entropy; viscosity, μ_n; and thermal conductivity, κ_n, as functions of temperature, T. For example, the enthalpy and viscosity for a particular species are given by

$$\begin{aligned} h_\mathrm{n} = R_\mathrm{n}T - a_{1,\mathrm{n}}T^{-2} + a_{2,\mathrm{n}}T^{-1}\ln T + a_{3,\mathrm{n}} + \frac{a_{4,\mathrm{n}}}{2}T + \frac{a_{5,\mathrm{n}}}{3}T^2 + \\ \frac{a_{6,\mathrm{n}}}{4}T^3 + \frac{a_{7,\mathrm{n}}}{5}T^4 + b_1T^{-1} + \Delta h^o_{\mathrm{f_n}}\,, \end{aligned} \tag{2.32}$$

$$\ln\mu_\mathrm{n} = A_\mathrm{n}\ln T + \frac{B_\mathrm{n}}{T} + \frac{C_\mathrm{n}}{T^2} + D_\mathrm{n}\,, \tag{2.33}$$

where $a_{\mathrm{k,n}}$, A_{n}, B_{n}, C_{n}, and D_{n} are the coefficients for the curve fits. The Gordon-McBride data set contains curve fits for over 2000 substances, including 50 reference elements.

The molecular viscosity, μ, and thermal conductivity, κ, of the reactive mixture are determined using the mixture rules of Wilke [105] (Equation (2.34)) and Mason and Saxena [106] (Equation (2.35)), respectively, and are given by

$$\mu = \sum_{\mathrm{n=1}}^{\mathrm{n=N}} \frac{\mu_{\mathrm{n}} \frac{\rho c_{\mathrm{n}}}{\mathcal{M}_{\mathrm{n}}}}{\sum_{\mathrm{j=1}}^{\mathrm{j=N}} \frac{\rho c_{\mathrm{n}}}{\mathcal{M}_{\mathrm{n}}} \phi_{\mathrm{i,j}}} \,, \tag{2.34}$$

$$\kappa = \sum_{\mathrm{n=1}}^{\mathrm{N}} \frac{\kappa_{\mathrm{n}}}{1 + \frac{1}{X_{\mathrm{n}}} \sum_{\mathrm{i=1,i\neq n}}^{\mathrm{N}} (X_{\mathrm{i}} G_{\mathrm{ni}})} \,. \tag{2.35}$$

2.4 Closure of the Chemical Source Terms

2.4.1 Reduced Chemical Kinetics

The primary goal of this research is to establish a computational framework for predicting complex reacting flows in practical combustor geometries. For this purpose, the use of simplified chemical mechanisms for gaseous fuels and turbulence-chemistry interaction models has allowed for the validation of the proposed solution algorithm without the added complexities and computational overhead of more complex mechanisms and sophisticated turbulence-chemistry interaction models.

For the gaseous methane-air combustion considered in the present work, the following reduced, one-step, five-species, chemical kinetic scheme of Westbrook and Dryer [107] is used:

$$\mathrm{CH_4} + 2\ \mathrm{O_2} \rightarrow \mathrm{CO_2} + 2\ \mathrm{H_2O}\,. \tag{2.36}$$

The five species are methane (CH_4), oxygen (O_2), carbon dioxide (CO_2), water (H_2O), and nitrogen (N_2). Nitrogen is taken to be inert.

2.4.2 Modelling Turbulence/Chemistry Interactions

The mean reaction rates, $\dot{\omega}_{\mathrm{n}}$, in Equation (2.12) describe the mean production and consumption of each of the chemical species due to the chemical reactions and strong

interactions between turbulence and chemistry. The accurate prediction of mean reaction rates, which can be strongly influenced and enhanced by small-scale turbulent mixing, represents the central problem and challenge of turbulent combustion. A large number of mean reaction rate formulations may be found in the literature [11, 108–110]. Various mean reaction rate models are described and discussed by Bray [111]. In this work, we shall focus on the eddy dissipation model (EDM) [112].

The interaction between turbulence and chemical reactions is best characterized in terms of the turbulent Damköhler number, which is defined as the ratio of the characteristic turbulent flow time, τ_{t}, to the characteristic chemical time, τ_{c}, i.e.,

$$D_{\mathrm{a}} = \frac{\tau_{\mathrm{t}}}{\tau_{\mathrm{c}}}\,. \tag{2.37}$$

As the turbulent Damköhler number approaches infinity, the chemical time scales are much smaller than those of the fluid dynamics. In this case, equilibrium (fast) chemistry can be assumed. If the Damköhler number is close to zero, the chemical reactions occur more slowly compared to fluid transport phenomena, and then a frozen-chemistry fluid can be assumed. The greatest interaction between turbulence and chemical reactions will occur when the Damköhler number is of the order of unity and in this case one must use finite-rate chemistry to model the chemical reactions.

The Arrhenius approach can be used to describe the mean reaction rates (kinetically controlled) for chemical species by neglecting the effects of turbulence on combustion, i.e., the mean reaction rate is assumed to be only function of mean quantities. The formula for the mean reaction rate for species n is given by

$$\dot{\omega_{\mathrm{n}}} = \frac{\mathcal{M}_{\mathrm{n}}}{\rho}\sum_{\mathrm{r}=1}^{\mathrm{Nr}}(\nu_{\mathrm{n,r}}^{''} - \nu_{\mathrm{n,r}}^{'})\Big\{\kappa_{\mathrm{f,r}}\prod_{\mathrm{i}=1}^{\mathrm{N}}\left[\frac{\rho c_{\mathrm{i}}}{\mathcal{M}_{\mathrm{i}}}\right]^{\nu_{\mathrm{i}}^{'}} - \kappa_{\mathrm{b,r}}\prod_{\mathrm{i}=1}^{\mathrm{N}}\left[\frac{\rho c_{\mathrm{i}}}{\mathcal{M}_{\mathrm{i}}}\right]^{\nu_{\mathrm{i}}^{''}}\Big\}\,, \tag{2.38}$$

where $\nu_{\mathrm{n,r}}^{'}$ and $\nu_{\mathrm{n,r}}^{''}$ are the stoichiometric coefficients for the reactants and for the product (related to species n in reaction r), respectively, M_{i} is the molecular mass of species i, $\kappa_{\mathrm{f,r}}$ and $\kappa_{\mathrm{b,r}}$ are forward and backward reaction rates, respectively, and Nr is the total number of reactions. The Arrhenius approach is only applicable for turbulent combustion at very low Damköhler numbers (i.e., $Da \ll 1$ and so $\tau_c \gg \tau_t$), under which conditions the reactants mix rapidly and burn slowly.

For high Damköhler number, the eddy dissipation model has been proposed. The eddy dissipation model is a direct extension to non-premixed/diffusion flames of the eddy break-up closure originally proposed for turbulent premixed combustion [112]. The latter is based on a phenomenological analysis of turbulent combustion assuming high Reynolds ($Re \gg 1$) and Damköhler ($Da \gg 1$) numbers. The key idea is that chemistry does not play an explicit role while turbulent motions control the reaction rate. The mean reaction rate is thus mainly controlled by the characteristic turbulent time for turbulent mixing.

What follows now is a brief description of the eddy dissipation model for non-premixed combustion. Consider a simple, single step and irreversible chemical reaction between a fuel and its oxidizer:

$$F + s\, O \rightarrow (1+s)\, P$$

where s is the mass stoichiometric coefficient. In terms of mass fractions, this chemical reaction may be written;

$$\nu_{\mathrm{F}}^{'}\, c_{\mathrm{F}} + \nu_{\mathrm{O}}^{'}\, c_{\mathrm{O}} \rightarrow \nu_{\mathrm{P}}^{''}\, c_{\mathrm{P}}\,,$$

where $\nu_i^{'}$ and $\nu_i^{''}$ are the stoichiometric coefficients of the reactants and products, respectively. For non-premixed flames, Magnussen and Hjertager in 1976 proposed the following eddy dissipation model for estimating the mean reaction rates:

$$\dot{\omega}_{\mathrm{F}} = -\mathcal{C}_{\mathrm{edm}}\, \frac{1}{\tau_{\mathrm{t}}}\, \min\left(c_{\mathrm{F}}, \frac{c_{\mathrm{O}}}{s}, \beta\frac{c_{\mathrm{P}}}{(1+s)}\right), \tag{2.39}$$

where model constants, $\mathcal{C}_{\mathrm{edm}}$ and β can be adjusted to incorporate various chemical features. In this study, $\mathcal{C}_{\mathrm{edm}} = 4.0$ and $\beta = 0$. Clearly, the reaction rate is limited by the deficient species and the turbulence mixing time. When $\beta \neq 0$, the products can also limit the rate since "this accounts for the burnt gases bringing the energy to burn the fresh reactants" [108].

The turbulent time scale, τ_{t}, is estimated from the dissipation rate per unit turbulent kinetic energy, ω, and given by

$$\tau_t \propto \frac{1}{\omega}$$

Accordingly, $\dot{\omega}_{\mathrm{O}}$ and $\dot{\omega}_{\mathrm{P}}$ may be computed by

$$\dot{\omega}_{\mathrm{O}} = s\,\dot{\omega}_{\mathrm{F}}\,,$$
$$\dot{\omega}_{\mathrm{P}} = -(1+s)\,\dot{\omega}_{\mathrm{F}}\,.$$

The mass stoichiometric coefficient, s, can be related to the stoichiometric coefficients $\nu_i^{'}$ by

$$s = \frac{\nu_{\mathrm{O}}^{'}\mathcal{M}_{\mathrm{O}}}{\nu_{\mathrm{F}}^{'}\mathcal{M}_{\mathrm{F}}}\,,$$
$$1+s = \frac{\nu_{\mathrm{P}}^{''}\mathcal{M}_{\mathrm{P}}}{\nu_{\mathrm{F}}^{'}\mathcal{M}_{\mathrm{F}}}\,,$$

where $\mathcal{M}_{\mathrm{F}}$, $\mathcal{M}_{\mathrm{O}}$ and $\mathcal{M}_{\mathrm{P}}$ are the molecular mass for fuel, oxidizer and product, respectively. In terms of molar stoichiometric coefficients, Equation (2.39) can be written as:

$$\dot{\omega_{\mathrm{F}}} = -\mathcal{C}_{\mathrm{edm}}\;\omega\;\nu_{\mathrm{F}}^{'}\mathcal{M}_{\mathrm{F}}\;\min\left(c_{\mathrm{F}}, \frac{c_{\mathrm{O}}}{\nu_{\mathrm{O}}^{'}\mathcal{M}_{\mathrm{O}}}, \beta\frac{c_{\mathrm{P}}}{\nu_{\mathrm{P}}^{''}\mathcal{M}_{\mathrm{P}}}\right). \tag{2.40}$$

The eddy dissipation model is manifestly easy to adopt for computational implementation because the reaction rate is calculated using mean quantities of temperatures and mass fractions without additional transport equations. It is useful for the prediction of diffusion flames as well as for partially premixed flames. However, extensions of this model to full chemistry mechanisms is not straightforward [108]. In this work, the individual species mean reaction rate is taken to be the minimum of the rates given by the finite-rate chemical kinetics (i.e., the law of mass action and Arrhenius reaction rates, Equation (2.38)) and the EDM value (Equation (2.39)) due to regions with different turbulence levels. In regions with high-turbulence levels, the eddy lifetime is short, so mixing is fast and, as a result, the reaction rate is kinetically controlled. On the other hand, in regions with low-turbulence levels, small scale mixing may be slow and limits the reaction rate. In this limit, the turbulent mixing rates are more important. Accordingly, the mean reaction rate is primarily controlled by the turbulent time scale τ_{t} (i.e., the time scale for the mixing of fuel and oxidizer by the turbulent motion) [108].

Chapter 3

Finite-Volume Scheme

This chapter presents the main elements of the finite-volume scheme, the time-marching scheme, and the full approximate storage (FAS) multigrid algorithm used in the proposed numerical algorithm. Section 3.1 briefly summarizes the key elements involved in the finite-volume scheme and presents the details on the numerical flux evaluations including both hyperbolic and elliptic parts. Section 3.2 presents the time marching scheme employed to integrate the coupled system of nonlinear ordinary differential equations in time. For two-dimensional flows, a preconditioned multigrid algorithm with a multi-stage time marching scheme has been implemented and is used to improve convergence. Section 3.3 reviews the preconditioned multigrid concept and the matrix preconditioner devised in this study. The incorporation of the finite-volume scheme within a parallel block-based AMR procedure is discussed later in Chapter 4.

3.1 Finite-Volume Method

The finite-volume method used herein starts from the integral form of the conservation equations. Applying the divergence theorem to the differential form of the system of governing equations, (Equations (2.8)–(2.10), (2.12), (2.21), and (2.22)),

one arrives at the integral form

$$\frac{d}{dt}\int_{V(t)} \mathbf{U} dV + \oint_{\Omega(t)} \vec{n}\cdot\vec{\mathbf{F}}\, d\Omega = \int_{v(t)} \mathbf{S} dV\,, \tag{3.1}$$

where $\mathbf{U}$ is the vector of flow solution variables, $\vec{\mathbf{F}}$ is the flux dyad, $\mathbf{S}$ is the source vector, V is the control volume, Ω is the closed surface of the control volume, and $\vec{n}$ is the unit outward vector normal to the closed surface. In the finite-volume method, the integral form of the conservation law is enforced discretely in each of many small contiguous control volumes making up a computational mesh. The conservation of any flow property, for example mass, momentum or energy, within each finite control volume can be expressed as a balance between the net solution fluxes and sources tending to increase or decrease its value. Solution fluxes can be generally categorized as either arising from wave propagation phenomena (hyperbolic fluxes) or from diffusion processes (elliptic fluxes). The reader is referred to the textbooks by Lomax, Pulliam, and Zingg [113] and by Hirsch [114, 115] for details regarding conservation equations and their properties.

To briefly illustrate the main elements of a finite-volume method, consider the discretization of the Favre-averaged Navier-Stokes equations over a set of control volumes in a two-dimensional axisymmetric coordinate frame where it is assumed that the control volumes (areas in two space dimensions) do not vary with time. First, the averaged value of $\mathbf{U}$ and of $\mathbf{S}$ within the cell can be defined by an integration over the control volume as follows:

$$\overline{\mathbf{U}} \equiv \frac{1}{A}\int_{\mathrm{A}} \mathbf{U}\; dA\,, \tag{3.2}$$

$$\overline{\mathbf{S}} \equiv \frac{1}{A}\int_{\mathrm{A}} \mathbf{S}\; dA\,, \tag{3.3}$$

where A is the cell area. Substituting these definitions into Equation (3.1), the final form of Equation (3.1) for a two-dimensional coordinate frame can be written as

$$\frac{d\overline{\mathbf{U}}}{dt} + \frac{1}{A}\oint_{\Omega} \vec{\mathbf{F}}\cdot\vec{n}\, d\ell = \overline{\mathbf{S}}(\overline{\mathbf{U}})\,, \tag{3.4}$$

where A is the cell area and $d\ell$ is an element of the closed contour containing the control volume or cell of interest. Assuming that the control volume, (i, j), is a

polygon defined by N_f straight-line segments or cell faces (line segments, 4 faces for quadrilateral cells), Equation (3.4) can be rewritten in semi-discrete form as

$$\frac{d\overline{\mathbf{U}}_{\mathrm{i,j}}}{dt} = -\frac{1}{A_{\mathrm{i,j}}} \sum_{\mathrm{m}=1}^{N_{\mathrm{f}}} \vec{\mathbf{F}}_{\mathrm{i,j,m}} \cdot \vec{n}_{\mathrm{i,j,m}} \, \Delta l_{\mathrm{i,j,m}} + \overline{\mathbf{S}}_{\mathrm{i,j}}(\overline{\mathbf{U}}) \,, \tag{3.5}$$

or

$$\frac{d\overline{\mathbf{U}}_{\mathrm{i,j}}}{dt} = -\mathbf{R}_{\mathrm{i,j}}(\overline{\mathbf{U}}) \,, \tag{3.6}$$

where $\vec{n}_{\mathrm{k}}$ and Δl_{k} are the outward unit outward norm and length of the k^{th} cell face (line segment), respectively, and $\mathbf{R}_{\mathrm{i,j}}(\overline{\mathbf{U}})$ is the so-called residual operator for the control volume (i, j).

The solution procedure for solving Equation (3.6) then involves three steps: **reconstruction**, **flux evaluation** and **evolution**. First, given the value of $\overline{\mathbf{U}}$ for each control volume, an approximation to $\mathbf{U}(\vec{x})$ in each control volume is constructed and used to find $\mathbf{U}$ at the boundaries of the control volume. The accuracy of the evaluation of cell-averaged solution and its derivatives for the cell-normal flux evaluation is dictated by the accuracy of this solution reconstruction procedure. Details of the piecewise linear limited reconstruction procedure used in this work are presented in Section 3.1.3. Next, the flux, $\vec{\mathbf{F}}(\overline{\mathbf{U}})$, at the boundary is evaluated as a function of the discontinuous states on either side of the interface, where the discontinuities arise due to the piecewise approximations for $\mathbf{U}$ in each control volume. In this work, the hyperbolic numerical flux for each cell-face is evaluated as the approximate solution of a Riemann problem, such as given by the Roe flux function [116] or the Harten-Lax-van-Leer-Einfeldt (HLLE) flux function [117], while the viscous component of the cell-face fluxes is evaluated by employing a centrally-weighted diamond-path reconstruction procedure as described by Coirier and Powell [118] in the two-dimensional case. In the three-dimensional case, the viscous component of the cell-face fluxes is evaluated using the formula proposed by Mathur and Murthy [119]. The evaluation of the numerical fluxes is discussed in Section 3.1.3. Finally, the solution is evolved forward in time using an appropriate time-marching method thereby obtaining new values for $\overline{\mathbf{U}}$. Details of the time marching method adopted herein can be found in Section 3.2.

The remainder of this chapter outlines aspects of the proposed finite-volume scheme for solution of the Favre-averaged Navier-Stokes equations for a compressible, thermally perfect, reactive, mixture in two and three space dimensions. For notational simplicity, in the remainder of the thesis, the bar sign "$^-$" is dropped for cell-averaged solution and source vectors.

3.1.1 Conservation Forms of Governing Equations

As noted, the present work considers solving turbulent combusting flows for both two-dimensional axisymmetric and three-dimensional coordinate frames. The divergence form of Equation (3.1) is obtained by applying Gauss's theorem to the flux integral, leading to

$$\frac{\partial \mathbf{U}}{\partial t} + \vec{\nabla} \cdot \vec{\mathbf{F}} = \mathbf{S}\,, \tag{3.7}$$

where the flux dyad, $\vec{\mathbf{F}}$, and source vector, $\mathbf{S}$, are given by

$$\vec{\mathbf{F}} = \begin{cases} (\mathbf{F} - \mathbf{F}_\mathrm{v}, \mathbf{G} - \mathbf{G}_\mathrm{v}) \\ (\mathbf{F} - \mathbf{F}_\mathrm{v}, \mathbf{G} - \mathbf{G}_\mathrm{v}, \mathbf{H} - \mathbf{H}_\mathrm{v}) \end{cases} \qquad \mathbf{S} = \begin{cases} -\frac{(\mathbf{S}_\mathrm{a} - \mathbf{S}_\mathrm{aV})}{r} + \mathbf{S}_\mathrm{t} + \mathbf{S}_\mathrm{c} & \text{for axisymmetric}\,, \\ \mathbf{S}_\mathrm{t} + \mathbf{S}_\mathrm{c} & \text{for three dimensions}\,. \end{cases}$$

The solution vector is given by $\mathbf{U}$, $\mathbf{F}$ and $\mathbf{F}_V$ are the inviscid and viscous flux vectors in the radial direction for axisymmetric flows and in the x direction for three-dimensional flows, respectively, and $\mathbf{G}$ and $\mathbf{G}_\mathrm{v}$ are the inviscid and viscous flux vectors in the axial direction for the axisymmetric system and in the y direction for the three-dimensional case, respectively, and $\mathbf{H}$ and $\mathbf{H}_\mathrm{v}$ are the inviscid and viscous flux vectors in the z direction for three-dimensional flows. Finally, $\mathbf{S}_\mathrm{a}$ and $\mathbf{S}_\mathrm{av}$ are the source terms associated with the axisymmetric coordinate, and $\mathbf{S}_\mathrm{t}$ and $\mathbf{S}_\mathrm{c}$ are the source terms associated with the turbulence modelling and finite-rate chemical kinetics.

Equations (2.8)–(2.10), (2.12), (2.21), and (2.22) can be re-expressed for both the two-dimensional (axisymmetric) case as

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial (\mathbf{F} - \mathbf{F}_\mathrm{v})}{\partial r} + \frac{\partial (\mathbf{G} - \mathbf{G}_\mathrm{v})}{\partial z} = -\frac{(\mathbf{S}_\mathrm{a} - \mathbf{S}_\mathrm{av})}{r} + \mathbf{S}_\mathrm{t} + \mathbf{S}_\mathrm{c}\,, \tag{3.8}$$

and the three-dimensional case as

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \left(\mathbf{F} - \mathbf{F}_\mathrm{v}\right)}{\partial x} + \frac{\partial \left(\mathbf{G} - \mathbf{G}_\mathrm{v}\right)}{\partial y} + \frac{\partial \left(\mathbf{H} - \mathbf{H}_\mathrm{v}\right)}{\partial z} = \mathbf{S}_\mathrm{t} + \mathbf{S}_\mathrm{c}\,, \tag{3.9}$$

where r and z denote the radial and axial coordinates in axisymmetric system, and x, y, and z are the coordinates of the three-dimensional Cartesian frame.

The elements of the conserved solution vector, flux vectors, and source vectors are provided in this section for the three-dimensional case. Details of those terms for the two-dimensional case are not repeated here. The components of the solution, flux, and source vectors for the two-dimensional case are given in appendix A. The vector of conserved solution variables for the three-dimensional case, $\mathbf{U}$, is given by

$$\mathbf{U} = \left[\; \rho,\, \rho v_{\mathrm{x}},\, \rho v_{\mathrm{y}},\, \rho v_{\mathrm{z}},\, \rho e,\, \rho k,\, \rho\omega,\, \rho c_1,\, \ldots,\, \rho c_{\mathrm{N}} \; \right]^{\mathrm{T}}, \tag{3.10}$$

and the inviscid and viscous x-direction flux vectors, $\mathbf{F}$ and $\mathbf{F}_{\mathrm{v}}$, can be written as

$$\mathbf{F} = \begin{bmatrix} \rho v_{\mathrm{x}} \\ \rho v_{\mathrm{x}}^2 + p \\ \rho v_{\mathrm{x}} v_{\mathrm{y}} \\ \rho v_{\mathrm{x}} v_{\mathrm{z}} \\ (\rho e + p) v_{\mathrm{x}} \\ \rho k v_{\mathrm{x}} \\ \rho\omega v_{\mathrm{x}} \\ \rho c_1 v_{\mathrm{x}} \\ \vdots \\ \rho c_{\mathrm{N}} v_{\mathrm{x}} \end{bmatrix}, \quad \mathbf{F}_{\mathrm{v}} = \begin{bmatrix} 0 \\ \tau_{\mathrm{xx}} + \lambda_{\mathrm{xx}} \\ \tau_{\mathrm{xy}} + \lambda_{\mathrm{xy}} \\ \tau_{\mathrm{xz}} + \lambda_{\mathrm{xz}} \\ \mathcal{W} - q_{\mathrm{x}} - q_{\mathrm{t_x}} + (\mu + \mu_{\mathrm{t}}\sigma^*)\frac{\partial k}{\partial x} \\ (\mu + \mu_{\mathrm{t}}\sigma^*)\frac{\partial k}{\partial x} \\ (\mu + \mu_{\mathrm{t}}\sigma)\frac{\partial \omega}{\partial x} \\ -\mathcal{J}_{1\mathrm{x}} - \mathcal{J}_{\mathrm{t}1\mathrm{x}} \\ \vdots \\ -\mathcal{J}_{\mathrm{N}\mathrm{x}} - \mathcal{J}_{\mathrm{t}\mathrm{N}\mathrm{x}} \end{bmatrix}, \tag{3.11}$$

where $\mathcal{W} = v_{\mathrm{x}}(\tau_{\mathrm{xx}} + \lambda_{\mathrm{xx}}) + v_{\mathrm{y}}(\tau_{\mathrm{xy}} + \lambda_{\mathrm{xy}}) + v_{\mathrm{z}}(\tau_{\mathrm{xz}} + \lambda_{\mathrm{xz}})$. The y- and z-direction flux vectors $\mathbf{G}$, $\mathbf{G}_{\mathrm{v}}$, $\mathbf{H}$, and $\mathbf{H}_{\mathrm{v}}$ have similar forms and are given in appendix A.

The source vectors, $\mathbf{S}_{\mathrm{t}}$ and $\mathbf{S}_{\mathrm{c}}$, appearing in Equations (3.9) contain terms related to the finite rate chemistry and turbulence modelling and have the form

$$\mathbf{S}_{\mathrm{t}} = \left[\; 0,\, 0,\, 0,\, 0,\, 0,\, \mathcal{P} - \beta^*\rho k\omega,\, \alpha\tfrac{\omega}{k}\mathcal{P} - \beta\rho\omega^2,\, 0,\, \ldots,\, 0 \; \right], \tag{3.12}$$

$$\mathbf{S}_{\mathrm{c}} = \left[\; 0,\, 0,\, 0,\, 0,\, 0,\, 0,\, 0,\, \rho\dot{\omega}_1,\, \ldots,\, \rho\dot{\omega}_{\mathrm{N}} \; \right], \tag{3.13}$$

with

$$\mathcal{P} = \lambda_{\mathrm{xx}}\frac{\partial v_{\mathrm{x}}}{\partial x} + \lambda_{\mathrm{xy}}\left(\frac{\partial v_{\mathrm{x}}}{\partial y} + \frac{\partial v_{\mathrm{y}}}{\partial x}\right) + \lambda_{\mathrm{xz}}\frac{\partial v_{\mathrm{x}}}{\partial z} + \lambda_{\mathrm{yy}}\left(\frac{\partial v_{\mathrm{y}}}{\partial y} + \frac{\partial v_{\mathrm{z}}}{\partial x}\right) + \lambda_{\mathrm{yz}}\left(\frac{\partial v}{\partial z} + \frac{\partial v_{\mathrm{z}}}{\partial y}\right) + \lambda_{\mathrm{zz}}\frac{\partial v_{\mathrm{z}}}{\partial z}, \tag{3.14}$$

and v_x, v_y, and v_z are the x, y, and z velocity components; q_x, q_y, and q_z are the x, y, and z components of the heat flux; τ_{xx}, τ_{xy}, τ_{yy}, τ_{xz}, τ_{yz}, and τ_{zz} are the components of the viscous fluid stresses; and λ_{xx}, λ_{xy}, λ_{xz}, λ_{xz}, λ_{yz}, and λ_{zz} are the Reynolds stresses.

3.1.2 Semi-Discrete Form for Three-Dimensional Flows

Equation (3.5) describes the coupled ordinary differential equations resulting from the finite-volume spatial discretization for two-dimensional flows. For the sake of completeness, the semi-discrete form of the three-dimensional conservation equations arising from application of the finite-volume discretization procedure is also given here. For three dimensions, the system of governing Equation (3.9) is integrated over hexahedral cells of a structured body-fitted multi-block hexahedral mesh. In doing so, the original PDEs are converted to a set of coupled ODEs which can be expressed as

$$\frac{d\mathbf{U}_{i,j,k}}{dt} = -\frac{1}{V_{i,j,k}} \sum_{m=1}^{N_f} \vec{\mathbf{F}}_{i,j,k,m} \cdot \vec{n}_{i,j,k,m} \, \Delta A_{i,j,k,m} + (\mathbf{S}_t + \mathbf{S}_c)_{i,j,k} = \mathbf{R}_{i,j,k}(\mathbf{U}) \,, \qquad (3.15)$$

where $V_{i,j,k}$ is the cell volume, N_f is the total number of cell faces (8 faces for hexahedral computational cells), and $\vec{n}_{i,j,k,m}$ and $\Delta A_{i,j,k,m}$ are the unit outward normal vector and the area of cell-face m, respectively. As mentioned above, the numerical fluxes though the cell boundaries, $\mathbf{F}$, given in Equation (3.15), include contributions from both hyperbolic and elliptic (inviscid and viscous) terms. The evaluation of these numerical fluxes are described next.

3.1.3 Inviscid (Hyperbolic) Flux Evaluation

Godunov-type finite-volume methods make use of the solution of locally one-dimensional Riemann problems. The solution of the Riemann problem provides a means for evaluating the numerical flux function at the cell boundaries. A higher-order Godunov-type finite-volume upwind formulation based on approximate Riemann solvers with a least-squares piece-wise limited linear solution reconstruction

procedure is used to evaluate the components of the hyperbolic solution flux. The emergence of high-resolution Godunov-type methods motivated the design of effective limiters for use in one-dimensional higher-order reconstructions [120]. Algorithms with high resolution in smooth regions and monotone resolution of discontinuities were devised based on the original concepts of nonlinear limiters introduced by Boris and Book [121] and van Leer [122]. These concepts which prevent the occurrence of numerical oscillations, were later generalized via the concept of total variation diminishing (TVD) by Harten [123]. The reader is referred to the paper by van Leer [120] for a systematic review and comparisons of various techniques related to this topic.

Piecewise Limited Linear Reconstruction

The solution reconstruction used in this work can be described as follows. For higher-order accuracy (i.e., second-order in smooth regions), a spatial reconstruction of the solution in each computational cell is required. The values of the left and right solution states at a cell interface are determined by least-squares piece-wise limited linear solution reconstruction. For example, for cell (i, j, k), at the cell interface (i+$\frac{1}{2}$, j, k), the flux has the form

$$\vec{\mathbf{F}}_{(\mathrm{i,j,k,m})} \cdot \vec{n}_{(\mathrm{i,j,k,m})} = \vec{\mathbf{F}}\left(\mathcal{R}\big(\mathbf{W}_{\mathrm{L}}, \mathbf{W}_{\mathrm{R}}, \vec{n}_{(\mathrm{i,j,k,m})}\big)\right),$$

where the $\vec{n}_{(\mathrm{i,j,k,1})}$ with $m = 1$ corresponds to the outward unit norm of the cell interface, $\mathcal{R}$ represents the solution of the Riemann problem, and $\mathbf{W}_{\mathrm{L}}$ and $\mathbf{W}_{\mathrm{R}}$ are the left and right primitive solution vectors from the piece-wise limited linear reconstruction procedure at the cell interface (i+$\frac{1}{2}$, j, k), and are given by

$$\begin{aligned} \mathbf{W}_{\mathrm{L}} &= \mathbf{W}_{\mathrm{i,j,k}} + \Phi_{\mathrm{i,j,k}} \vec{\nabla} \mathbf{W}_{\mathrm{i,j,k}} \cdot d\vec{x}_{\mathrm{L}}\,, \\ \mathbf{W}_{\mathrm{R}} &= \mathbf{W}_{\mathrm{i+1,j,k}} + \Phi_{\mathrm{i+1,j,k}} \vec{\nabla} \mathbf{W}_{\mathrm{i+1,j,k}} \cdot d\vec{x}_{\mathrm{R}}\,. \end{aligned} \tag{3.16}$$

In Equation (3.16), Φ is the slope limiter, $d\vec{x}_{\mathrm{L}} = \vec{x} - \vec{x}_{\mathrm{i,j,k}}$ and $d\vec{x}_{\mathrm{R}} = \vec{x} - \vec{x}_{\mathrm{i+1,j,k}}$ ($\vec{x}$ is the location of interface center), and $\mathbf{W}_{\mathrm{i,j,k}}$ and $\mathbf{W}_{\mathrm{i+1,j,k}}$ are cell-averaged primitive solution vectors.

The slope limiter, Φ, is introduced to limit the solution gradient in order to ensure solution monotonicity. Both the Barth-Jespersen and the Venkatakrishnan slope limiters have been implemented in this algorithm. The Barth-Jespersen limiter is given by

$$\Phi_{\mathrm{i,j,k}} = \begin{cases} \min\left(1, \frac{\mathbf{W}_{\max}-\mathbf{W}_{\mathrm{i,j,k}}}{\mathbf{W}_k-\mathbf{W}_{\mathrm{i,j,k}}}\right) & \text{for} \quad \mathbf{W}_k - \mathbf{W}_{\mathrm{i,j,k}} > 0 \\ \min\left(1, \frac{\mathbf{W}_{\min}-\mathbf{W}_{\mathrm{i,j,k}}}{\mathbf{W}_k-\mathbf{W}_{\mathrm{i,j,k}}}\right) & \text{for} \quad \mathbf{W}_k - \mathbf{W}_{\mathrm{i,j,k}} < 0 \\ 1 & \text{otherwise} \end{cases}, \tag{3.17}$$

where $\mathbf{W}_{\max} = \max(\mathbf{W}_{\mathrm{i,j,k}}, \mathbf{W}_{\text{neighbours}})$, $\mathbf{W}_{\min} = \min(\mathbf{W}_{\mathrm{i,j,k}}, \mathbf{W}_{\text{neighbours}})$, and $\mathbf{W}_k$ is the unlimited reconstructed solution value at the k^{th} flux quadrature point. In many cases, the use of highly-nonlinear limiters such as this can inhibit convergence to steady state solutions. Venkatakrishnan [124] states that the Barth-Jespersen limiter is active in the near-constant regions and responds to oscillations at the noise level which is responsible for the convergence stall. The limiter proposed by Venkatakrishnan, a modification of the Barth-Jespersen limiter to help suppress oscillations in the near-constant regions of the solution in a smooth manner, is expected to ameliorate these convergence issues. The limiter can be expressed as

$$\Phi_{\mathrm{i,j,k}} = \begin{cases} \phi\left(\frac{\mathbf{W}_{\max}-\mathbf{W}_{\mathrm{i,j,k}}}{\mathbf{W}_k-\mathbf{W}_{\mathrm{i,j,k}}}\right) & \text{for} \quad \mathbf{W}_k - \mathbf{W}_{\mathrm{i,j,k}} > 0 \\ \phi\left(\frac{\mathbf{W}_{\min}-\mathbf{W}_{\mathrm{i,j,k}}}{\mathbf{W}_k-\mathbf{W}_{\mathrm{i,j,k}}}\right) & \text{for} \quad \mathbf{W}_k - \mathbf{W}_{\mathrm{i,j,k}} < 0 \\ 1 & \text{otherwise} \end{cases}, \tag{3.18}$$

where $\phi(y)$ is a smooth function given by

$$\phi(y) = \frac{y^2 + 2y}{y^2 + y + 2}\,. \tag{3.19}$$

In practice, the use of limiter freezing can also help in situations where convergence might stall, i.e., the limiter is "frozen" after the residual has dropped to some predefined level. However, for all the numerical solutions obtained in this research work, the limiter freezing technique was not required.

The gradients of the primitive variables, $\vec{\nabla}\mathbf{W}$, are determined by applying a least-squares approach [125], a technique which is suitable for both structured and unstructured mesh and relies on a stencil formed by the nearest and possibly next to nearest

neighbouring cells. For the boundary stencil, a layer of ghost cells containing boundary condition information is used to generalize the procedure without reducing the reconstruction stencil. Refer to Chapter 4 for additional information concerning ghost cells. For a cell-centered discretization in three dimensions, the stencil is formed by joining the nearest twenty-six neighbouring cell centroids. The approximate gradients using the least-squares gradient construction procedure are obtained by minimizing the error defined by

$$\sum_{\mathrm{k}=1}^{k=N} \epsilon_{\mathrm{ik}}^2 = \sum_{\mathrm{k}=1}^{k=N} \left(\Delta \mathbf{W}_{\mathrm{ik}} - \vec{\nabla}\mathbf{W}_{\mathrm{i}} \cdot \vec{dx}_{\mathrm{ik}}\right)^2 , \tag{3.20}$$

where $\Delta \mathbf{W}_{\mathrm{ik}} = \mathbf{W}_{\mathrm{i}} - \mathbf{W}_{\mathrm{k}}$, $\vec{dx}_{\mathrm{ik}} = \vec{x}_{\mathrm{i}} - \vec{x}_{\mathrm{k}}$, and $N = 8$ for two dimensions, or $N = 26$ for three dimensions. The 3 by 3 system of linear algebraic equations resulting from the minimization problem can be expressed as

$$\begin{bmatrix} \overline{(\Delta x)^2} & \overline{\Delta x \Delta y} & \overline{\Delta x \Delta z} \\ \overline{\Delta x \Delta y} & \overline{(\Delta y)^2} & \overline{\Delta y \Delta z} \\ \overline{\Delta x \Delta z} & \overline{\Delta y \Delta z} & \overline{(\Delta z)^2} \end{bmatrix} \begin{bmatrix} \frac{\partial \mathbf{W}}{\partial x} \\ \frac{\partial \mathbf{W}}{\partial y} \\ \frac{\partial \mathbf{W}}{\partial z} \end{bmatrix} = \begin{bmatrix} \overline{\mathbf{W}\Delta x} \\ \overline{\mathbf{W}\Delta y} \\ \overline{\mathbf{W}\Delta z} \end{bmatrix} , \tag{3.21}$$

where

$$\overline{\Delta x^2} = \frac{1}{N} \sum_{\mathrm{k}=1}^{N} \Delta x_{\mathrm{ki}}^2, \tag{3.22}$$

$$\overline{\Delta x \Delta y} = \frac{1}{N} \sum_{\mathrm{k}=1}^{N} \Delta x_{\mathrm{ki}} \Delta y_{\mathrm{ki}}, \tag{3.23}$$

and

$$\overline{\Delta \mathbf{W} \Delta x} = \frac{1}{N} \sum_{\mathrm{k}=1}^{N} \Delta \mathbf{W}_{\mathrm{ki}} \Delta x_{\mathrm{ki}} \, . \tag{3.24}$$

The other terms have a similar form. The above terms only depend on grid geometry and so can be precomputed and stored.

Solutions of the 3×3 linear system represented by Equation (3.21) can be readily obtained using Cramer's rule. Mavriplis [126] suggested that the use of weighting in Equation (3.20) as $\sum_{\mathrm{k}=1}^{k=N} w_{\mathrm{ik}} \epsilon_{\mathrm{ik}}^2$ with weighting factor, $w_{\mathrm{ik}} = \frac{1}{\sqrt{\vec{dx}_{\mathrm{ik}} \cdot \vec{dx}_{\mathrm{ik}}}}$, results in a better

conditioned system; however, for cell-centered discretizations on highly-stretched and curved triangular meshes, it was found that both the unweighted and weighted least-squares constructions fail to provide suitable gradient estimates. A weighted least-squares approach is not used in this work. Alternative solution techniques for the least-squares problem solution, such as QR factorization methods [127, 128] are also not considered but may be the subject of future investigations.

Approximate Riemann Solvers

As discussed above, Godunov-type finite-volume methods require the solution of locally one-dimensional Riemann problems. The Riemann problem is a special initial-value problem with discontinuous initial states and self-similar solutions. It is posed at the interface between adjacent cells. The solution of the Riemann problem provides a means for evaluating the numerical flux function at the cell boundaries. One approach is to make use of an exact solution procedure for the Riemann problem as outlined by Gottlieb and Groth [129]. However, often an approximation is sufficient for use in a finite-volume scheme, since only an interface flux is needed, and the details of the sub-grid solution are averaged out after each time step. The most detailed approximation for the wave system associated with the Riemann problem are found in the solvers of Roe [116] which is based on a local linearization of the flow equations and Osher [130], which replaces shock waves by inverted isentropic waves [131]. A family of solvers in which a smaller or larger number of waves are "lumped" together was presented by Harten, Lax, and van Leer (HLL) [132], which is particularly useful when the detailed Riemann solution is complicated or when a steady flow solution is sought in which certain kinds of waves never appear [133]. A desirable feature of the upwind flux formula based on Roe's approximate Riemann solver, is that it yields a steady normal-shock structure, that contains at most one internal cell, whereas the differential flux formulae of Osher [130] and van Leer [134] include one or two internal cells. This property is lost for shocks oblique to the grid, which serves as a motivation for the search of truly multi-dimensional upwind methods, such as residual distribution schemes [135–137]. Note that a complete review of Riemann solvers is

beyond the scope of the present thesis. The reader is again referred to the paper by van Leer [120] for further details.

This thesis research considers both the Roe's and the Harten-Lax-van Leer Einfeldt (HLLE) approximate Riemann solvers and details related to these two solvers are given next. The Roe's approximate Riemann solver is a rather ingenious way of extending linear wave decomposition, which provides the exact solution to the Riemann problem for linear hyperbolic systems, to the approximate solution of the Riemann problem for nonlinear hyperbolic equations. In order to understand Roe's approach, consider the nonlinear system in one-space dimension of the form

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} = \frac{\partial \mathbf{U}}{\partial t} + A\frac{\partial \mathbf{U}}{\partial x} = 0\,, \tag{3.25}$$

where $A = \frac{\partial \mathbf{F}}{\partial \mathrm{U}}$ is the flux Jacobian matrix. Roe's idea was to linearize the above nonlinear system producing a constant coefficient "locally" linear system

$$\frac{\partial \mathbf{U}}{\partial t} + \hat{A}\frac{\partial \mathbf{U}}{\partial x} = 0\,, \tag{3.26}$$

where $\hat{A}$ is a function of the local left and right values for $\mathbf{U}$. Roe [116] suggested that the following conditions should be imposed on $\hat{A}$, so that the resulting scheme is conservative: (i) for any pair of ($\mathbf{U}_\mathrm{i}$,$\mathbf{U}_{\mathrm{i+1}}$), conservation requires that $\mathbf{F}_{\mathrm{i+1}} - \mathbf{F}_\mathrm{i} = \hat{A}(\mathbf{U}_\mathrm{i}, \mathbf{U}_{\mathrm{i+1}})(\mathbf{U}_{\mathrm{i+1}} - \mathbf{U}_\mathrm{i})$; (ii) for consistency, if $\mathbf{U}_\mathrm{i} = \mathbf{U}_{\mathrm{i+1}} = \mathbf{U}$, then one should have $\hat{A}(\mathbf{U}, \mathbf{U}) = A(\mathbf{U}) = \frac{\partial \mathbf{F}}{\partial \mathbf{U}}$; and (iii) $\hat{A}$ has real eigenvalues with a complete set of linearly independent eigenvectors so that the system remains strictly hyperbolic. Roe's numerical flux then has the form

$$\mathbf{F}(\mathbf{U}_\mathrm{L}, \mathbf{U}_\mathrm{R}, \vec{n}) = \frac{1}{2}(\mathbf{F}_\mathrm{L} + \mathbf{F}_\mathrm{R}) - \frac{1}{2}\sum_{i}^{\mathrm{n}} \hat{\alpha}_\mathrm{i}|\hat{\lambda}_\mathrm{i}|\hat{r}_\mathrm{i}\,, \tag{3.27}$$

where α_i is the wave amplitude (jump in the characteristic variables, related to left eigenvectors of $\hat{A}$) based on the eigensystem decomposition of $\hat{A}$, $\hat{\lambda}_\mathrm{i}$ the wave speed, and $\hat{r}_\mathrm{i}$ is the right eigenvector of Roe's matrix, $\hat{A}$.

Once an approximate linearization is derived for the flux Jacobian matrix (certainly not a trivial problem, details on the construction of the Roe matrix can be found [114–116, 138]), the solution of the Riemann problem in terms of the eigensystem decomposition can be determined directly. For the systems of governing equations of interest here (Equations (3.9)), the Roe-averaged velocity, $\hat{v_\mathrm{x}}$, $\hat{v_\mathrm{y}}$, and $\hat{v_\mathrm{z}}$,

Roe-averaged turbulence quantities, $\hat{k}$, $\hat{\omega}$, and Roe-averaged sensible enthalpy, $\hat{h}$, and species mass fractions, $\hat{c_{\mathrm{N}}}$, are all given by the general formula of

$$\hat{\mathbf{\Psi}} = \frac{\sqrt{\rho_{\mathrm{L}}}\mathbf{\Psi}_{\mathrm{L}} + \sqrt{\rho_{\mathrm{R}}}\mathbf{\Psi}_{\mathrm{R}}}{\sqrt{\rho_{\mathrm{L}}} + \sqrt{\rho_{\mathrm{R}}}}\,, \tag{3.28}$$

where $\mathbf{\Psi}_{\mathrm{L}}$ and $\mathbf{\Psi}_{\mathrm{L}}$ represent the left and the right sate vectors consisting of quantities listed above, and the Roe-averaged density, $\hat{\rho}$, is evaluated by $\hat{\rho} = \sqrt{\rho_{\mathrm{L}}\rho_{\mathrm{R}}}$. The Roe-averaged mixture temperature, $\hat{T}$, as required is then computed iteratively by solving $\hat{T} = f(\hat{h} - \frac{1}{2}\hat{\vec{u}}\cdot\hat{\vec{u}} - \hat{k})$. The averaged mixture pressure, $\hat{p}$, is then obtained from the gas equation of state $\hat{p} = \hat{\rho}\hat{R}\hat{T}$ with the averaged mixture gas constant $\hat{R} = \sum_{\mathrm{n=1}}^{\mathrm{N}} \hat{c}_{\mathrm{i}} R_{\mathrm{i}}$, where N is the total number of species involved in the calculation and R_{i} is the gas constant for individual species. The Roe-averaged specific heats for the mixture, $\hat{c}_{\mathrm{v}}$ and $\hat{c}_{\mathrm{p}}$, are computed as $\hat{c}_{\mathrm{v}} = \sum_{\mathrm{n=1}}^{\mathrm{N}} \hat{c}_{\mathrm{i}}\mathcal{C}_{\mathrm{vi}}(\hat{T})$ and $\hat{c}_{\mathrm{p}} = \sum_{\mathrm{n=1}}^{\mathrm{N}} \hat{c}_{\mathrm{i}}\mathcal{C}_{\mathrm{pi}}(\hat{T})$. The Roe-averaged ratio of specific heats is $\hat{\gamma} = \hat{c}_{\mathrm{p}}/\hat{c}_{\mathrm{v}}$, and finally, the Roe-averaged sound-speed is $\hat{a} = \big(\hat{\gamma} - 1\big)\big(\hat{h} - \frac{1}{2}\hat{\vec{u}}\cdot\hat{\vec{u}} - \hat{k}\big)$. This averaged sound-speed is modified to be $\hat{c} = \sqrt{\hat{a} + \frac{2}{3}\hat{\gamma}\hat{k}}$ to account for the fact that the term, $\frac{2}{3}\rho k$, from the normal Reynolds stresses should be "lumped" together with mixture pressure, since this is a hyperbolic term and needs to be treated in an upwind manner.

The eigenvalues of the matrix $\hat{A}$ can be viewed as the wave speeds of the approximate Riemann problem and the right eigenvectors represent how the solution changes across each of the waves. Roe's scheme, Equation (3.27), requires knowledge of the eigenvalues and right eigenvectors of the flux Jacobian matrices evaluated at the Roe-averaged state and formed in terms of primitive variables. The resulting eigenvalues from the averaged matrix in the x direction are $\hat{\lambda}_1 = \hat{v}_{\mathrm{x}} - \hat{c}$, $\hat{\lambda}_{2,3,4} = \hat{v}_{\mathrm{x}}$, $\hat{\lambda}_5 = \hat{v}_{\mathrm{x}} + \hat{c}$, $\hat{\lambda}_{6,\ldots,\mathrm{N}} = \hat{v}_{\mathrm{x}}$. The matrix of the right eigenvectors for the flux Jacobians

in the x direction is given by

$$\begin{bmatrix}
1 & 1 & . & . & 1 & . & . & . & \cdots & . \\
\hat{v}_{\mathrm{x}} & \hat{v}_{\mathrm{x}} - \hat{c} & . & . & \hat{v}_{\mathrm{x}} + \hat{c} & . & . & . & \cdots & . \\
\hat{v}_{\mathrm{y}} & \hat{v}_{\mathrm{y}} & \hat{\rho} & . & \hat{v}_{\mathrm{y}} & . & . & . & \cdots & . \\
\hat{v}_{\mathrm{z}} & \hat{v}_{\mathrm{z}} & . & \hat{\rho} & \hat{v}_{\mathrm{z}} & . & . & . & \cdots & . \\
\mathcal{H} - \hat{c}_{\mathrm{p}}\hat{T} & \mathcal{H} - \hat{v}_{\mathrm{x}}\hat{c} & \hat{\rho}\hat{v}_{\mathrm{y}} & \hat{\rho}\hat{v}_{\mathrm{z}} & \mathcal{H} + \hat{v}_{\mathrm{x}}\hat{c} & \hat{\rho} & 0 & \hat{\rho}\eta_1 & \cdots & \hat{\rho}\eta_{\mathrm{N}} \\
\hat{k} & \hat{k} & . & . & \hat{k} & \hat{\rho} & . & . & \cdots & . \\
\hat{\omega} & \hat{\omega} & . & . & \hat{\omega} & . & \hat{\rho} & . & \cdots & . \\
\hat{c}_1 & \hat{c}_1 & . & . & \hat{c}_1 & . & . & \hat{\rho} & \cdots & . \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
\hat{c}_{\mathrm{N}} & \hat{c}_{\mathrm{N}} & . & . & \hat{c}_{\mathrm{N}} & . & . & . & \cdots & \hat{\rho}
\end{bmatrix},$$

where $\eta_{\mathrm{i}} = \hat{\epsilon}_{\mathrm{i}} - \frac{\hat{c}_{\mathrm{v}} R_{\mathrm{i}} \hat{T}}{\hat{R}}$, $\mathcal{H}$ is the specific enthalpy given by $\mathcal{H} = \frac{\hat{\vec{v}}\cdot\hat{\vec{v}}}{2} + \hat{h} + \hat{k}$, and $\hat{\epsilon}_{\mathrm{i}}$ is the specific internal energy for species i. The right-eigenvector matrices in the y and the z directions have a similar form, and are given in appendix B.

An entropy fix is necessary to account for the fact that the Roe's approximate Riemann solver cannot reasonably represent expansion waves associated with acoustic waves having wavespeeds $\hat{\lambda}_1$ and $\hat{\lambda}_5$, in which the the expansion fan has a head and tail moving in opposite directions (i.e., near sonic points). The averaged eigenvalues, $|\hat{\lambda}_{\mathrm{k}}|$, in Roe's flux function (Equation (3.27)) are replaced by Harten's entropy fix [139], thereby increasing the magnitude of these two acoustic waves near sonic points such that $|\hat{\lambda}^*_{\mathrm{k}}|$ is given by

$$|\hat{\lambda}^*_{\mathrm{k}}| = \begin{cases} |\hat{\lambda}_{\mathrm{k}}| & \text{if } |\hat{\lambda}_{\mathrm{k}}| \geq \frac{\Delta\lambda_{\mathrm{k}}}{2}\,, \\ \frac{\hat{\mathrm{k}}^2}{\Delta\lambda_{\mathrm{k}}} + \frac{\Delta\lambda_{\mathrm{k}}}{4} & \text{if } |\hat{\lambda}_{\mathrm{k}}| < \frac{\Delta\lambda_{\mathrm{k}}}{2}\,, \end{cases} \tag{3.29}$$

where $\Delta\lambda_{\lambda_{\mathrm{k}}} = \max(0, 4(\lambda_{\mathrm{k}}(\mathbf{U}_{\mathrm{R}}) - \lambda_{\mathrm{k}}(\mathbf{U}_{\mathrm{L}})))\,, \mathrm{k} = 1, 5.$

The Harten, Lax, and van Leer's [132] (HLL) approximate Riemann solver is also used in this work. This approximate solver is based on a two-wave solution to the

Riemann problem. The HLL flux function is given by

$$\mathbf{F}(\mathbf{U}_\mathrm{L},\mathbf{U}_\mathrm{R},\vec{n}) = \begin{cases} \mathbf{F}_\mathrm{L} & \text{if } S_\mathrm{L} \geq 0\,, \\ \frac{S_\mathrm{R}\mathbf{F}_\mathrm{L} - S_\mathrm{L}\mathbf{F}_\mathrm{R} + S_\mathrm{L}S_\mathrm{R}(\mathbf{U}_\mathrm{R}-\mathbf{U}_\mathrm{L})}{S_\mathrm{R}-S_\mathrm{L}} & \text{if } S_\mathrm{L} \leq 0 \leq S_\mathrm{R}\,, \\ \mathbf{F}_\mathrm{R} & \text{if } S_\mathrm{R} \leq 0\,, \end{cases} \tag{3.30}$$

where S_L and S_R are left and right signal velocities. One primary defect of this scheme is exposed by contact discontinuities, shear waves and material interfaces due to the missing intermediate waves. Einfeldt [117] proposed the following estimates for S_L and S_R

$$\begin{aligned} S_\mathrm{R} &= \max(\lambda_\mathrm{R}^{\max}, \hat{\lambda}^{\max}(\mathbf{U}_\mathrm{L},\mathbf{U}_\mathrm{R}))\,, \\ S_\mathrm{L} &= \max(\lambda_\mathrm{L}^{\min}, \hat{\lambda}^{\min}(\mathbf{U}_\mathrm{L},\mathbf{U}_\mathrm{R}))\,, \end{aligned} \tag{3.31}$$

where $\hat{\lambda}$ represents Roe's averaged eigenvalue. This modification produces an effective and robust scheme, which is referred to herein as the HLLE approximate Riemann solver.

Frame Rotation

The hyperbolic numerical flux at each cell face is evaluated by solving a Riemann problem in a rotated frame, whose x-axis is aligned with the normal to the cell face. A rotation matrix, $\mathbf{\Gamma}$, can be used to describe the transformation from the unrotated frame, $\vec{X}$, to a rotated frame, $\vec{x}$, that is $\vec{X} = \mathbf{\Gamma}\vec{x}$. In two dimensions, the transformation is straightforward, the rotated $\vec{x}$ coincides with the face norm $\vec{n}$, and the rotation matrix has the form of:

$$\mathbf{\Gamma} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}\,,$$

where θ is the angle between the rotated $\vec{x}$ coordinate and the unrotated $\vec{X}$ coordinate.

For the three-dimensional case, the frame rotation is expressed here using Euler angles. The orientation of the body-fixed coordinate system with respect to the space-fixed coordinate system is described by three angles, referred to as Euler angles, as

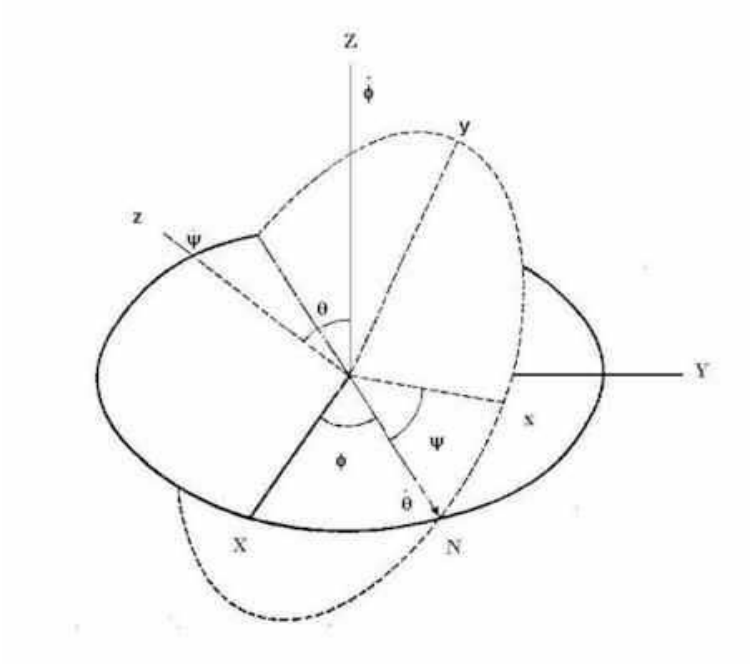


Figure 3.1: Schematic showing three-dimensional frame rotation in terms of the Euler angles.

defined in Figure 3.1. Angle ϕ is the rotation angle about the Z-axis, $\vec{Z}$, θ is about an axis oriented in the direction $\vec{N}$, and ψ is about the z-axis, $\vec{z}$. The transformation matrix can be written as the matrix product $\mathbf{\Gamma} = \chi_{\mathrm{Z}}\chi_\theta\chi_\psi$, where the matrices χ_{Z}, χ_θ, and χ_ψ are given by

$$\chi_{\mathrm{Z}} = \begin{bmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \chi_\theta = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix} \quad \chi_\psi = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} .$$

Let $\mathbf{F}^*$ be the flux in the rotated frame, then flux is given by $\mathbf{F} = \mathbf{\Gamma}^{-1}\mathbf{F}^*$ where the

inverse transformation matrix, $\mathbf{\Gamma}^{-1}$, has the form

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & \cdot & 0 \\ 0 & \cos\psi\cos\phi - \cos\theta\sin\phi\sin\psi & \cos\theta\sin\phi\cos\psi + \sin\psi\cos\phi & \sin\theta\sin\phi & 0 & 0 & 0 \\ 0 & \cos\psi\sin\phi + \cos\theta\cos\phi\sin\psi & \cos\theta\cos\phi\cos\psi - \sin\psi\sin\phi & -\sin\theta\cos\phi & 0 & 0 & 0 \\ 0 & \sin\theta\sin\psi & \sin\theta\cos\psi & \cos\theta & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

For the frame rotations, it can be assumed that, $\phi = 0$, due to the fact that the rotation is not unique and one need only align the rotated x-axis with the face normal direction. The other two angles are determined using the face normal components, $\vec{n} = n_{\mathrm{x}}i + n_{\mathrm{y}}j + n_{\mathrm{z}}k$, and are given by

$$\cos\psi = n_{\mathrm{x}} \quad \sin\psi = \sqrt{n_{\mathrm{y}}^2 + n_{\mathrm{z}}^2} \quad \cos\theta = \frac{n_{\mathrm{y}}}{\sqrt{n_{\mathrm{y}}^2 + n_{\mathrm{z}}^2}} \quad \sin\theta = \frac{n_{\mathrm{z}}}{\sqrt{n_{\mathrm{y}}^2 + n_{\mathrm{z}}^2}}. \quad (3.32)$$

An alternative method can also be utilized to carry out the frame rotation. This second approach does not require the evaluation of the Euler angles due to the fact that the rotated flux results from the rotated velocity vector, since all the scalar quantities are not affected by the rotated frame. Suppose $\mathbf{W}_{\mathrm{s}}$ represents the primitive scalar solution variables (these values are the same for both rotated and unrotated frames), such as density, pressure, mass fractions and so on; then $\vec{\mathbf{F}}^* = \mathcal{F}(\vec{u}^*, \vec{W}_{\mathrm{s}})$. Let $\vec{u}_{\mathrm{L}}$ represent the velocity vector for left state and $\vec{u}_{\mathrm{R}}$ for the right state in the unrotated frame, while letting $\vec{u}_{\mathrm{L}}^*$ and $\vec{u}_{\mathrm{R}}^*$ represent the velocity vectors for the left and the right states, respectively, in the rotated frame which are related as

$$\vec{u}_{\mathrm{L}}^* = \begin{bmatrix} \vec{u}_{\mathrm{L}} \cdot \vec{n} \\ |\vec{u}_{\mathrm{L}} - \vec{n}| \\ 0 \end{bmatrix}, \vec{u}_{\mathrm{R}}^* = \begin{bmatrix} \vec{u}_{\mathrm{R}} \cdot \vec{n} \\ (\vec{u}_{\mathrm{R}} - \vec{n}) \cdot (\vec{u}_{\mathrm{L}} - \vec{n}) \\ |(\vec{u}_{\mathrm{R}} - \vec{n}) \otimes (\vec{u}_{\mathrm{L}} - \vec{n})| \end{bmatrix},$$

$$\vec{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \vec{n} \\ \frac{\vec{u}_{\mathrm{L}} - \vec{n}}{\left| \vec{u}_{\mathrm{L}} - \vec{n} \right|} \\ \frac{(\vec{u}_{\mathrm{R}} - \vec{n}) \otimes (\vec{u}_{\mathrm{L}} - \vec{n})}{\left| (\vec{u}_{\mathrm{R}} - \vec{n}) \otimes (\vec{u}_{\mathrm{L}} - \vec{n}) \right|} \end{bmatrix} .$$

In the case of zero velocity (or zero velocity decomposition) occurring in the left state, then the right state is taken as the reference. The numerical flux is then given by $\vec{\mathbf{F}} = \mathbf{\Gamma}^{-1} \vec{\mathbf{F}^*}$ with $\mathbf{\Gamma}^{-1}$ expressed as

$$\Gamma^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & \vec{n}_{\mathrm{x}} & \frac{(\vec{u}_{\mathrm{L}} - \vec{n})_{\mathrm{x}}}{\left| \vec{u}_{\mathrm{L}} - \vec{n} \right|} & \frac{(\vec{u}_{\mathrm{R}} - \vec{n}) \otimes (\vec{u}_{\mathrm{L}} - \vec{n})_{\mathrm{x}}}{\left| (\vec{u}_{\mathrm{R}} - \vec{n}) \otimes (\vec{u}_{\mathrm{L}} - \vec{n}) \right|} & 0 & \cdots & 0 \\ 0 & \vec{n}_{\mathrm{y}} & \frac{(\vec{u}_{\mathrm{L}} - \vec{n})_{\mathrm{y}}}{\left| \vec{u}_{\mathrm{L}} - \vec{n} \right|} & \frac{(\vec{u}_{\mathrm{R}} - \vec{n}) \otimes (\vec{u}_{\mathrm{L}} - \vec{n})_{\mathrm{y}}}{\left| (\vec{u}_{\mathrm{R}} - \vec{n}) \otimes (\vec{u}_{\mathrm{L}} - \vec{n}) \right|} & 0 & \cdots & 0 \\ 0 & \vec{n}_{\mathrm{z}} & \frac{(\vec{u}_{\mathrm{L}} - \vec{n})_{\mathrm{z}}}{\left| \vec{u}_{\mathrm{L}} - \vec{n} \right|} & \frac{(\vec{u}_{\mathrm{R}} - \vec{n}) \otimes (\vec{u}_{\mathrm{L}} - \vec{n})_{\mathrm{z}}}{\left| (\vec{u}_{\mathrm{R}} - \vec{n}) \otimes (\vec{u}_{\mathrm{L}} - \vec{n}) \right|} & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & 1 \end{bmatrix} ,$$

where the subscripts, $_{\mathrm{x}}$, $_{\mathrm{y}}$, and $_{\mathrm{z}}$, represent the x, y, and z component of a vector, respectively. Both approaches produce the identical numerical solutions, although the latter is probably computationally faster than the former one. The rotation based on the velocities does however have a dependence on the local velocities that may lead to issues with convergence. For this reason, the approach based on the Euler angles was used exclusively in all of this thesis research.

3.1.4 Viscous (Elliptic) Flux Evaluation

Evaluation of the viscous component of the numerical flux in Equation (3.5) depends on both the solution state and its gradients at the cell interfaces and has the

form

$$\vec{\mathbf{F}} \cdot \vec{n} = \vec{\mathbf{F}}(\mathbf{W}_{\mathrm{i}+\frac{1}{2},\mathrm{j},\mathrm{k}}, \vec{\nabla}\mathbf{W}_{\mathrm{i}+\frac{1}{2},\mathrm{j},\mathrm{k}}) \,, \tag{3.33}$$

where $\mathbf{W}_{\mathrm{i}+\frac{1}{2},\mathrm{j},\mathrm{k}}$ is the primitive solution vector at the cell interface which is evaluated by averaging the left and the right reconstructed solution states,

$$\mathbf{W}_{\mathrm{i}+\frac{1}{2},\mathrm{j},\mathrm{k}} = \frac{(\mathbf{W}_{\mathrm{L}} + \mathbf{W}_{\mathrm{R}})}{2} \,. \tag{3.34}$$

The evaluation of the gradients for the primitive variables at the cell interface, $\vec{\nabla}\mathbf{W}_{(\mathrm{i}+\frac{1}{2},\mathrm{j},\mathrm{k})}$, requires some additional work and is described next.

The required gradients can be evaluated at each cell-face by applying the divergence theorem to a polygon, formed by joining the centroids of cells, vertices of cells, or both, in a path surrounding the face. Figures 3.2 illustrates a choice of three different paths in two dimensions: (a) centroidal path; (b) existing faces co-volume; and (c) diamond path on Cartesian grid. The diamond path on a curvilinear grid is shown in Figure 3.2(d). Coirier [39] performed an assessment of a Green-Gauss reconstruction procedure based on these three paths using a generic Laplacian operator (the Laplacian is representative of the viscous stress terms of the incompressible Navier-Stokes equations with a constant viscosity). Each reconstruction path was evaluated on three Cartesian grids: a uniform grid, a uni-directionally stretched grid, and a one-sided refined grid. Coirier found that centroidal path produces decoupling that may lead to a checker-board type of numerical instability. The existing faces co-volume reconstruction path completely decouples all the nearest-layer neighbours on the uniform grid and causes directional decoupling on both the uni-directionally stretched grid and the one-sided refined grid resulting in severe inconsistencies in the scheme. The diamond path with the linearity preserving weighting function proposed by Holmes and Connell forms a proper reconstruction procedure although an inconsistent and non-positive scheme can still be produced for the one-sided refined grid. The present work adopts the procedure of Green-Gauss integration over the diamond path using the linearity-preserving weighting function derived by Holmes and Connell to evaluate the gradients on each cell interface in two space dimensions as

$$\vec{\nabla}\mathbf{W}_{\mathrm{i}+\frac{1}{2},\mathrm{j}} = \frac{\vec{n}}{\vec{n} \cdot \vec{e_{\mathrm{s}}}} \left(\frac{\mathbf{W}_{\mathrm{i}+1,\mathrm{j}} - \mathbf{W}_{\mathrm{i},\mathrm{j}}}{ds} + \frac{\mathbf{W}_{\mathrm{i}+\frac{1}{2},\mathrm{j}+\frac{1}{2}} - \mathbf{W}_{\mathrm{i}+\frac{1}{2},\mathrm{j}-\frac{1}{2}}}{dl} \vec{e_{\mathrm{t}}} \cdot \vec{e_{\mathrm{s}}} \right) . \tag{3.35}$$

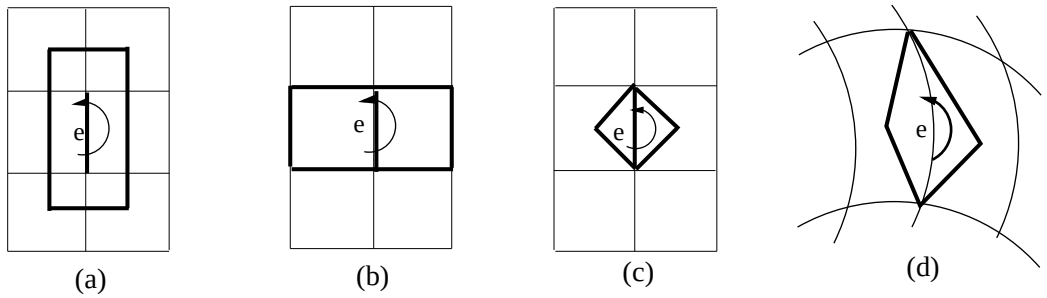


Figure 3.2: Possible reconstruction paths showing: (a) centroidal path; (b) existing faces co-volume; and (c) diamond path on Cartesian grid.

In Equation (3.35), ds is the distance between two centroids, dl is the face length, and unit vectors, $\vec{e_t}$, $\vec{n}$, and $\vec{e_s}$ are the tangential vector, face norm, and the distance vector from the cell centroid to its neighbour's as shown in Figure 3.3.

For three space dimensions, the edges of the diamond path are replaced by surfaces. However, extending Equation (3.35) to three dimensions is not straightforward due to the fact that the face tangential vectors are not uniquely defined for most hexahedral mesh. In this research work, the cell-face gradients are evaluated using the formula proposed by Mathur and Murthy [119]

$$\vec{\nabla}\mathbf{W}\Big|_{\mathrm{i}+\frac{1}{2},\mathrm{j},\mathrm{k}} = \frac{\mathbf{W}_{\mathrm{i}+1,\mathrm{j},\mathrm{k}} - \mathbf{W}_{\mathrm{i},\mathrm{j},\mathrm{k}}}{ds}\frac{\vec{n}}{\vec{n}\cdot\vec{e_s}} + \left(\overline{\vec{\nabla}\mathbf{W}} - \overline{\vec{\nabla}\mathbf{W}}\cdot\vec{e_s}\,\frac{\vec{n}}{\vec{n}\cdot\vec{e_s}}\right), \tag{3.36}$$

where $\overline{\vec{\nabla}\mathbf{W}}$ is the weighted average of the cell centred gradient at the cell interface given by

$$\overline{\vec{\nabla}\mathbf{W}}\Big|_{\mathrm{i}+\frac{1}{2},\mathrm{j},\mathrm{k}} = \alpha\vec{\nabla}\mathbf{W}_{\mathrm{i},\mathrm{j},\mathrm{k}} + (1-\alpha)\vec{\nabla}\mathbf{W}_{\mathrm{i}+1,\mathrm{j},\mathrm{k}}\,. \tag{3.37}$$

The weighting factor, α, is based on cell volume ratios and given by

$$\alpha = V_{\mathrm{i},\mathrm{j},\mathrm{k}}/(V_{\mathrm{i},\mathrm{j},\mathrm{k}} + V_{\mathrm{i}+1,\mathrm{j},\mathrm{k}})\,. \tag{3.38}$$

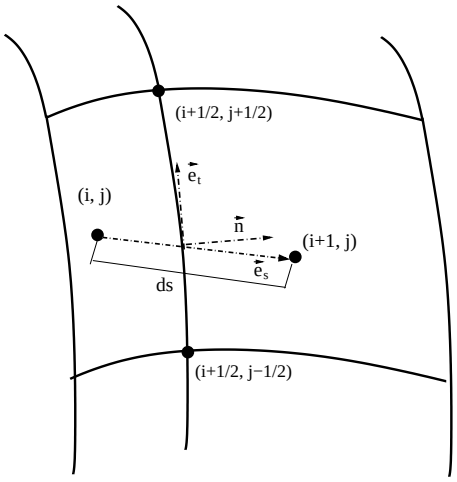


Figure 3.3: Face gradient reconstruction illustration in two space dimensions.

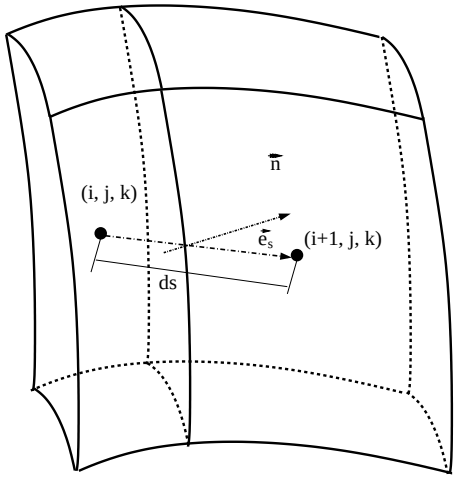


Figure 3.4: Face gradient reconstruction illustration in three space dimensions.

3.2 Time Marching Scheme

The coupled system of nonlinear ODEs given by Equation (3.5) resulting from the finite-volume spatial discretization can be integrated forward in time using a time-marching method, thereby obtaining a time-accurate solution for unsteady problems. For steady flow problems, a time-marching method can also be considered to remove the transient portion of the solution as quickly as possible until the solution is sufficiently close to the steady state. Time accuracy is not required in this case. A wide variety of time-marching methods, such as explicit methods (Euler, Adams-Moulton, and Runge-Kutta methods, etc.) and implicit methods (Euler, Trapezoidal, Runge-Kutta methods, etc.) can be used for these different purposes. The reader is referred to the textbooks by Lomax, Pulliam and Zingg [113] and by Hirsch [114,115] for more information on time marching schemes.

For the time-invariant calculations performed as part of this study, a multi-stage time-marching scheme is used to solve the coupled set of non-linear ODEs (Equation(3.5)) that arise from the finite-volume spatial discretization procedure. The time-marching scheme is based on the optimally-smoothing multi-stage time marching scheme developed by van Leer *et al.* [140]. The general M stage optimally smoothing time-marching scheme for integrating Equation(3.5) from the time level n

to time level $n+1$ can be written as

$$\text{m stage:} \begin{cases} \mathbf{U}^{0}_{\mathrm{i,j,k}} = \mathbf{U}^{\mathrm{n}}_{\mathrm{i,j,k}} \\ \mathbf{U}^{\mathrm{m}}_{\mathrm{i,j,k}} = \mathbf{U}^{0}_{\mathrm{i,j,k}} - \alpha_{\mathrm{m}} \Delta t^{\mathrm{n}} \, \mathbf{R}_{\mathrm{i,j,k}} \left(\mathbf{U}^{\mathrm{m-1}}\right) \; \mathrm{m} = 1 \cdots \mathrm{M} \\ \mathbf{U}^{\mathrm{n+1}}_{\mathrm{i,j,k}} = \mathbf{U}^{\mathrm{M}}_{\mathrm{i,j,k}} \end{cases} ,$$

where Δt^{n}=$t^{\mathrm{n+1}} - t^{\mathrm{n}}$ is the size of the time step and α_{m} are multi-stage coefficients. The coefficients used here have been selected to optimize the high-frequency damping for first- and second-order upwind discretizations of the scalar advection equation in multigrid applications [140]. They are not optimized for diffusion problems or viscous flows. Kleb *et al.* [141] suggested a set of varying multistage coefficients for viscous flows and their adaptive application to multigrid relaxation. Having the coefficients vary with local cell Reynolds number should be beneficial for the turbulent flows. However, this form of optimized time marching scheme was not considered as part of this thesis work. Although a multigrid scheme for two-dimensional flows was considered (to be discussed next), optimizing the solution of the ODEs resulting from the proposed spatial discretization procedure for steady problems was not a primary concern and is left for future follow-on research. As noted in the introduction, the focus of this thesis research was on the development of a highly scalable parallel finite-volume scheme with AMR.

The source terms associated with finite-rate chemistry and turbulence modeling are usually responsible for much of the numerical stiffness in the resulting discretized system of equations. The use of semi-implicit time integration can be utilized to cope with the stiffness of the system. This method treats source terms implicitly, while treating the fluxes explicitly. Hence, this method avoids solving the large block matrices associated with the fully implicit scheme. The semi-implicit form couples with the multistage scheme by replacing the update-stage of the multistage scheme as

$$\left[\vec{I} - \nu \alpha_{\mathrm{m}} \Delta t^{\mathrm{n}} \frac{\partial \mathbf{S}^{(0)}}{\partial \mathbf{U}}\right] \Delta \mathbf{U}^{\mathrm{m}}_{\mathrm{i,j,k}} = \nu \alpha_{\mathrm{m}} \Delta t^{\mathrm{n}} \mathbf{R}_{\mathrm{i,j,k}}(\mathbf{U}^{(\mathrm{m-1})}) \,, \tag{3.39}$$

where $\vec{I}$ is the identity matrix, $\Delta \mathbf{U}^{\mathrm{m}} = \mathbf{U}^{\mathrm{m}} - \mathbf{U}^{0}$ is the solution change, and $\frac{\partial \mathbf{S}^{(0)}}{\partial \mathbf{U}}$ is the source Jacobian term. A local linear system of equations is then solved to obtain

the solution change using a dense matrix solver. In this case a LU decomposition was used.

The inviscid Courant-Friedrichs-Lewy stability, viscous von Neumann stability, and turbulent and chemical time step constraints are imposed when selecting the time step. Note that, for reacting flows, the inverse of the maximum diagonal of the chemical source term Jacobian is added to the time step calculation. The time step, Δt^{n}, is then determined by

$$\Delta t^{\mathrm{n}} = \min\left(\mathrm{CFL}\ \frac{\Delta l}{|\vec{u}|+c}, \frac{\alpha}{2}\frac{\rho\Delta l^2}{\max\left(\mu,\mu_{\mathrm{t}}\right)}, \left(\beta\ \max(\frac{\partial \mathbf{S}_{\mathrm{c}}}{\partial \mathbf{U}})\right)^{-1}\right), \tag{3.40}$$

where Δl is the cell-face length of a cell, c is the sound speed, and μ and μ_{t} are molecular viscosity and turbulent eddy viscosity, respectively, and where α and β are scaling factors.

3.3 Full Approximation Storage Multigrid For Steady Flows

Multigrid techniques have proved to be very effective methods for calculating steady state solutions of both elliptic and hyperbolic partial differential equations [142]. Essentially, the multigrid technique involves using a set of coarser girds to accelerate the convergence of iterative schemes. Standard iterative methods tend to damp out high-frequency components of the error more efficiently than low-frequency ones. With an appropriate coarse-grid approximation of the fine grid system of equations, low-frequency error modes on the fine-grid become high-frequency ones on the coarser grids. Together with an efficient high-frequency error-damping relaxation method, the multigrid technique can dramatically reduce computational cost. Details of multigrid methods may be found in the literature, see for example [143, 144].

A full approximation storage (FAS) multigrid has been developed and applied to the solution of two-dimensional steady state problems as part of this thesis research. Application to the three-dimensional case was not considered. Here, we intentionally

keep the discussion of the multigrid technique brief and only introduce the main elements involved in the multigrid solution procedure employed in this study.

The main elements of a FAS multigrid method can be summarized as follows:

- perform n_1 iterations of the selected relaxation method on the fine-grid.
- transfer both the solution and the residual to the to next coarser grid level (**solution restriction** and **residual restriction**).

$$\mathcal{I}_{\mathrm{k}\rightarrow\mathrm{k}_{\mathrm{k}-1}}\mathbf{U}^{\mathrm{k}} = \frac{1}{\Omega^{\mathrm{k}-1}}\sum_{\mathrm{l}=1}^{\mathrm{n}}\Omega_{\mathrm{l}}^{\mathrm{k}}\mathbf{U}_{\mathrm{l}}^{\mathrm{k}}\,, \tag{3.41}$$

$$\mathcal{I}_{\mathrm{k}\rightarrow\mathrm{k}_{\mathrm{k}-1}}\mathbf{R}^{\mathrm{k}} = \sum_{\mathrm{l}=1}^{\mathrm{k}}\mathbf{R}_{\mathrm{l}}^{\mathrm{k}}\,, \tag{3.42}$$

$$\mathbf{S}^{\mathrm{k}-1} = \mathcal{I}_{\mathrm{k}\rightarrow\mathrm{k}_{\mathrm{k}-1}}\mathbf{S}^{\mathrm{k}}\,. \tag{3.43}$$

  where $\mathcal{I}_{\mathrm{k}\rightarrow\mathrm{k}_{\mathrm{k}-1}}$ is the transfer operator, k and k-1 represents a fine and coarse-grid level, respectively, Ω is the cell area in two space dimensions and cell volume in three space dimensions, and n is 4 in two space dimensions and 8 in three space dimensions.

- perform some relaxation or smoothing cycles on the coarse level (**solution smoothing**), then the solution and residual are restricted to the next coarser level until the coarsest level is reached, and the problem is solved on the coarsest grid level.
- Interpolate the solution back to the finer level (**solution prolongation**): the solution and residual from this finer level are then interpolated to next finer level after some relaxation iterations. Solution prolongation is continued until the finest level is reached.

The above process is repeated until satisfactory convergence is obtained.

Next, some of the difficulties in applying a multigrid convergence technique to turbulent combustion are discussed and several remedies are proposed herein including modifications to the restriction and prolongation operators to improve multigrid convergence and performance.

3.3.1 Smoothing Operator

Multigrid Navier-Stokes solvers can result in convergence inefficiency (slower convergence) in cases where highly stretched meshes are used. Application of multigrid to the RANS equations can also result in convergence rates that are far from optimal due to the stiff source term associated with the turbulence models. Classic multigrid remedies for these multigrid difficulties, such as directional-coarsening, directional implicit smoother, combining directional coarsening and smoothing, and combining a point-implicit block-Jacobi preconditioner and J-coarsening, etc., are well documented and the effects are illustrated for some example problems by Pierce and Giles [145, 146] and Mavriplis [142]. Zhang and Liu suggested freezing the nonlinear terms to preserve the turbulence quantities on the coarser levels, and furthermore, limiting the change of turbulence quantities at every iteration to preserve the positivity of turbulence quantities [147]. Park and Kwow adopted most of the suggestions of Zhang and Liu, but proposed additionally the use of different CFL numbers according to grid levels [148]. Gerlinger and Brüggemann investigated the q-ω model and proposed the techniques of computing the production term and the divergence of velocity field only on the finest mesh and restricted these values to coarser meshes. For complicated geometries and simple flow field initializations, they initiated the calculation with several fine mesh iterations before restricting to coarser meshes [149].

To remedy the multigrid difficulties in stability and convergence due to the stiff turbulence production terms and chemical source terms and the use of highly stretched meshes, a point-implicit block-Jacobi preconditioner (matrix preconditioner) is employed herein in combination with the multigrid solver. The turbulence quantities are restricted to the coarse mesh but not updated so as to enhance the stability of the scheme and avoid non-physical solutions. Note that we do not believe that the point-Jacobi preconditioning will provide a fully satisfactory solution to issues with high-aspect-ratio meshes; however, our experiences show that the preconditioning combined with modifications to the restriction and prolongation operators partially alleviates the problem.

The point-implicit block-Jacobi preconditioner is based on the form of the discrete

residual operator, $\mathbf{R}$, and obtained by extracting the terms corresponding to the center cell in the stencil. The application of the matrix preconditioner to the multi-stage time-marching scheme of Equation (3.39) is rewritten as

$$\text{m stage:} \begin{cases} \mathbf{U}^{0}_{\mathrm{i,j}} = \mathbf{U}^{\mathrm{n}}_{\mathrm{i,j}} \\ \mathbf{U}^{\mathrm{m}}_{\mathrm{i,j}} = \mathbf{U}^{0}_{\mathrm{i,j}} - \nu\alpha_{\mathrm{m}}\ \mathbf{P}^{-1}_{\mathrm{i,j}}\ \mathbf{R}_{\mathrm{i,j}}\ (\mathbf{U}^{\mathrm{m-1}}) \ \mathrm{m} = 1 \cdots \mathrm{M} \\ \mathbf{U}^{\mathrm{n+1}}_{\mathrm{i,j}} = \mathbf{U}^{\mathrm{M}}_{\mathrm{i,j}} \end{cases}$$

where ν includes the time step and $\mathbf{P}^{-1}_{\mathrm{i,j}}$ is the inverse of the matrix preconditioner for cell (i,j). The construction of the matrix preconditioner is illustrated here using the system of governing equations in a two-dimensional axisymmetric system. Given the residual function for cell (i,j), $\mathbf{R}_{\mathrm{i,j}}$, which can be written as

$$\mathbf{R}_{\mathrm{i,j}} = \frac{d\mathbf{U}_{\mathrm{i,j}}}{dt} = -\frac{1}{A_{\mathrm{i,j}}} \sum_{k=1}^{N_{\mathrm{f}}} \vec{\mathbf{F}}_{\mathrm{k}} \cdot \vec{n}_{\mathrm{k}}\ \Delta l_{\mathrm{k}} + \frac{\mathcal{S}_{\mathrm{a\,i,j}}}{r} + \mathbf{S}_{\mathrm{t\,i,j}} + \mathbf{S}_{\mathrm{c\,i,j}}\ , \tag{3.44}$$

the matrix preconditioner, $\mathbf{P}_{\mathrm{i,j}} = \frac{\partial \mathbf{R}}{\partial \mathbf{U}}\big|_{\mathrm{i,j}}$, is a N×N matrix and has the form

$$\mathbf{P}_{\mathrm{i,j}} = \frac{\partial \mathbf{R}_{\mathrm{i,j}}\big(\mathbf{U}, \nabla \mathbf{U}\big)}{\partial \mathbf{U}_{\mathrm{i,j}}} = \begin{bmatrix} \frac{\partial \mathrm{R}_1}{\partial \mathrm{U}_1} & \frac{\partial \mathrm{R}_1}{\partial \mathrm{U}_2} & \cdots & \frac{\partial \mathrm{R}_1}{\partial \mathrm{U}_{\mathrm{N}}} \\ \frac{\partial \mathrm{R}_2}{\partial \mathrm{U}_1} & \frac{\partial \mathrm{R}_2}{\partial \mathrm{U}_2} & \cdots & \frac{\partial \mathrm{R}_2}{\partial \mathrm{U}_{\mathrm{N}}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \mathrm{R}_{\mathrm{N}}}{\partial \mathrm{U}_1} & \frac{\partial \mathrm{R}_{\mathrm{N}}}{\partial \mathrm{U}_2} & \cdots & \frac{\partial \mathrm{R}_{\mathrm{N}}}{\partial \mathrm{U}_{\mathrm{N}}} \end{bmatrix} . \tag{3.45}$$

Each term $\mathbf{P}_{\mathrm{i,j}}$ consists of five components and they result from the inviscid numerical flux Jacobian, $\mathbf{C}_{\mathrm{i}}$, the viscous flux Jacobian, $\mathbf{C}_{\mathrm{v}}$, and the source Jacobians due to axisymmetric coordinate system, $\mathbf{C}_{\mathrm{a}}$, the source Jacobian due to turbulence, $\mathbf{C}_{\mathrm{t}}$, and the source Jacobian due to chemistry, $\mathbf{C}_{\mathrm{c}}$. The evaluation of each of these Jacobian matrices is discussed below.

The inviscid numerical flux Jacobian at each cell face is evaluated by the solutions of a Riemann problem in a rotated frame aligned with the normal to the cell face and takes on the form of

$$\mathbf{C}_{\mathrm{i}} = \frac{\partial(\vec{\mathbf{F}} \cdot \vec{n})}{\partial \mathbf{U}_{\mathrm{i,j}}} = \frac{\partial \mathbf{F}}{\partial \mathbf{F}^{*}} \frac{\partial \mathbf{F}^{*}}{\partial \mathbf{U}^{*}_{\mathrm{L/R}}} \frac{\partial \mathbf{U}^{*}_{\mathrm{L/R}}}{\partial \mathbf{U}_{\mathrm{i,j}}}\,, \tag{3.46}$$

where $\mathbf{U}^{*}_{\mathrm{L/R}}$ and $\mathbf{F}^{*}$ are the solution state and flux in the rotated frame. The term $\frac{\partial \mathbf{F}^{*}}{\partial \mathbf{U}^{*}_{\mathrm{L/R}}}$ is evaluated for both the Roe and HLLE flux functions. This inviscid Jacobian evaluation is approximated by assuming that the eigenvalues and eigenvectors

(appearing in Roe and HLLE flux functions) are constant and, when using higher order scheme, the gradients of the primitive variables are also assumed to be constant. Pierce and Giles [146] also suggested that using the matrix precondition based on a first-order discretization for higher-order schemes is acceptable.

The viscous numerical flux Jacobian is formulated depending on the method used to evaluate the viscous flux. For two-dimensional flows, the viscous numerical flux Jacobian is determined using a diamond-path procedure. The general form for the viscous flux Jacobian is

$$\mathbf{C}_{\mathrm{v}} = \frac{\partial \vec{\mathbf{F}}_{\mathrm{v}} \cdot \vec{n}}{\partial \mathbf{U}_{\mathrm{i,j}}} = \frac{\partial \Big(n_{\mathrm{r}}\ \mathbf{F}_{\mathrm{v}} + n_{\mathrm{z}}\ \mathbf{G}_{\mathrm{v}} \Big)}{\partial \mathbf{U}_{\mathrm{i,j}}} \,. \tag{3.47}$$

In the source Jacobians, $\mathbf{C}_{\mathrm{s}}$, the three terms are lumped together as

$$\mathbf{C}_{\mathrm{s}} = -\frac{\partial (\frac{\mathbf{S}_{\mathrm{a\,i,j}}}{r})}{\partial \mathbf{U}_{\mathrm{i,j}}} + \frac{\partial \mathbf{S}_{\mathrm{t\,i,j}}}{\partial \mathbf{U}_{\mathrm{i,j}}} + \frac{\partial \mathbf{S}_{\mathrm{c\,i,j}}}{\partial \mathbf{U}_{\mathrm{i,j}}} \,. \tag{3.48}$$

The detailed derivation of this matrix preconditioner was performed with the aid of Maple [150].

Some testing was performed to ensure the correct implementation of the preconditioner in the numerical algorithm. The preconditioner was tested using a finite-difference method with a first order accurate approximation of the residual Jacobian:

$$\left. \frac{\partial \mathbf{R}_{\mathrm{i,j}}\big(\mathbf{U}, \nabla \mathbf{U}\big)}{\partial \mathbf{U}} \right|_{\mathrm{i,j}} \approx \frac{\mathbf{R}(\mathbf{U}_{\mathrm{i,j}} + \epsilon) - \mathbf{R}(\mathbf{U}_{\mathrm{i,j}})}{\epsilon} + \mathcal{O}(\epsilon) \,,$$

where $\epsilon = \eta \mathbf{U}$ with $\eta = 10^{-6} \sim 10^{-5}$. The verification has been performed with varied solution fields and different mesh stretching factors. The observed maximum relative error was found to be about

$$\sigma = \left| \frac{\frac{\partial \mathbf{R}}{\partial \mathbf{U}_{\mathrm{i,j}}} - \frac{\mathbf{R}(\mathbf{U}_{\mathrm{i,j}} + \epsilon) - \mathbf{R}(\mathbf{U}_{\mathrm{i,j}})}{\epsilon}}{\frac{\partial \mathbf{R}}{\partial \mathbf{U}_{\mathrm{i,j}}}} \right| \leq 2.0\% \,. \tag{3.49}$$

It is felt that this error is acceptable considering the approximations used in the Jacobian evaluation and the numerical error associated with the numerical scheme. It provides some level of confidence that the matrix preconditioner in the current numerical algorithm has been correctly implemented.

3.3.2 Restriction and Prolongation Operators

The use of multigrid acceleration for reactive flow calculations was not adequately examined until the early 1990s. Previous efforts related to the application of the multigrid method for combustion calculations achieved no or only small convergence acceleration [149,151]. This might be due to the fact that the convergence acceleration of multigrid techniques is achieved by damping the low frequencies of the error more efficiently on coarser-grid levels which is in contrast to the local behavior of turbulence and chemical production terms. Slomski and Radespiel simulated reacting flows with relatively small reaction schemes using standard multigrid procedures [152]. Edwards [153] pointed out that Slomski and Radespiel's approach was the only multigrid-based algorithm for computing hypersonic chemically reacting flows up until 1996. Edwards subsequently employed multigrid techniques for hypersonic chemically reacting flows and hydrogen combustion and achieved reasonable speedup [153]. Sheffer *et al.* obtained considerable speedups for detonation waves using a two-level multigrid method [154]. Gerlinger *et al.* later employed a four-level multigrid method and achieved considerable convergence speedups. Bellucci and Bruno employed a four-level multigrid method for three-dimensional incompressible flow with combustion; however, they presented convergence rates only for non-reacting cases [155].

Based on some of the ideas from this and other previous work, the following strategy was employed in this thesis work in order to produce a robust multigrid algorithm for applications to turbulent combusting flows:

- First, the turbulence quantities are not updated on coarser meshes. The turbulence quantity, ω, is explicitly involved in the combustion modelling, so this technique also helps to preserve the flame shape and location computed from the finest mesh without any errors associated with changing grids.

- The meshes used in this research are often highly stretched in order to resolve laminar sublayers, shear layers, and thin flame fronts. This results in some computational cells with very large cell aspect ratios, leading to additional difficulties for solution by the multigrid method. In particular, it was

found that the multigrid convergence was significantly affected by the prolongation operator. A standard bi-linear interpolation was used initially to transfer corrections from coarse to fine mesh; however, the performance of the multigrid scheme suffered. Convergence was hampered and, in some cases, the procedure failed to converge. In this work, efforts have been made to devise more effective prolongation operators for meshes with large cell aspect ratios. Figure 3.5 illustrates a stretched grid. Several prolongation operators are investigated: simple injection, $\mathbf{U}_{\mathrm{i,j\ fine}} = \mathbf{U}_{\mathrm{i,j\ coarse}}$; area weighted injection, $\mathbf{U}_{\mathrm{i,j\ fine}} = (A_{\mathrm{i,j\ coarse}}/\sum A_{\mathrm{i,j\ fine}})\ \mathbf{U}_{\mathrm{i,j\ coarse}}$; a standard bi-linear interpolation; and finally, a standard bi-linear interpolation plus a linear interpolation, meaning that interpolating the value for the coarse node (i, j) using standard bi-linear first and then interpolating for the fine cell (i, j) with values from both coarse node (i, j) and coarse cell (i, j) with a linear interpolation. The effectiveness of using these different prolongation operators will be demonstrated for a laminar non-reacting flow problem using stretched computational grids in Chapter 5.

Note that the multigrid algorithm is applied directly to each of the solution blocks without regard to the level of refinement for the grid blocks associated with each solution block, i.e., the grids are not necessarily on the same refinement level due to the AMR procedure. This block-based approach to the multigrid algorithm can adversely affect its performance. This performance degradation may be remedied by utilizing additional multigrid levels spanning across blocks to bring all solution content to the same level of refinement, but this is not considered here.

3.3.3 Acceleration Efficiency

The "explicit" nature of the proposed point-implicit smoother allows the multigrid algorithm to be implemented in a fairly straightforward manner on a parallel computational platform. The use of the preconditioned multigrid acceleration technique enables efficient solutions of the turbulent reactive flow calculations. The study of the preconditioned-multigrid convergence acceleration with the proposed enhancement strategies was carried out for several two-dimensional flow problems: fully-developed

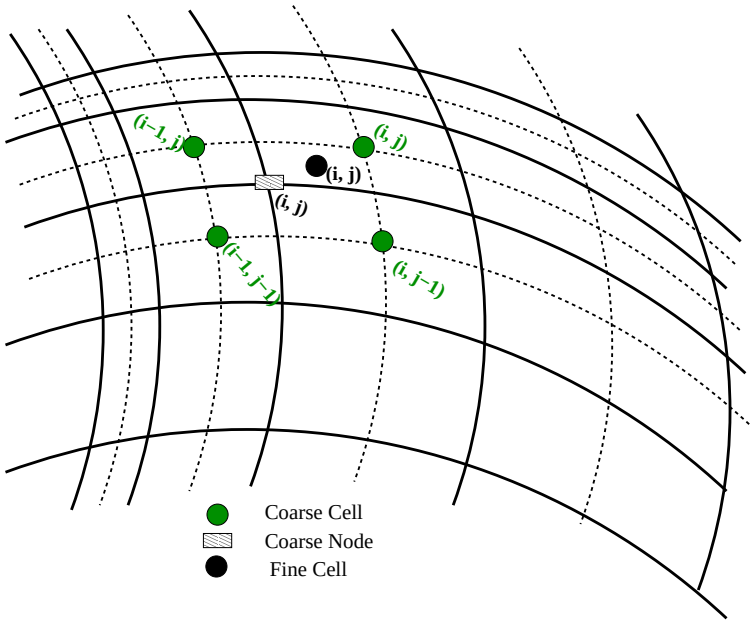


Figure 3.5: Illustration of different prolongation operators on a stretched body-fitted quadrilateral mesh.

turbulent pipe flow and turbulent non-reacting and reacting flows about a bluff-body burner. The numerical results and the computational performance for these flow problems are described in Chapters 5 and 6.

Chapter 4

Parallel Adaptive Mesh Refinement of Multi-Block Body-Fitted Meshes

Adaptive mesh refinement algorithms adapt the mesh automatically to the solution of the governing equations and can be very effective in treating problems with disparate length scales. This chapter outlines the main features of the proposed parallel AMR algorithm and its parallel implementation. Section 4.2 provides details of the proposed grid refinement procedure and discusses the design of appropriate refinement criteria. A flexible block-based hierarchical tree data structure has been developed and is used to maintain the connectivity of the solution blocks in the multi-block mesh and this is described in Section 4.3. The computation of solution block connectivity is outlined in detail, including the treatment for unstructured block connectivity of the root blocks. This chapter concludes with a discussion of the parallel implementation, including load balancing and Morton ordering procedures.

4.1 Overview of Parallel AMR Scheme

Adaptive mesh refinement algorithms permit local mesh refinement and thereby minimize the number of computational cells required for a particular calculation. As

mentioned in the first chapter, a block-based AMR approach is employed herein. The proposed AMR formulation borrows from previous work by Berger and co-workers [13,14,16,18–20,24] for Cartesian meshes and has similarities with the block-based approaches described by Quirk [19] and Berger [20]. Other researchers have considered the extension of Cartesian mesh adaptation procedures to more arbitrary quadrilateral and hexagonal meshes [156, 157]. In this thesis work, the multi-block body-fitted AMR scheme allows for the use of anisotropic meshes for resolving thin shear and boundary layers. The proposed refinement procedure preserves the original stretching of the mesh. Following the approach developed by Groth [30, 31] for computational magnetohydrodynamics, a flexible block-based hierarchical data structure has been developed. This data structure facilitates automatic solution-directed mesh adaptation on multi-block meshes according to physics-based refinement criteria.

The implementation of the AMR procedure in the proposed algorithm involves the following steps:

1. evaluation of the refinement measures for each solution block and marking of solution blocks for refinement and coarsening;

2. assessment of the refinement levels for all solution blocks to ensure that the refinement ratio between adjacent blocks is no greater than 1:2;

3. removal of solution blocks associated with coarsening of grid;

4. addition of "leaves" representing new children solution blocks in the tree data structure;

5. update of block connectivity and block information used in sharing solution data between blocks;

6. carry out actual coarsening and refinement of blocks marked for a resolution change with a redistribution of the children solution blocks among the processors to ensure load balancing.

The remainder of this chapter is devoted to discussions of various aspects of the parallel AMR scheme.

4.2 Refinement and Coarsening of Solution Blocks

The finite-volume scheme described in the previous chapter (Chapter 3) is applied to multi-block body-fitted mesh in which the grid is composed of a number of structured blocks. Each of these structured blocks of the computational mesh consists of $N_i \times N_j$ quadrilateral cells in two space dimensions and $N_i \times N_j \times N_k$ hexahedral cells in the three-dimensional case, where N_i, N_j and N_k are even integers greater than or equal to four. The values of N_i, N_j and N_k are not necessarily the same for each block; however, the use of unstructured root block connectivity (described later in this chapter) imposes some constraints on the relationships between N_i, N_j and N_k for each of the grid blocks. Furthermore, having the same number of computational cells in each grid block greatly facilitates the parallel implementation of the algorithm as shall be seen.

Mesh adaptation is accomplished by dividing and coarsening of appropriate grid blocks. In regions requiring increased cell resolution, a "parent" block is refined by dividing itself into four or eight "children" or "offspring" depending on the dimensionality. Each of the four or eight children of a parent block becomes a new block having the same number of cells as the parent and thereby doubling the cell resolution in the region of interest. This refinement process can be reversed in regions that are deemed over-resolved and four or eight children are coarsened or merged into a single parent block. Figure 4.1 illustrates two neighbouring $8 \times 8 \times 8$ hexahedral blocks of a three-dimensional mesh, one of which has undergone one level of refinement and one of which has not. The resulting refined grid consists of nine blocks. The refined grid can be coarsened or de-refined by reversing the division process and merging eight blocks into one.

The grid adaption is constrained such that the grid resolution changes by only a factor of two between adjacent blocks and the minimum resolution is not less than that of the initial mesh. Once the grid is refined, standard multigrid-type restriction and prolongation operators are used to evaluate the solution on all blocks created by the coarsening and division processes, respectively.

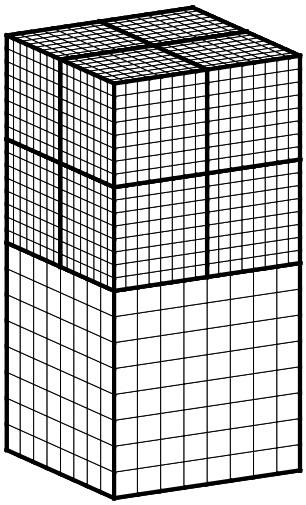


Figure 4.1: An example of two neighbouring $8 \times 8 \times 8$ hexahedral solution blocks: one which has undergone refinement and one which has not.

4.2.1 Refined Grid Generation

A key operation in the proposed block-based AMR procedure is the generation of the mesh points in the refined grid blocks from the initial mesh. This section describes the technique used to refine a given block in the mesh using the grid metrics. The resulting procedure is very effective in preserving the original mesh point clustering in the body-fitted mesh and maintaining the smoothness and locations of the grid lines in the mesh.

In the proposed grid refinement procedure, it is assumed that the nodal locations in the physical domain, $\vec{x} = (\mathrm{x}, \mathrm{y}, \mathrm{z})$, can be mapped onto a uniformly spaced Cartesian computational domain, (ξ, η, ζ). The physical location of the additional nodes in the refined grid can then be computed using Taylor approximation theory and the grid metrics. Specifically, the refined nodes along the edges, on the faces, and along the center of the volume are calculated using a Taylor series expansion truncated to

second-order for the node locations in physical space in terms of the coordinates, ξ, η, ζ, of the computational space given by

$$\begin{aligned}\vec{x}(\xi+\Delta\xi,\eta+\Delta\eta,\zeta+\Delta\zeta) = \vec{x}(\xi,\eta,\zeta) &+ \left.\frac{\partial\vec{x}}{\partial\xi}\right|_{\xi,\eta,\zeta}\Delta\xi + \left.\frac{\partial\vec{x}}{\partial\eta}\right|_{\xi,\eta,\zeta}\Delta\eta + \left.\frac{\partial\vec{x}}{\partial\zeta}\right|_{\xi,\eta,\zeta}\Delta\zeta \\ &+\frac{1}{2}\left(\left.\frac{\partial^2\vec{x}}{\partial\xi^2}\right|_{\xi,\eta,\zeta}(\Delta\xi)^2 + 2\left.\frac{\partial^2\vec{x}}{\partial\xi\partial\eta}\right|_{\xi,\eta,\zeta}\Delta\xi\Delta\eta + \left.\frac{\partial^2\vec{x}}{\partial\eta^2}\right|_{\xi,\eta,\zeta}(\Delta\eta)^2\right. \\ &\left.+2\left.\frac{\partial^2\vec{x}}{\partial\xi\partial\zeta}\right|_{\xi,\eta,\zeta}\Delta\xi\Delta\zeta + \left.\frac{\partial^2\vec{x}}{\partial\zeta^2}\right|_{\xi,\eta,\zeta}(\Delta\zeta)^2 + 2\left.\frac{\partial^2\vec{x}}{\partial\eta\partial\zeta}\right|_{\xi,\eta,\zeta}\Delta\eta\Delta\zeta\right) \\ &+\mathcal{O}\Big((\Delta\xi)^3,(\Delta\eta)^3,(\Delta\zeta)^3\Big). \end{aligned} \tag{4.1}$$

Approximate expressions for the various derivatives appearing in Equation (4.1) are required. The first derivatives

$$\frac{\partial\vec{x}}{\partial\xi}, \quad \frac{\partial\vec{x}}{\partial\eta}, \quad \frac{\partial\vec{x}}{\partial\zeta},$$

are estimated using second-order-accurate forward/backward finite-difference formulae for boundary nodes and second-order-accurate centre difference for interior coarse nodes. As shown in Figure 4.2, the first derivatives are computed at the coarse nodes of the original mesh which are represented by solid-filled circles. The second-order derivatives,

$$\frac{\partial^2\vec{x}}{\partial\xi^2}, \quad \frac{\partial^2\vec{x}}{\partial\eta^2}, \quad \frac{\partial^2\vec{x}}{\partial\zeta^2},$$

are computed for each midpoint on the edge (labelled with non-filled circles) based on finite differences of the first derivatives. These expressions are used to approximate

the second derivatives for any vertex on the edge as follows:

$$\begin{aligned}
\left.\frac{\partial^2\vec{x}}{\partial\xi^2}\right|_{\xi,\eta,\zeta} &= \left.\frac{\partial^2\vec{x}}{\partial\xi^2}\right|_{\xi+\Delta\xi,\eta,\zeta} \approx \left.\frac{\partial^2\vec{x}}{\partial\xi^2}\right|_{\xi+\frac{\Delta\xi}{2},\eta,\zeta}, \\
\left.\frac{\partial^2\vec{x}}{\partial\eta^2}\right|_{\xi,\eta,\zeta} &= \left.\frac{\partial^2\vec{x}}{\partial\eta^2}\right|_{\xi,\eta+\Delta\eta,\zeta} \approx \left.\frac{\partial^2\vec{x}}{\partial\eta^2}\right|_{\xi,\eta+\frac{\Delta\eta}{2},\zeta}, \\
\left.\frac{\partial^2\vec{x}}{\partial\zeta^2}\right|_{\xi,\eta,\zeta} &= \left.\frac{\partial^2\vec{x}}{\partial\zeta^2}\right|_{\xi,\eta,\zeta+\Delta\zeta} \approx \left.\frac{\partial^2\vec{x}}{\partial\zeta^2}\right|_{\xi,\eta,\zeta+\frac{\Delta\zeta}{2}}.
\end{aligned} \tag{4.2}$$

Mixed derivatives,

$$\frac{\partial^2\vec{x}}{\partial\xi\partial\eta}, \quad \frac{\partial^2\vec{x}}{\partial\xi\partial\zeta}, \quad \frac{\partial^2\vec{x}}{\partial\eta\partial\zeta},$$

are computed at the face centroid (labelled with shaded square in Figure 4.2) from coarse-grid vertex information (labelled with solid-filled circles). These values would be used to approximate the mixed derivatives at the coarse nodes on the face. The mixed derivatives for the four coarse nodes on a face as shown in Figure 4.3 are approximated by

$$\left.\frac{\partial^2\vec{x}}{\partial\xi\partial\zeta}\right|_{\xi,\eta,\zeta} = \left.\frac{\partial^2\vec{x}}{\partial\xi\partial\zeta}\right|_{\xi+\Delta\xi,\eta,\zeta} = \left.\frac{\partial^2\vec{x}}{\partial\xi\partial\zeta}\right|_{\xi,\eta,\zeta+\Delta\zeta} = \left.\frac{\partial^2\vec{x}}{\partial\xi\partial\zeta}\right|_{\xi+\Delta\xi,\eta,\zeta+\Delta\zeta} \approx \left.\frac{\partial^2\vec{x}}{\partial\xi\partial\zeta}\right|_{\xi+\frac{\Delta\xi}{2},\eta,\zeta+\frac{\Delta\zeta}{2}}. \tag{4.3}$$

However, an averaging procedure used to estimate the refined nodes on each face and at volume centroid described below results in the cancellation of the mixed derivatives and, in practice, the mixed derivatives are not required nor computed. The averaging procedure and the cancellation of the mixed derivatives due to this averaging procedure are demonstrated next.

A second-order averaging procedure is used to combine the Taylor series expansions, Equation (4.1), at each of the four nodes of a given coarse face when evaluating the new mesh points of the fine grid. For a midpoint along the edge, (for example, see Figure 4.3), its physical location, $\vec{x}(\xi + \frac{\Delta\xi}{2}, \eta, \zeta)$, is determined by averaging the

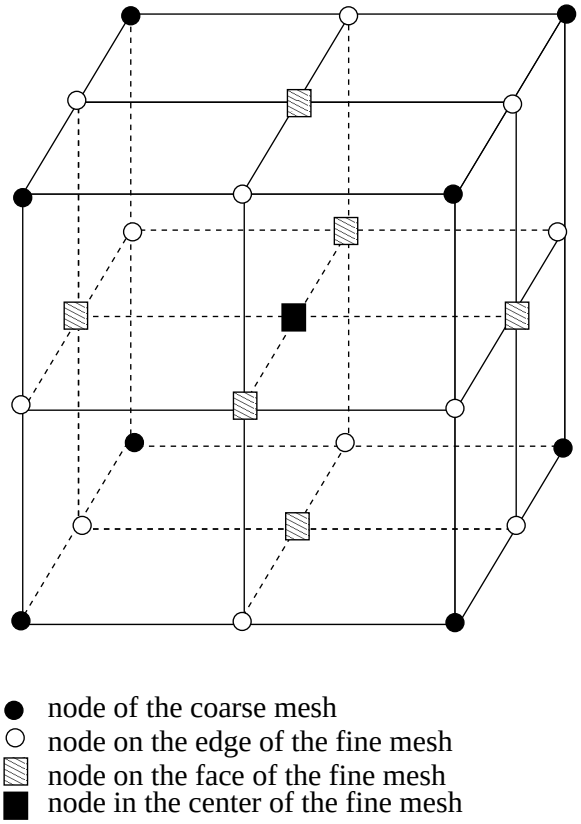


Figure 4.2: Illustration of refining a grid with grid metrics.

two approximated locations obtained using Taylor expansion (Equation (4.1)) from its left and right coarse nodes, $\vec{x}(\xi, \eta, \zeta)$, and $\vec{x}(\xi + \Delta\xi, \eta, \zeta)$, respectively. For the centroid of a face, values from four coarse nodes are averaged, and similarly, for the center of the volume, values from eight coarse nodes are averaged. Note again that the mixed second derivatives cancel out during the averaging process for faces and volumes, obviating the need to compute them. The cancellation of the mixed derivatives can be illustrated with the aid of Figure 4.3. The physical location of the node on the center of a face, $\vec{x}(\xi + \frac{\Delta\xi}{2}, \eta, \zeta + \frac{\Delta\zeta}{2})$, can be approximated by using Taylor

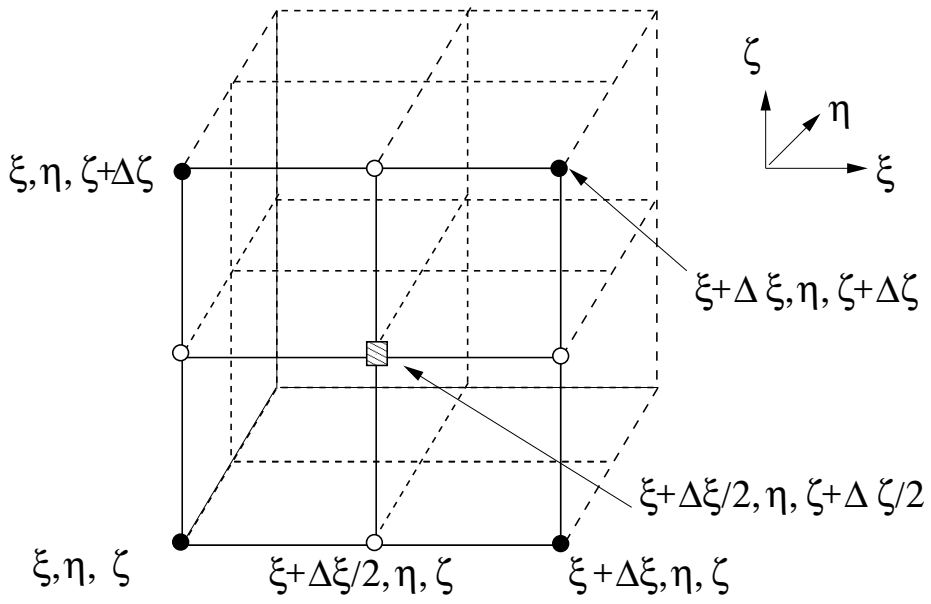


Figure 4.3: Illustration of the four coarse nodes on a face.

expansion from the node of ξ, η, ζ, and is given by

$$\vec{x}(\xi + \frac{\Delta\xi}{2}, \eta, \zeta + \frac{\Delta\zeta}{2}) = \vec{x}(\xi, \eta, \zeta) + \left.\frac{\partial \vec{x}}{\partial \xi}\right|_{\xi,\eta,\zeta} \frac{\Delta\xi}{2} + \left.\frac{\partial \vec{x}}{\partial \zeta}\right|_{\xi,\eta,\zeta} \frac{\Delta\zeta}{2} + \frac{1}{2}$$
$$\left(\left.\frac{\partial^2 \vec{x}}{\partial \xi^2}\right|_{\xi,\eta,\zeta} \frac{(\Delta\xi)^2}{4} + \left.\frac{\partial^2 \vec{x}}{\partial \xi \partial \zeta}\right|_{\xi,\eta,\zeta} \frac{\Delta\xi\Delta\zeta}{2} + \left.\frac{\partial^2 \vec{x}}{\partial \zeta^2}\right|_{\xi,\eta,\zeta} \frac{(\Delta\zeta)^2}{4} \right)$$
$$+\mathcal{O}\Big((\Delta\xi)^3, (\Delta\eta)^3, (\Delta\zeta)^3 \Big) . \tag{4.4}$$

The physical location of the node of interest here can also be approximated by using Taylor expansions from the other nodes $(\xi+\Delta\xi, \eta, \zeta)$, $(\xi, \eta, \zeta+\Delta\zeta)$, and $\vec{x}(\xi+\Delta\xi, \eta, \zeta+\Delta\zeta)$:

$$\vec{x}(\xi + \frac{\Delta\xi}{2}, \eta, \zeta + \frac{\Delta\zeta}{2}) = \vec{x}(\xi + \Delta\xi, \eta, \zeta) + \left.\frac{\partial \vec{x}}{\partial \xi}\right|_{\xi+\Delta\xi,\eta,\zeta} \frac{-\Delta\xi}{2} + \left.\frac{\partial \vec{x}}{\partial \zeta}\right|_{\xi+\Delta\xi,\eta,\zeta} \frac{\Delta\zeta}{2} +$$
$$\frac{1}{2}\left(\left.\frac{\partial^2 \vec{x}}{\partial \xi^2}\right|_{\xi+\Delta\xi,\eta,\zeta} \frac{(-\Delta\xi)^2}{4} + \left.\frac{\partial^2 \vec{x}}{\partial \xi \partial \zeta}\right|_{\xi+\Delta\xi,\eta,\zeta} \frac{-\Delta\xi\Delta\zeta}{2} + \left.\frac{\partial^2 \vec{x}}{\partial \zeta^2}\right|_{\xi+\Delta\xi,\eta,\zeta} \frac{(\Delta\zeta)^2}{4} \right)$$
$$+\mathcal{O}\Big((\Delta\xi)^3, (\Delta\eta)^3, (\Delta\zeta)^3 \Big) , \tag{4.5}$$

$$\vec{x}(\xi+\frac{\Delta\xi}{2},\eta,\zeta+\frac{\Delta\zeta}{2}) = \vec{x}(\xi,\eta,\zeta+\Delta\zeta) + \left.\frac{\partial\vec{x}}{\partial\xi}\right|_{\xi,\eta,\zeta+\Delta\zeta}\frac{\Delta\xi}{2} + \left.\frac{\partial\vec{x}}{\partial\zeta}\right|_{\xi,\eta,\zeta+\Delta\zeta}\frac{-\Delta\zeta}{2} + \frac{1}{2}\left(\left.\frac{\partial^2\vec{x}}{\partial\xi^2}\right|_{\xi,\eta,\zeta+\Delta\zeta}\frac{(\Delta\xi)^2}{4} + \left.\frac{\partial^2\vec{x}}{\partial\xi\partial\zeta}\right|_{\xi,\eta,\zeta+\Delta\zeta}\frac{-\Delta\xi\Delta\zeta}{2} + \left.\frac{\partial^2\vec{x}}{\partial\zeta^2}\right|_{\xi,\eta,\zeta+\Delta\zeta}\frac{(-\Delta\zeta)^2}{4}\right) + \mathcal{O}\Big((\Delta\xi)^3,(\Delta\eta)^3,(\Delta\zeta)^3\Big), \tag{4.6}$$

$$\vec{x}(\xi+\frac{\Delta\xi}{2},\eta,\zeta+\frac{\Delta\zeta}{2}) = \vec{x}(\xi+\Delta\xi,\eta,\zeta+\Delta\zeta) + \left.\frac{\partial\vec{x}}{\partial\xi}\right|_{\xi+\Delta\xi,\eta,\zeta+\Delta\zeta}\frac{-\Delta\xi}{2} + \left.\frac{\partial\vec{x}}{\partial\zeta}\right|_{\xi+\Delta\xi,\eta,\zeta+\Delta\zeta}\frac{-\Delta\zeta}{2} + \frac{1}{2}\left(\left.\frac{\partial^2\vec{x}}{\partial\xi^2}\right|_{\xi+\Delta\xi,\eta,\zeta+\Delta\zeta}\frac{(-\Delta\xi)^2}{4} + \left.\frac{\partial^2\vec{x}}{\partial\xi\partial\zeta}\right|_{\xi+\Delta\xi,\eta,\zeta+\Delta\zeta}\frac{\Delta\xi\Delta\zeta}{2} + \left.\frac{\partial^2\vec{x}}{\partial\zeta^2}\right|_{\xi+\Delta\xi,\eta,\zeta+\Delta\zeta}\frac{(-\Delta\zeta)^2}{4}\right) + \mathcal{O}\Big((\Delta\xi)^3,(\Delta\eta)^3,(\Delta\zeta)^3\Big). \tag{4.7}$$

The physical location of the node on the center of a face, $\vec{x}(\xi+\frac{\Delta\xi}{2},\eta,\zeta+\frac{\Delta\zeta}{2})$, is then determined by averaging the four approximations above. Clearly, using Equation (4.3), the terms, involving the mixed derivatives all cancel out.

The performance of the proposed mesh refinement scheme based on the grid metrics can be assessed by comparing refined meshes generated by the second-order averaging procedure and grid refinement based on straightforward linear interpolation. Linear interpolation is performed by averaging the adjacent coarse-grid vertex coordinates. Figure 4.4 depicts a refined grid obtained via the linear interpolation procedure from a coarse curvilinear grid block. It is evident that the curvature of the grid lines and mesh point clustering are not smoothly preserved. Figure 4.5 shows a similarly refined grid generated using the second-order averaging approach described above. In this case, it is clear that the refined grid preserves the topology of the original mesh.

Further evidence of the capabilities of the refined mesh generation technique proposed here based on the grid metrics is provided by considering the refinement of the multi-block grid for a pipe or duct with a circular cross section as shown in Figures 4.6 and 4.7. In the figures, it can be seen that the refined mesh not only preserves the original stretching, but quite accurately reproduce the curved arc of the boundary.

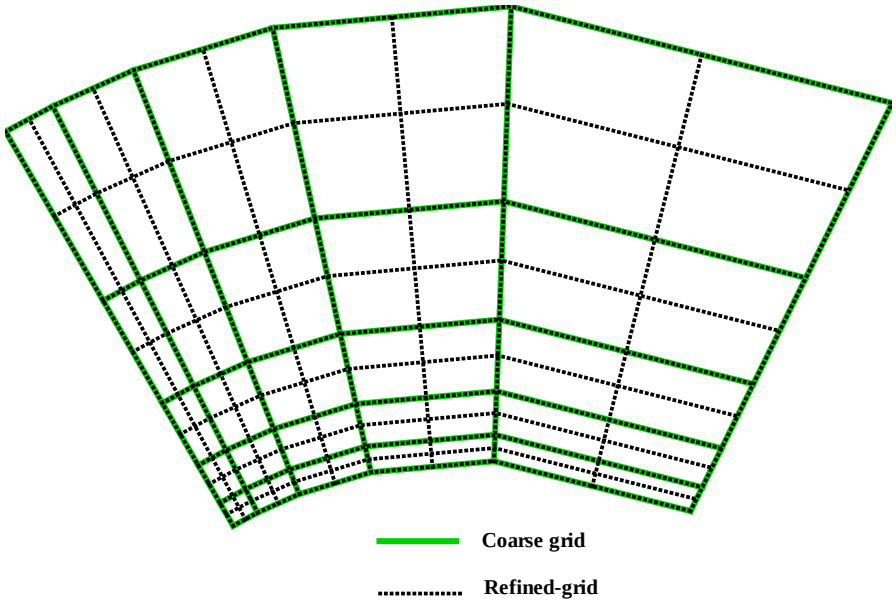


Figure 4.4: A refined grid using linear interpolation approach.

Provided that the boundary can be represented by a continuous surface, this approach helps to avoid the need for projecting the locations of the refined-mesh boundary nodes exactly onto the physical geometry and preserves the smoothness of the interior grid lines. For the somewhat simple flow geometries considered herein, algebraic relations are used to ensure boundary nodes conform to the physical boundaries. In the more general case, special treatment must be developed for dealing with the physical boundary geometry (i.e., projecting the refined-mesh boundary grid points exactly onto the physical geometry), particularly when encountering some highly (or pathologically) curved physical boundaries with meshes having high aspect ratios.

Each physical block can be refined in isolation. By using one-sided differences near the boundary, the proposed refinement procedure based on the metrics does not require ghost node information from adjacent grid blocks during the refining process. The ghost nodes (details of the ghost-cell structure are given in section 4.4.1) are

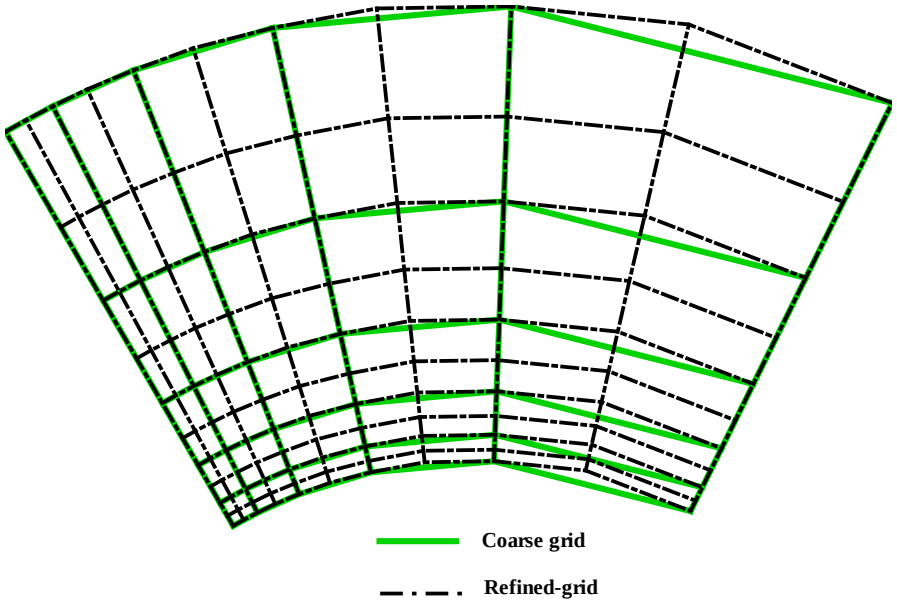


Figure 4.5: A refined grid using second-order averaging approach.

grouped into sections corresponding to the physical blocks to which they belong. Each of these sections is refined in isolation exactly as the rest of the physical block (to which they belong) would be. As Figure 4.8 illustrates, since two layers of refined ghost cells collapse into one layer of coarse ghost cells, there is sufficient information to compute the metrics exactly as would be done for the complete physical block. Hence, there is no need for an exchange of grid geometry information between blocks at the boundary during refinement.

4.2.2 Coarse Grid Generation

Coarsening of the computational mesh can be accomplished in a straightforward manner by simply reversing the refinement procedure. This is accomplished by the elimination of mesh points, and thereby reverting the fine mesh to its original unre-

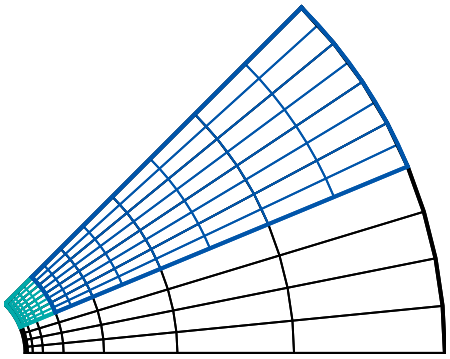


Figure 4.6: A refined segment of a pipe grid geometry using second-order averaging approach.

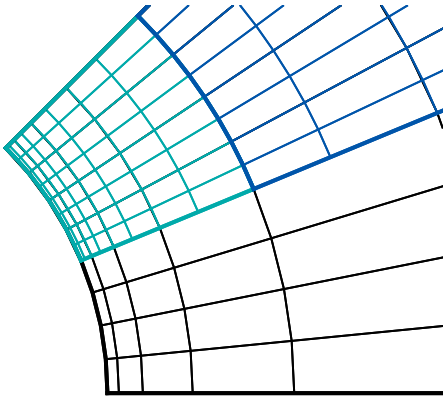


Figure 4.7: A close-up view of the refined grid showing that the refined grid maintains the original stretching.

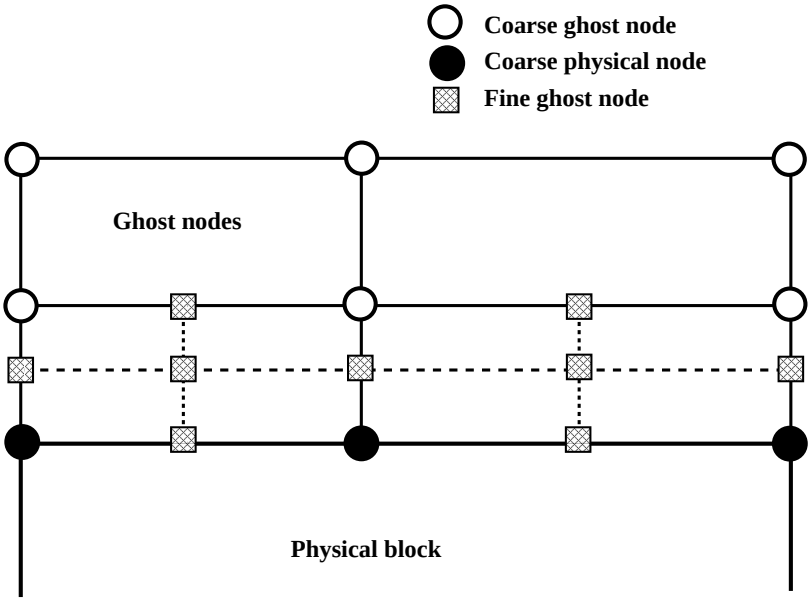


Figure 4.8: Illustration of generation of two layers of fine ghost nodes using the grid metrics from two layers of coarse nodes.

fined mesh. The coarsened mesh will retain only every second node of the fine mesh. Accordingly, four solution blocks are merged into one solution block for two space

dimensions and eight solution blocks are merged into one solution block for three space dimensions.

4.2.3 Refinement Criteria

Solution-directed mesh-refinement requires criteria for deciding where to refine or coarsen the mesh. A key issue is to reliably determine whether a solution is acceptable and ensure that the error concerning physical quantities of interest has been brought below a prescribed tolerance. If this is not the case, refinement criteria can be used to generate a new computational mesh on which a more accurate numerical solution can be obtained. In this study, a heuristic set of refinement criteria based on our physical understanding of the flow properties of interest is used (so-called physics-based refinement criteria). For the non-reacting flows considered here, the following measures are used

$$\epsilon_1 \propto |\vec{\nabla}\rho| \qquad \epsilon_2 \propto |\vec{\nabla}\cdot\vec{u}| \qquad \epsilon_3 \propto |\vec{\nabla}\otimes\vec{u}|\,, \tag{4.8}$$

in the decision to refine or coarsen a solution block. These three quantities correspond to local measures of the density gradient, compressibility, and vorticity of the mean flow field and enable the detection of contact surfaces, shocks, and shear layers. For combusting flows, additional measures were identified for directing the mesh adaption. The following four additional measures, ϵ_4, ϵ_5, ϵ_6, and ϵ_7 are used and given by

$$\epsilon_4 \propto |\vec{\nabla}k| \qquad \epsilon_5 \propto |\vec{\nabla}\omega| \qquad \epsilon_6 \propto |\vec{\nabla}T| \qquad \epsilon_7 \propto |\vec{\nabla}c_{\mathrm{n}}|\,. \tag{4.9}$$

The first two measures correspond to gradients of the specific turbulent kinetic energy and dissipation rate per unit turbulent kinetic energy, respectively, and relate to the structure of the turbulent field. The last two quantities measure the gradients of mean temperature and mean concentration for species n, respectively, and provide reliable detection of flame fronts and combustion zones for reactive flows. In addition, for the resolution of turbulent wall boundary layers, the quantity, y^+, a dimensionless distance from the wall surface, can also be used as a measure to direct the refinement. A smaller y^+ indicates that the location is closer to the wall surface. Given a threshold

for y^+ based on the flow property and geometry characteristics, together with other refinement measures above, one can expect the AMR procedure to refine the mesh so as to resolve wall boundary layers.

Using these measures, the decision for refinement of a given solution block is determined according to the following procedure:

- calculate the refinement measures (for example, ϵ_1 is normalized by $\sqrt[3]{V}/\rho$ and ϵ_2 and ϵ_3 are normalized by $\sqrt[3]{V}/a$ where a is the sound speed) for each cell and assign the maximum value for all cells as the refinement measures for the solution block;

- determine the global minimal and maximal values of the refinement criteria for all solution blocks.

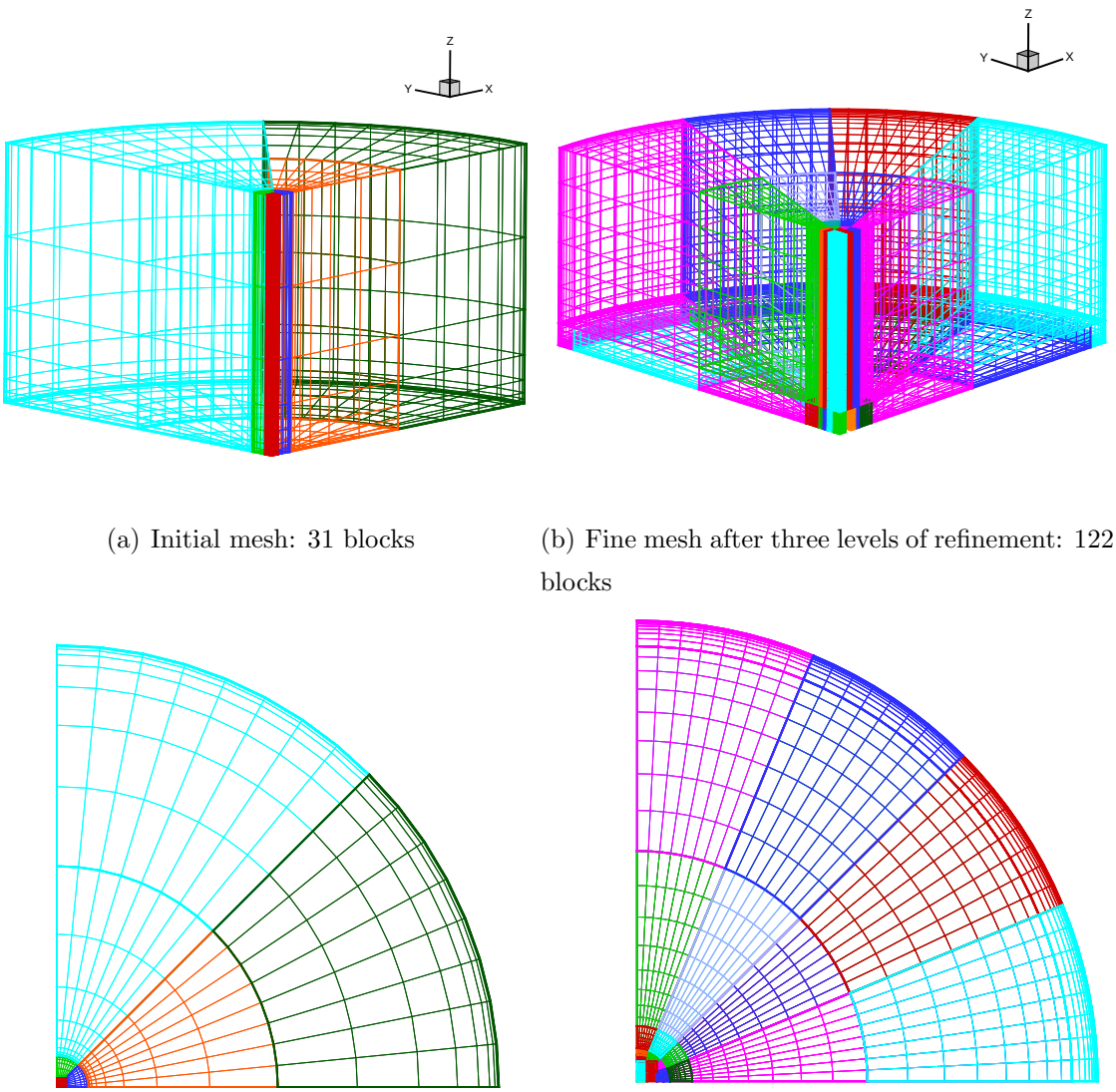
- mark solution blocks to be refined/coarsened after comparing their refinement measures to the refinement/coarsening thresholds scaled by the global extrema, i.e., blocks with refinement measures below some specified minimum measure are coarsened and blocks with measures above some upper bound are refined.

For example, let the $\epsilon_{3\min}$ and $\epsilon_{3\max}$ be the global minimal and maximal values of the refinement measure of curl of the velocity field. Then a threshold for refinement, $\epsilon_{3\mathrm{t}}$, is defined as

$$\epsilon_{3\mathrm{t}} = \epsilon_{3\min} + \alpha_{\mathrm{t}}(\epsilon_{3\max} - \epsilon_{3\min}) \,, \tag{4.10}$$

where α_{t} is an adjustable value between $0.5 \sim 0.75$. Blocks with values of the refinement measure, ϵ_3, greater than the threshold for refinement, $\epsilon_{3\mathrm{t}}$, will be refined. The decision for coarsening a block is made in a similar manner. The threshold for coarsening, α_{t}, usually is selected to be $0.1 \sim 0.2$. For those blocks with refinement measures, ϵ_3, smaller than the threshold for coarsening, $\epsilon_{3\mathrm{t}}$, a coarsening procedure is performed. Figure 4.9 illustrates the adaptation of a three-dimensional multi-block hexahedral mesh using the above refinement criteria.

It is recognized that the current set of refinement measures is by no means optimal, but experience has shown that it generally works well for the flow problems considered in this study. One deficiency of the proposed set of refinement measures



(a) Initial mesh: 31 blocks

(b) Fine mesh after three levels of refinement: 122 blocks

(c) xy plane for initial mesh

(d) xy plane for fine mesh

Figure 4.9: Illustration of AMR for a three-dimensional multi-block hexahedral mesh.

is that it does not provide a reliable criterion for terminating the refinement process. An alternative strategy for adaptive mesh refinement relies on equidistribution of solution error based on local estimates of the gradient and curvature of the solution [158,159]. Although not implemented herein, this methodology has been applied to steady [62–66] and unsteady [26,67] combustion simulations. Note however, in most

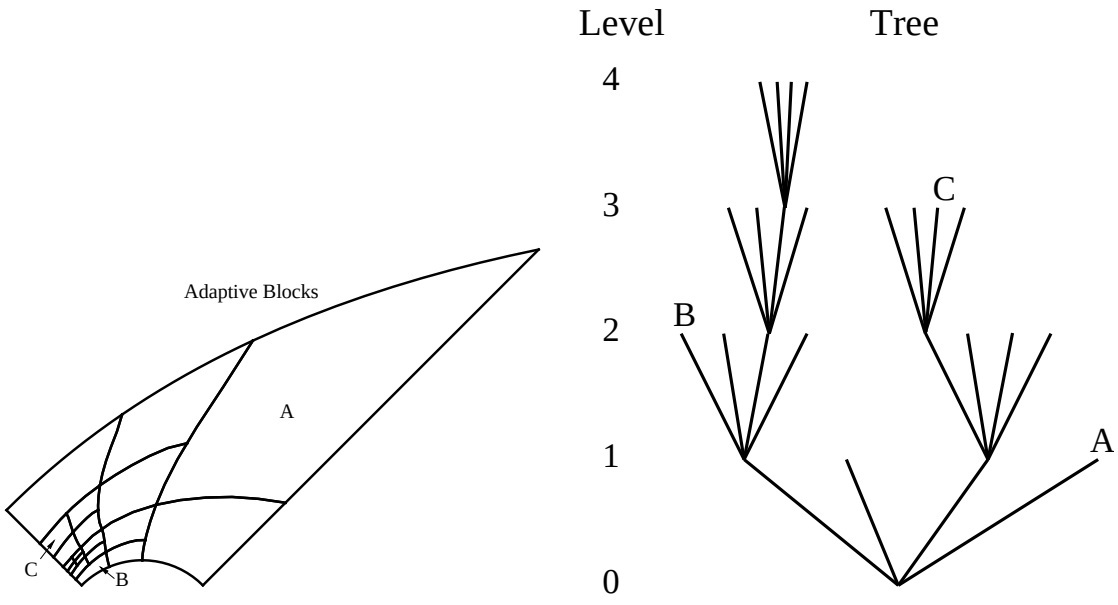
applications involving nonlinear partial differential equations, selecting the error indicators is not straightforward and sometimes, the error indicators lack theoretical justifications [160]. Chen [159] reviewed a number of strategies on mesh adaptation from a mathematical point of view and compared their performances in solving nonlinear diffusion models. Venditti and Darmofal [161–163] proposed an adjoint error estimation approach and a more conservative criterion for adaption based on residual errors that lead to improvements in the quality of the error estimates. Please consult this previous research for further details.

4.3 Solution Block Connectivity

4.3.1 Hierarchical Tree Data Structure

A flexible block-based hierarchical tree-like data structure is used herein to maintain the connectivity of the solution blocks in the multi-block mesh. In particular, quadtree and octree data structures are used for tracking the connectivity of blocks in the two- and the three-dimensional cases, respectively. For the two-dimensional case, Figure 4.10(a) shows multi-block quadrilateral AMR mesh solution blocks at various levels of refinement. Figure 4.10(b) illustrates the corresponding quadtree data structure used to keep track of mesh refinement and the connectivity between solution blocks. Figure 4.11 depicts a three-dimensional multi-block hexahedral AMR mesh consisting of solution blocks at various levels of refinement and the corresponding octree data structure. For the three-dimensional case, a block-based hierarchical octree data structure, as illustrated in Figure 4.11, has been developed and is used to keep track of mesh refinement and block connectivity.

The quadtree/octree data structure developed here naturally keeps track of the refinement level and connectivity between grid blocks during isotropic refinement processes. Although it is not strictly anisotropic, the refinement approach here preserves original stretching of the mesh and allows for anisotropic mesh and improved treatment of thin boundary and shear layers. Note that strictly anisotropic mesh adaption strategy has been considered by other researchers [39,164,165] and a hierar-



(a) Multi-block refinement (b) Quadtree

Figure 4.10: Multi-block quadrilateral AMR mesh showing solution blocks at various levels of refinement and the corresponding quadtree data structure.

chical binary-tree data structure [39] and/or an indexing scheme for Cartesian mesh can be used to keep track of the grid connectivity [164, 165].

4.3.2 Computation of Solution Block Connectivity

Neighbour information of each block is required in order to exchange solution and/or geometry information during the solution procedure. Obviously, adaptive mesh refinement complicates the process for determining neighbouring blocks. The technique developed herein for searching for nearest neighbours is now described. The searching algorithm is based on existing knowledge of neighbouring solution blocks that is stored in the quad/octree data structure. In other words, before any refinement process, it is assumed that each block has all of its neighbour information. This knowledge is used to understand the relative orientation between two branches of blocks, with one branch containing a *work* block of interest and the other containing the *neighbour* block of interest, and to define a so-called "bridge" between two

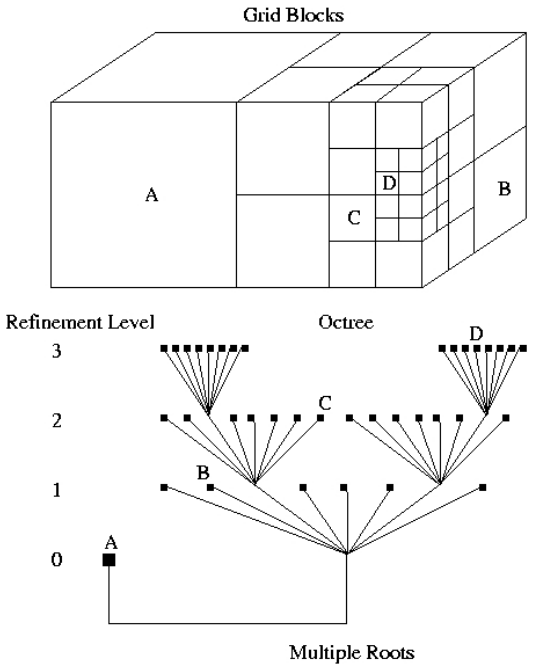


Figure 4.11: Multi-block hexahedral AMR mesh showing solution blocks at various levels of refinement and the corresponding octree data structure.

branches (a pointer that provides connection to two different branches of the tree: in this case, *work* and *neighbour* branches).

Consider execution of the AMR process at a given point during the solution procedure. The neighbour information for many of the solution blocks has been changed due to the coarsening and refinement process. The neighbouring blocks on the twenty-six boundary elements (6 faces, 12 edges and 8 vertices) of a given block, labelled the *work* block, must be found and the neighbour information, such as block number, relative refinement level, orientation, etc., need to be stored for future information exchange (message passing) between grid blocks. The number of neighbour blocks depends on the resolution change across a boundary element. If there is no resolution change, an interior corner, edge, and face element can only have one neighbouring block. If there is a resolution change (the maximum change in resolution is restricted to 2:1), each of the corner boundary elements can only have one neighbour block; oth-

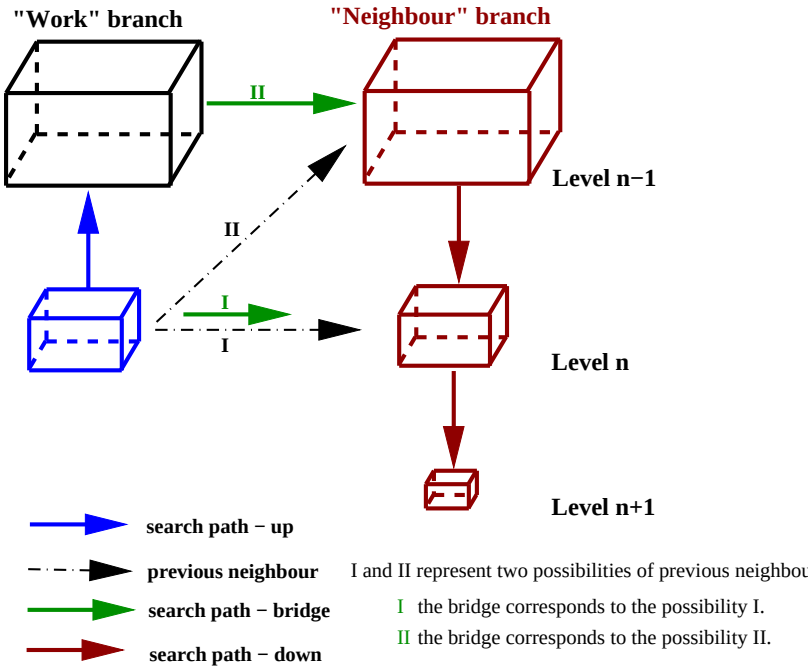


Figure 4.12: The refinement flags for the *work* block (blue) and its previous *neighbour* block (dark red) are "flagged for no change" and "flagged for refinement", respectively, and the refinement level for the previous *neighbour* can be either level n-1 or level n.

erwise there are 2 and 4 neighbouring blocks for the edge and face boundary elements, respectively.

The status of refinement/coarsening flags of both the *work* block and its *neighbour* block are first checked. Nine possible combinations of the relative refinement status between the *work* block and its *neighbour* block are listed in Table 4.1. The discussion to follow below will illustrate the four cases associated with mesh refinement. Cases related to mesh coarsening can be treated with similar logic and the details will not be repeated herein.

For the first case of interest, both the *work* and neighbour blocks remain unchanged, the neighbouring block information stored from a previous refinement process can be reused, and further computation of neighbouring block information for this boundary element is not required. Figures 4.12–4.14 depict the refinement process in the other three cases associated with refinement.

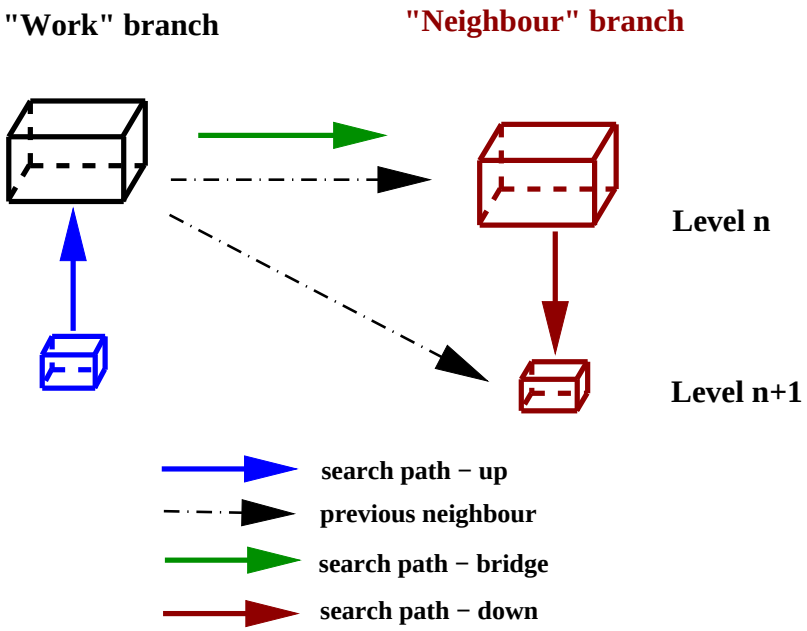


Figure 4.13: The refinement flags for the *work* block (blue) and its previous *neighbour* block (dark red) are "flagged for refinement" and "flagged for no change", respectively, and the refinement level for the previous *neighbour* can be either level n or level n+1.

Table 4.1: Nine possible combinations of the relative refinement status between the *work* block and its *neighbour* block.

Status	*work* block	*neighbour* block
(1)	flagged for no change	flagged for no change
(2)	flagged for no change	flagged for refinement
(3)	flagged for refinement	flagged for no change
(4)	flagged for refinement	flagged for refinement
(5)	flagged for no change	flagged for coarsening
(6)	flagged for coarsening	flagged for no change
(7)	flagged for coarsening	flagged for coarsening
(8)	flagged for coarsening	flagged for refinement
(9)	flagged for refinement	flagged for coarsening

Figure 4.12 illustrates the second case where the *work* block (blue) is flagged for refinement and the *neighbour* block is flagged for no change. The *work* block can have a previous neighbour (dark red) with refinement level of n-1 or n. Note that here n is

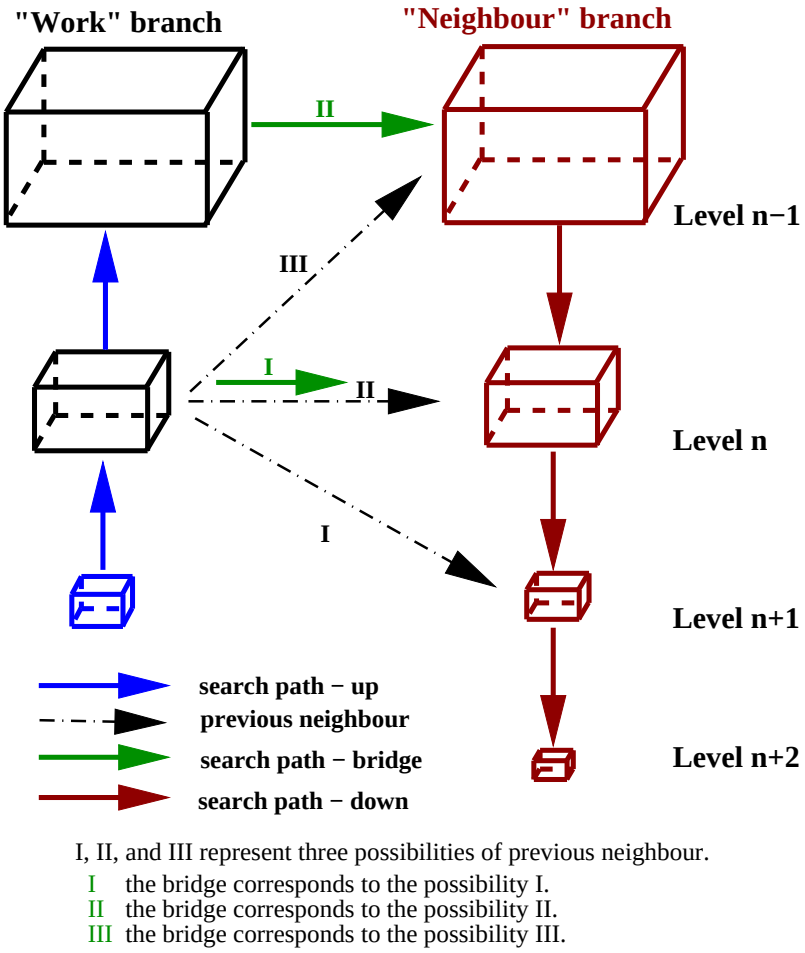


Figure 4.14: The refinement flags for the *work* block (blue) and its previous *neighbour* block (dark red) are "flagged for refinement" and "flagged for refinement", respectively, and the refinement level for the previous *neighbour* can have three possibilities: level n-1, n, and n+1.

defined to be the current refinement level, a refinement level of n-1 implies one level coarser than level n, and a refinement level of n+1 indicates one level finer than the level n. If the previous neighbour level is at level n-1, the *work* block needs to ascend the tree to its parent to retrieve its own sector information first and then traverse the tree to the neighbour branch by using the bridge built between the work and neighbour branches at level n-1. The neighbour branch is then descended to obtain the appropriate neighbour information (the appropriate neighbour's sector is in the

opposite direction of that used to connect to the *work* block). If the previous level of the *neighbour* block is n, then a bridge is defined between this *work* block and this previous neighbour block at level n. Neighbour information can then be obtained by first following the bridge to the neighbour branch and then descending the tree to determine the neighbour information for newly created block.

Figure 4.13 shows the case where the refinement flags for the *work* block and its previous *neighbour* block are "flagged for refinement" to "flagged for no change", respectively. The previous neighbour can be at either level n or level n+1. The bridge needs to be set between the *work* and *neighbour* branch at level n for these two possibilities since the *work* block (i.e., the new child block) knows nothing about its *neighbour* and must ascend the tree up to its parent and then utilizes its parent's connectivity information for determining its neighbour.

Figure 4.14 indicates the most complicated of the first four cases, i.e., both the *work* block and its *neighbour* are marked to be refined. The three possibilities for the previous level of the neighbour block are the levels n-1, n and n+1, and accordingly, there are two possibilities for the bridge. If the previous level of the neighbour block is at level n-1, then the *work* block needs to ascend the tree to obtain its own sector information and traverse the tree by using the bridge and then descending the tree to obtain its new neighbour information. The other possibility for the bridge is between two branches at the level n, for the previous level of the *neighbour* block is at either level n or n+1. Once the bridge is defined, the routine can be utilized by the *work* block to retrieve its neighbour's information. Note that the neighbour information for all the solution blocks must be updated after the entire neighbour searching procedure is complete.

4.3.3 Computation of Unstructured Root-Block Connectivity

The connectivity between unstructured blocks needs to be either specified or computed in order to carry out message passing of solution information between blocks. Unlike a structured arrangement of blocks, where the block connectivity can be eas-

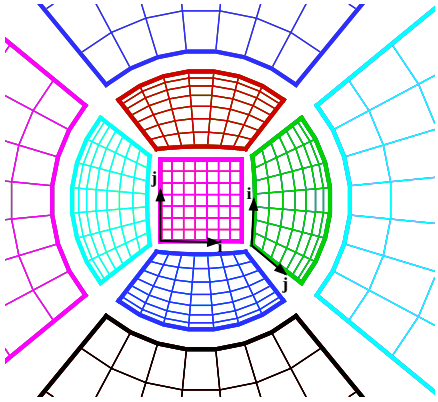


Figure 4.15: Illustration of an unstructured connectivity.

ily obtained since the connectivity is stored logically in two- and three-dimensional arrays, unstructured connectivity and orientation have to be computed. As noted in the previous section, for each block, there are 26 boundary elements (8 vertices, 12 edges and 6 faces) and connectivity of each element is computed and stored for reuse. The unstructured root-block connectivity is computed once and this connectivity information is then stored and propagated down the tree.

The logic employed in this numerical algorithm for representing the unstructured connectivity follows the methodology proposed in the CFD General Notation System (CGNS) [166]. To illustrate the unstructured connectivity between blocks, consider the grid blocks shown in Figure 4.15.

Generally, the following steps are involved in obtaining the block connectivity:

- blocks are matched to one another using block faces defined from coordinate information at the corners of the hexahedral blocks;
- transformation matrices and offsets describing the relative orientations of two blocks sharing a matching face are computed;
- neighbour information across each boundary element of a block is then stored,

including the neighbour index, matching faces, and orientation.

A transformation matrix, T, is used to relate the i, j, k (coordinates of computational frame) indices used for accessing the solution data contained in arrays of the two adjacent structured solution blocks. The transformation matrix itself has full rank and contains elements with possible values of +1, -1 and 0; the matrix is orthonormal and its inverse is its transpose. The transformation matrix is stored in a compact form as [a , b , c]. The full matrix, T, is then reconstructed as

$$\mathrm{T} = \begin{bmatrix} \mathrm{sgn(a)\ del(a\text{-}1)} & \mathrm{sgn(b)\ del(b\text{-}1)} & \mathrm{sgn(c)\ del(c\text{-}1)} \\ \mathrm{sgn(a)\ del(a\text{-}2)} & \mathrm{sgn(b)\ del(b\text{-}2)} & \mathrm{sgn(c)\ del(c\text{-}2)} \\ \mathrm{sgn(a)\ del(a\text{-}3)} & \mathrm{sgn(b)\ del(b\text{-}3)} & \mathrm{sgn(c)\ del(c\text{-}3)\,,} \end{bmatrix} .$$

where

$$\mathrm{sgn(x)} \equiv \begin{cases} +1 & \mathrm{if} x \geq 0\,, \\ -1 & \mathrm{if} x < 0,, \end{cases} \tag{4.11}$$

$$\mathrm{del(x-y)} \equiv \begin{cases} +1 & \mathrm{if} |x| = |y|\,, \\ 0 & \mathrm{otherwise}\,. \end{cases} \tag{4.12}$$

For example, suppose the computational coordinates of the centre block in Figure 4.15 is the reference system, then the values of $[\mathrm{a, b, c}] = [+2, +1, -3]$ indicate the orientation of its right neighbour block in this reference system is $[j, i, -k]$. In other words, an increase in the indices i and j for this block of interest corresponds to a decrease in index j and an increase in i for its neighbour block to the right, respectively. Furthermore, an increase in k in this block corresponds to a decrease in k for its neighbour. Since computation of the connectivity is based on block coordinate information only, it is not dependent on additional information from the grid generator; however, abutting 1-to-1 block connectivity is required. The connectivity and orientation computed from the original mesh are then stored after computation and this information can be reused, even if some of the original grid blocks are later refined.

Unstructured root-block connectivity can present some additional challenges associated with neighbour information for the edge and corner boundary elements. The number of neighbour blocks sharing each of these elements can vary and in this work

is assumed to be up to a maximum of 3, 4 or 5 depending on the structure of the multi-block mesh. Special consideration is also required for gradient reconstruction for these cases. The gradient computation procedure described in Section 3.1.3 requires modification to account for those boundary cells that may be missing because of unstructured block connectivity. To illustrate this matter, a two-dimensional case is used as an example. In Figure 4.16, the cell in block "I" marked with a filled-circle is under gradient reconstruction, the cells marked "x" denote the ghost cells that provide information for reconstruction, and the cell marked with an empty circle is the corner ghost cell. For the case of the unstructured connectivity, the neighbour block "N" may be missing along with the corner ghost cell; accordingly, the reconstruction procedure will not utilize that corner cells. In the proposed approach, a procedure is implemented to check the unstructured connectivity for each bock, and ensure that only available ghost cells are involved in the gradient reconstruction for situations with three abutting blocks. In the case of 5 blocks sharing a corner, as shown in Figure 4.17, there are more corner ghost cells being involved in the gradient reconstruction procedure, and similarly, a procedure is used to take this situation into account.

4.4 Exchange of Solution Information Between Blocks

4.4.1 Ghost-Cell Structure

Solution information is shared between adjacent blocks having common interfaces by employing two additional layers of overlapping "ghost" cells. Figures 4.18(a) and 4.18(b) show the ghost cells used for two- and three-dimensional solution blocks, respectively. The ghost cells provide solution information from neighbouring blocks and are used to facilitate communications between solution blocks. When the ghost cells are updated, buffers are used to facilitate the exchange of messages between blocks. Naturally, unstructured connectivity complicates this exchange of informa-

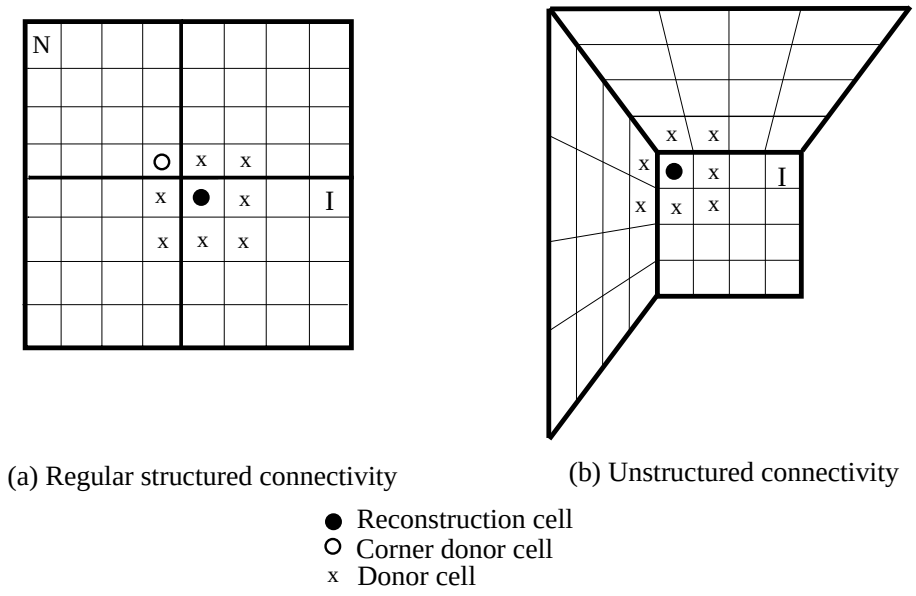


Figure 4.16: Illustration of corner ghost cells for: (a) regular structured connectivity (4 bocks abutting one another); (b) unstructured connectivity (3 blocks abutting one another).

tion. Our strategy is to load a one-dimensional buffer with the sending block's information, but with that information re-ordered according to the neighbouring block's orientation. The unloading of the buffer is therefore straightforward.

4.4.2 Conservative Flux Corrections

Additional inter-block communication is also required at interfaces with resolution changes to strictly enforce the flux conservation properties of the finite-volume scheme [13, 14]. In particular, the interface fluxes computed on more refined blocks are used to correct the interface fluxes computed on coarser neighbouring blocks and ensure that the solution fluxes are conserved across block interfaces. For three-dimensional multi-block body-fitted meshes at each time step during the solution procedure, the flux correction is determined and applied as follows:

- the fluxes of the four fine cells are summed, $\vec{\mathbf{F}}_{\text{fine}} = \sum_{\text{n}=1}^{4} \vec{\mathbf{F}}_{\text{fn}}\ A_{\text{n}}/A_{\text{coarse}}$ and are then passed to its nearest neighbour solution blocks having a lower refinement

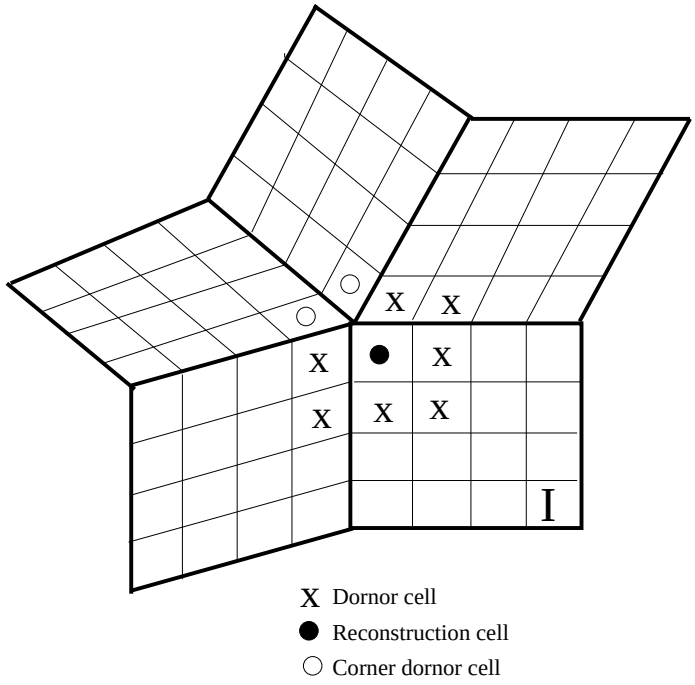
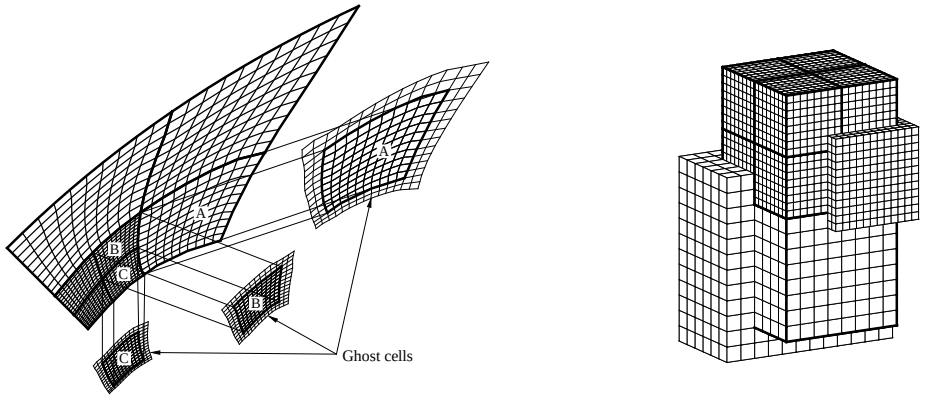


Figure 4.17: Illustration of additional corner cells introduced for general unstructured connectivity with a 5 blocks abutting one another.



(a) Two space dimensions

(b) Three space dimensions

Figure 4.18: Two layers of overlapping "ghost" cells contain solution information from neighbouring blocks.

level;

- evaluate $\Delta\vec{\mathbf{F}} = \vec{\mathbf{F}}_{\text{fine}} - \vec{\mathbf{F}}_{\text{coarse}}$, where $\vec{\mathbf{F}}_{\text{coarse}}$ is the net flux through the face of the coarse cell;
- correct the residual for the coarse cell $(\mathrm{i},\mathrm{j},\mathrm{k})$ using $\vec{\mathbf{R}}_{\mathrm{i,j,k}} = \vec{\mathbf{R}}_{\mathrm{i,j,k}} - \frac{\mathrm{CFL}\Delta \mathrm{t}_{\mathrm{i,j,k}} \mathrm{A}_{\mathrm{coarse}} \Delta\tilde{\mathbf{F}}}{V_{\mathrm{i,j,k}}}$, where $\Delta \mathrm{t}_{\mathrm{i,j,k}}$ is the time step and $V_{\mathrm{i,j,k}}$ is the cell volume.

A similar variant of this procedure can be applied for two-dimensional multiblock meshes.

4.5 Domain Decomposition and Parallel Implementation

4.5.1 Domain Decomposition

Domain decomposition is a technique of solving partial differential equations (PDEs) by decomposing an original domain into a set of smaller sub-domains [167]. In parallel computing for computational fluid dynamics, domain decomposition involves decomposing a computational mesh and distributing the sub-meshes among the processors in a multi-processor architecture. In this thesis, the computational domain of interest is a multi-block mesh, which lends itself naturally to domain decomposition. The solution blocks can be easily distributed to the processors, with more than one block permitted on each processor as shown in Figure 4.19.

4.5.2 Load Balancing and Morton Ordering

Avoiding load imbalance and limiting the communication overhead are two important considerations for a parallel solution algorithm. Many factors can lead to the load imbalance and communication overhead, such as characteristics of the computational architectures and/or the nature of the numerical algorithm.

Some strategies can be implemented to achieve effective load balancing and reduce communication costs. For homogeneous architectures (identical processors), as

used herein for all parallel computations, 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, 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. In some cases, the amount of work for each block might be variable. For example, the prediction of turbulent combusting flows can result in some blocks being significantly more involved in performing finite-rate chemical kinetics calculations. In such situations, it may become necessary to design a weighting algorithm based on the computation time for each solution block. The domain decomposition procedure can then detect and handle in a dynamic fashion load imbalances which may occur during the code execution.

Placing nearest-neighbour blocks on the same processor can also help to reduce the overall communication costs. This is usually realized by utilizing space-filling curves which can provide rather high quality partitions at very low computational costs [168–170] due to their "proximity preserving" mappings of a multidimensional space to one-dimensional space. In this work, a Morton ordering space-filling curve is adopted to provide nearest-neighbour ordering of the solution blocks in the multi-block quadrilateral and hexahedral AMR meshes, and improve the parallel perfor-

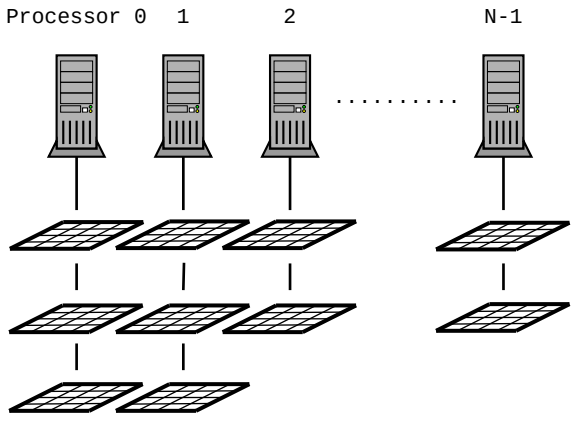


Figure 4.19: Domain decomposition is carried out by farming the solution blocks out to the separate processors, with more than one block permitted on each processor.

mance of the proposed solution method [170]. Figures 4.20–4.22 show the Morton ordering space filling curves (colored red lines) passing through each of the solution blocks (solid black lines) in a two-dimensional multiblock quadrilateral mesh, and two three-dimensional multiblock hexahedral meshes for a box geometry and a gas turbine combustor simulator, respectively.

4.5.3 Implementation Using MPI

The parallel implementation of the block-based AMR scheme was developed using the C++ programming language [171] and the MPI (message passing interface) library [172]. Use of these standards greatly enhances the portability of the computer code. 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.

The proposed domain decomposition procedure results in an efficient and highly scalable parallel algorithm that has been applied to the prediction of laminar combusting flows [94], turbulent combusting flows [95, 96], turbulent multi-phase rocket motor core flows [173], micro-scale flows [174], and compressible flows with a high-order scheme [175], in two space dimensions and the predictions for turbulent combusting flows in three space dimensions [97]. Details of the parallel performance of the solution method, along with numerical results for two- and three-dimensional flows, follow in Chapters 5 and 6.

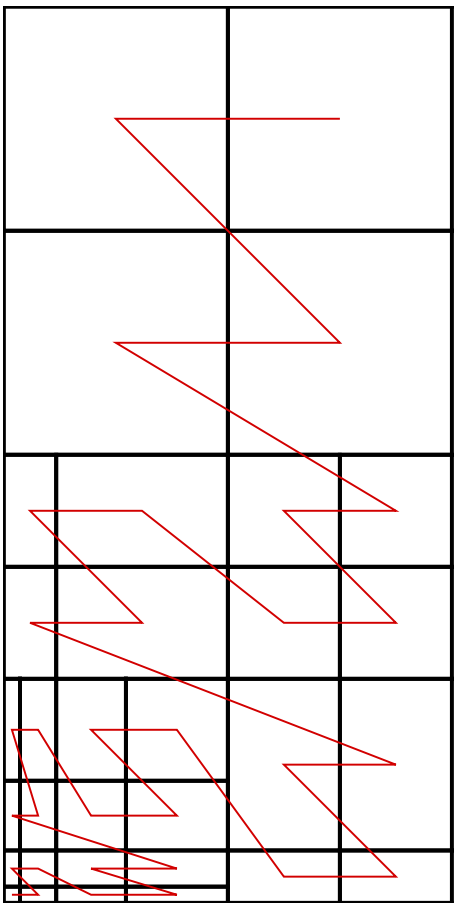


Figure 4.20: Morton ordering space filling curve used to provide nearest-neighbor ordering of blocks for efficient load balancing of blocks on multiple processors. The colored red line represents the space filling curve passing through each of the solution blocks in the multi-block quadrilateral mesh.

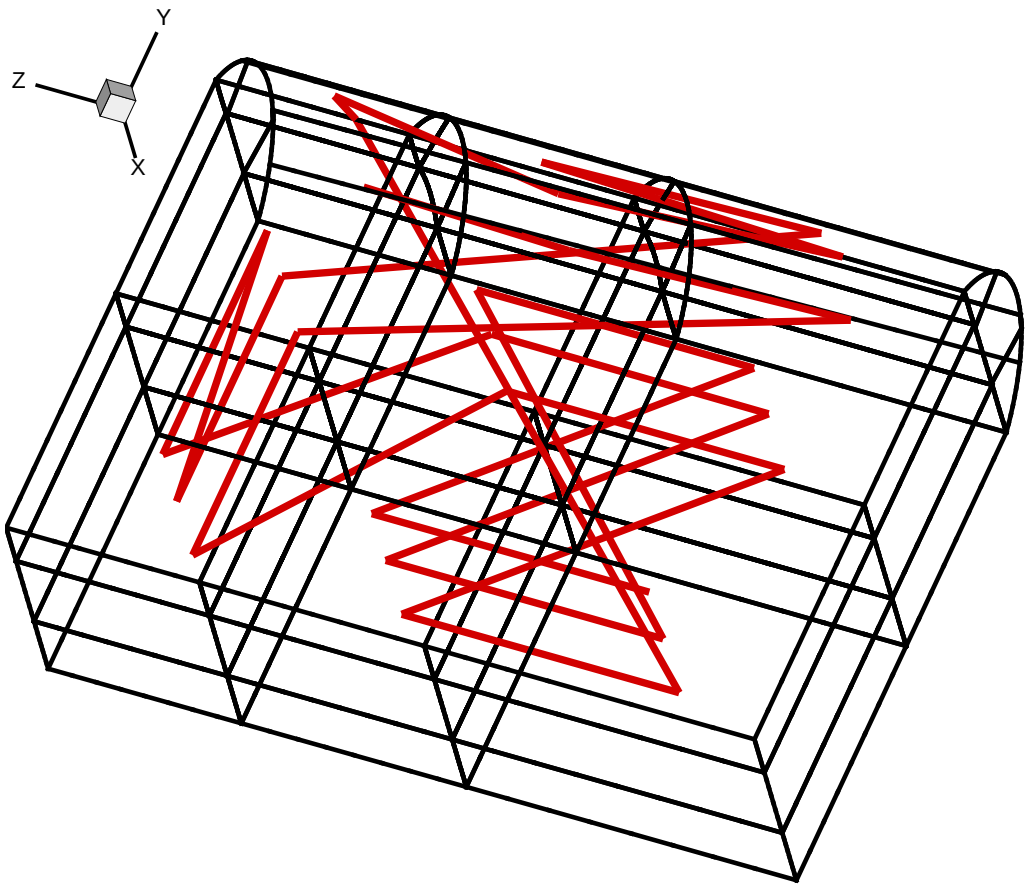


Figure 4.21: Morton ordering space filling curve used to provide nearest-neighbor ordering of blocks for efficient load balancing of blocks on multiple processors. The colored red line represents the space filling curve passing through each of the solution blocks in the multi-block hexahedral mesh for a box.

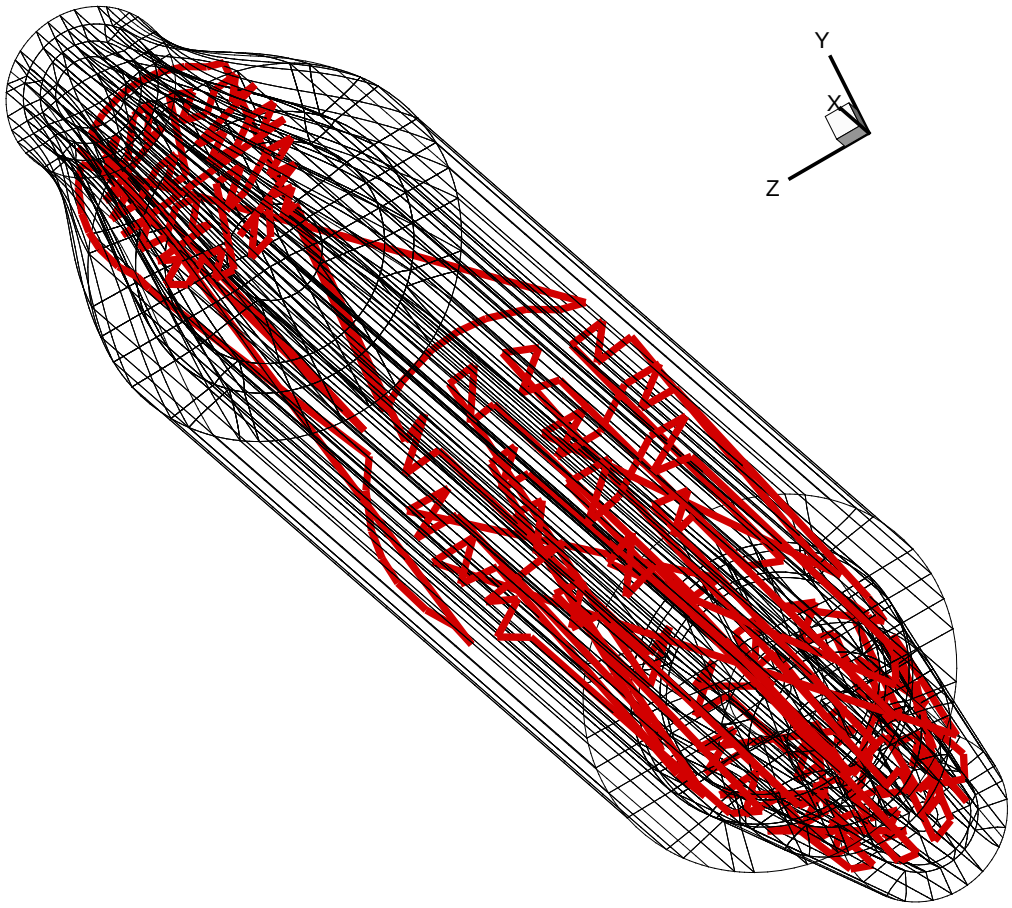


Figure 4.22: Morton ordering space filling curve used to provide nearest-neighbor ordering of blocks for efficient load balancing of blocks on multiple processors. The colored red line represents the space filling curve passing through each of the solution blocks in the multi-block hexahedral mesh for a gas turbine combustor simulator.

Chapter 5

Verification of Proposed Numerical Scheme

Verification and validation of CFD solutions and/or simulations is an important component of solution algorithm development in order to establish credibility and performance of newly proposed methods. While no standards yet exist for CFD verification and validation, verification and validation guidelines have been established by the AIAA [176]. According to these guidelines, validation is defined as the "process of determining the degree to which a model is an accurate representation of the real world from the perspective of the intended uses of the model" whereas verification is defined as the "process of determining that a model implementation accurately represents the developer's conceptual description of the model and the solution to the model" [176]. In other words, a verification assessment examines if the implementation of the mathematical model and/or solution procedure is correct and produces expected results. A validation assessment determines the degree to which the predicted numerical results agree with the known science and/or physical reality.

While a complete and formal verification and validation study has not been carried out for the proposed parallel AMR scheme described herein, partial verification of some key aspects of the proposed solution method is provided in this chapter by considering numerical predictions for several one-, two-, and three-dimensional flow problems with well established solutions. Both non-reacting and reacting flows are

considered. Having established some credibility in this manner, the performance of the proposed parallel solution algorithm for turbulent non-premixed combusting flows (with some validation) is then considered in Chapter 6.

An overview of this chapter is as follows. Section 5.1 compares the predictions to the analytic solution for a shock tube problem to verify the implementation of the inviscid operators. In Section 5.2, the calculations of a non-reacting laminar cavity flow driven by a moving lid and a Couette flow in a channel with a moving wall are considered in order to verify the implementation and accuracy of the spatial discretization scheme for the viscous operators. Section 5.3 provides verification of the turbulence two-equation k-ω model implementation by comparing numerical results to the experimental data of Laufer for non-reacting fully-developed turbulent flows in both a duct and a pipe. The preconditioned multigrid convergence-acceleration for the turbulent pipe flow is also discussed. The parallel efficiency of the parallel AMR algorithm is assessed for two-dimensional flows. Finally, Section 5.4 provides partial verification of the proposed algorithm for reactive flow simulations.

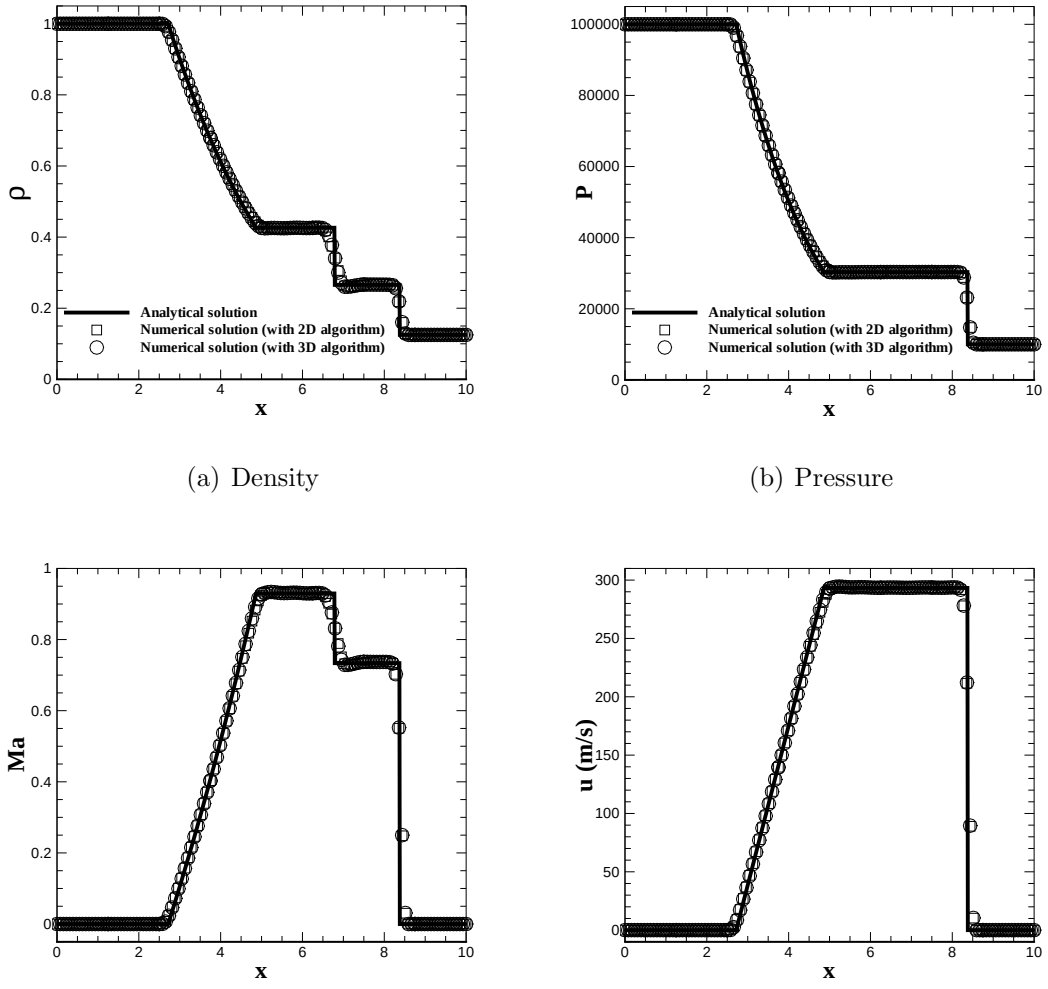
The parallel implementation of the proposed parallel AMR scheme was carried out on a parallel cluster of 4-way Hewlett-Packard ES40, ES45, and Integrity rx4640 servers with a total of 244 Alpha and Itanium 2 processors. A low-latency Myrinet network and switch are used to interconnect the servers in the cluster. All of the numerical results reported in this chapter were obtained using this parallel cluster.

5.1 Verification of Inviscid Operators

Previous studies have considered the verification of the inviscid spatial discretization operators used herein for two-dimensional flows [94, 177]. Further verification of the implementation and accuracy of the inviscid operators is provided here by comparing numerical predictions to the analytic solution of a one-dimensional shock-tube problem for a non-reacting inviscid gas. The initial data for the one-dimensional shock-tube problem contains two regions of stagnant gas (left and right), at different pressures ($p_{\mathrm{L}}/p_{\mathrm{R}}=10$) and densities ($\rho_{\mathrm{L}}/\rho_{\mathrm{R}}=10$). The working gas is assumed to be air and is defined in terms of its two major components: nitrogen (N_2) and oxygen

(O_2), with mass fractions $C_{N_2} = 0.735$ and $C_{O_2} = 0.235$, respectively. The mixture was assumed to have a fixed composition and to be non-reacting with thermodynamic properties obtained from the McBride and Gordon empirical data sets [104].

The shock-tube problem was solved in a time accurate fashion using a second-order Runge-Kutta method until time t = 6.1 ms. The Roe flux function with the Barth-Jespersen limiter, was used to solve the problem on a 10 m by 1 m computational grid of 128×2 cells using the two-dimensional algorithm, and on a 10 m by 1 m by 1 m computational grid of $128 \times 2 \times 2$ cells using the three-dimensional algorithm, with a CFL number of 0.5. The predictions of the density, pressure, Mach number and velocity are all compared to the analytic solution in Figures 5.1(a)–5.1(d). It can be seen that the two- and three-dimensional results are virtually identical and agree very well with the analytical solution. Numerical predictions were also carried out for both the two-dimensional shock box and the three-dimensional shock cube problems and the results are compared to each other. Again, air was used as the working gas and the flow was taken to be initially stagnant with a quarter region of this square domain having a pressure of 1.013×10^5 Pa and a density of $1.225\,\mathrm{kg/m^3}$. The rest of the domain was assigned a pressure and density of four times that of the low pressure region. The predictions were obtained for time t = 2 ms using the same time marching method, flux function and the limiter as those prescribed above. A 1 m by 1 m computational grid of 96×96 cells was used for solutions from the two-dimensional algorithm, and a 1 m by 1 m by 1 m computational grid of $96 \times 96 \times 2$ cells was used for solutions from the three-dimensional algorithm. The CFL number in both cases was 0.65. The predictions of the Mach number and pressure for the shock-box problem are compared in Figures 5.2(a)–5.2(d). The Mach number contours obtained using the two- and the three-dimensional algorithms are identical to each other as shown in Figures 5.2(a) and 5.2(b) and so are the predictions for pressure as indicated by Figures 5.2(c) and 5.2(d). Note the three-dimensional results are compared here by considering a two-dimensional cross section of the solution. The good agreement between the two- and three-dimensional results for this case, combined with the good agreement found in the one-dimensional case described above, are strong indications that the inviscid operators have been correctly implemented.



(a) Density

(b) Pressure

(c) Mach number

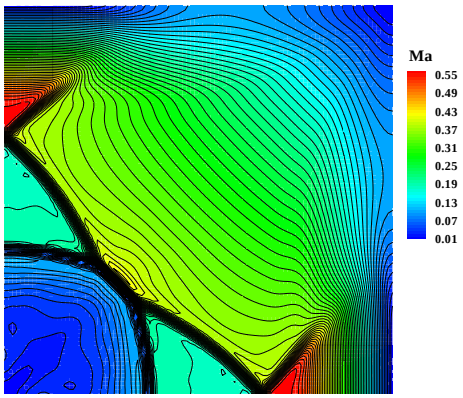
(d) Velocity

Figure 5.1: Comparisons of the numerical predictions obtained from two- and three-dimensional algorithms and the analytic solution for the one-dimensional shock-tube flow problem.

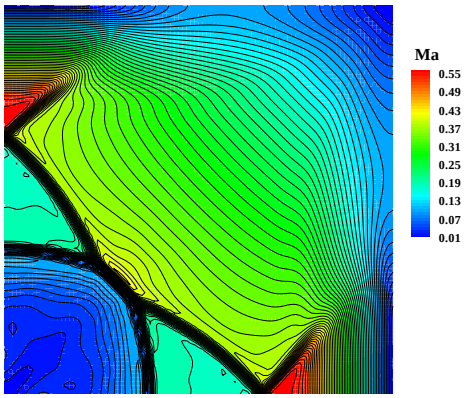
5.2 Verification of Viscous Operators

5.2.1 Two-Dimensional Laminar Couette Flow

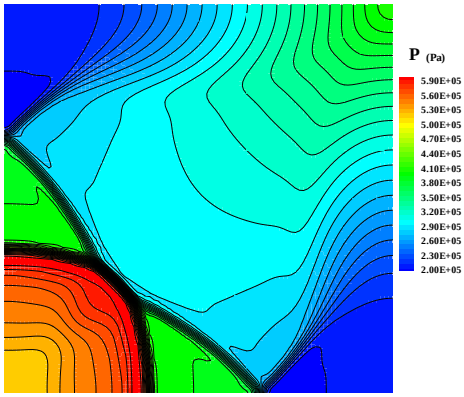
The computation of non-reacting laminar Couette flow of air in a channel with a moving wall was considered in order to demonstrate the accuracy of the viscous spatial



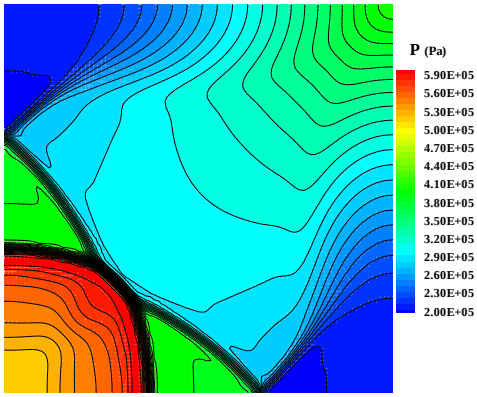
(a) Mach number (2D algorithm)



(b) Mach number (3D algorithm)



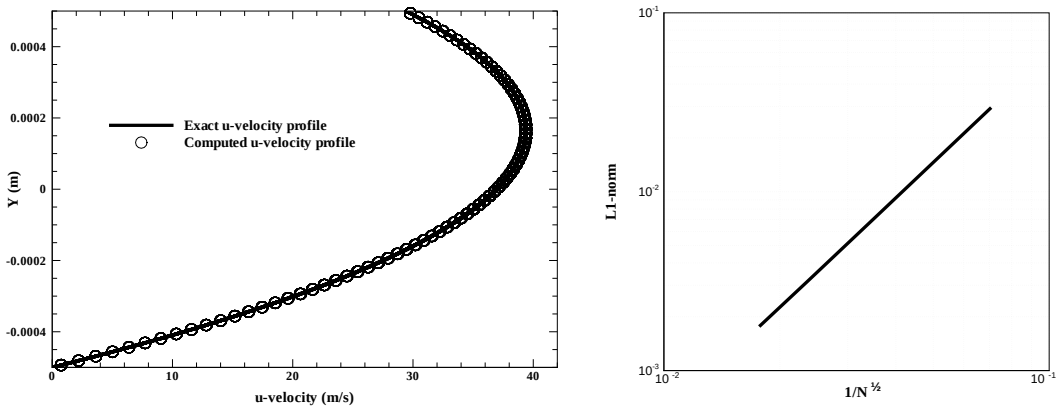
(c) Pressure (2D algorithm)



(d) Pressure (3D algorithm)

Figure 5.2: Predicted Mach number and pressure contours obtained from two- and three-dimensional algorithms for the shock box flow problem.

discretization scheme. The case considered has an upper wall velocity of 29.4 m/s and a favourable pressure gradient of $dp/dx = -3,177$ Pa/m. Numerical solutions were obtained on a domain of 0.2 m by 0.001 m with a computational grid of 60×80 cells. The results are compared to the exact analytical solution that can be obtained for this case. The predicted velocity profile is compared to the exact analytic solution



(a) Comparison of predicted and exact solutions of the axial velocity profile

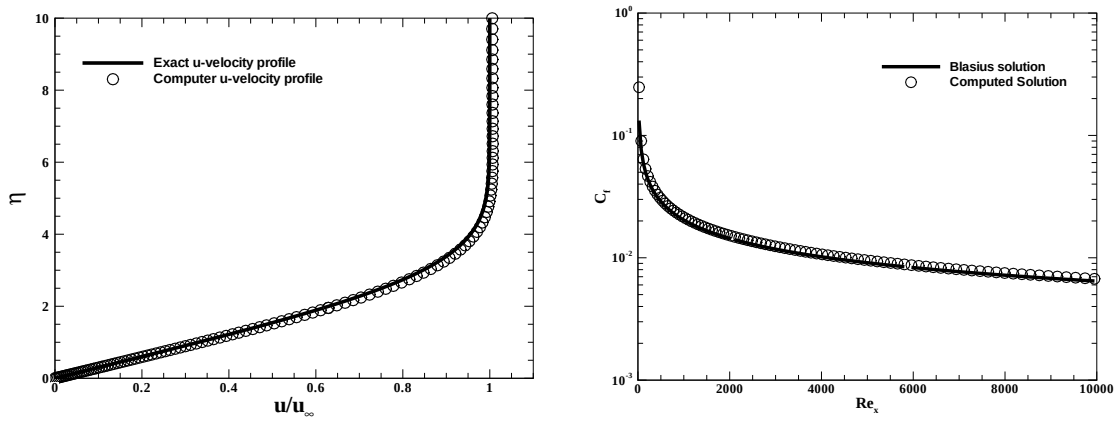
(b) L_1-norm of the solution error as a function of mesh size

Figure 5.3: Numerical predictions of two-dimensional laminar Couette flow, Re = 1.98×10^3 and Ma = 0.086.

for this essentially incompressible isothermal flow in Figure 5.3(a). Good agreement between the numerical and analytical solutions can be observed. The L_1-norm of the error in axial component of velocity is also shown in Figure 5.3(b). The slope of the norm is 2.02, indicating that the proposed finite-volume scheme is indeed second-order accurate.

5.2.2 Two-Dimensional Laminar Flat Plate Boundary Layer Flow

The computation of laminar flow of non-reacting air over a flat plate at zero incidence is considered next to further demonstrate the accuracy of the viscous spatial discretization procedure. The flat plate has a length of 0.002 m. The computational domain was 0.018 m×0.008 m. At the inflow plane of the computational domain, which is set about 0.008 m upstream from the leading edge of the plate, Dirichlet-type boundary conditions were used for all flow quantities except for pressure. A Neumann-type boundary condition was used for the pressure. An adiabatic wall boundary condition was applied to the surface of the flat plate. At the outflow



(a) Non-dimensional u-velocity component at $\mathrm{Re}_x = 8.0\times10^3$

(b) Comparison of predicted and exact skin friction coefficients

Figure 5.4: Numerical predictions of two-dimensional Laminar flat plate boundary layer, $\mathrm{Re}=1.0\times10^4$ and $\mathrm{Ma}=0.2$.

plane, the ambient pressure was specified and Neumann-type boundary conditions are applied for all other flow quantities. The free-stream Mach number and Reynolds number, based on the length of the plate, for the case considered are $\mathrm{Ma}=0.2$ and of $\mathrm{Re}=1.0\times10^4$, respectively. The exact solution of the incompressible boundary layer equations first obtained by Blasius is given by Schlichting [178].

The predictions of the non-dimensional x-direction velocity component and the skin friction coefficient were obtained on a mesh consisting of 92 blocks and 70,656 cells. The mesh was clustered towards the surface of the flat plat and the first cell normal to the plate is located at a distance of approximately 3×10^{-5} m. The profiles of x-direction velocity component shown in Figure 5.4(a) are at $\mathrm{Re}_x=8.0\times10^3$. Figure 5.4(b) compares the numerical and the exact analytical skin friction coefficients. It can be seen that the x-direction velocity component and the skin friction coefficient are in excellent agreement with the Blasius solution, providing further confidence in the scheme's accuracy.

5.2.3 Two-Dimensional Laminar Cavity Flow

The proposed two-dimensional solution algorithm has also been applied to a well known benchmark problem: lid-driven cavity flow. Figure 5.5 shows a schematic of the cavity-flow geometry and the boundary conditions. Despite its simple geometry, the driven cavity flow possesses rich flow physics manifested by multiple counter-rotating and re-circulating regions in the corners of the cavity depending on the Reynolds number. The steady laminar flow in this square, lid-driven cavity with a moving wall velocity of 34.1 m/s (Re = 100) was computed and compared with the previous computational results of Ghia *et al.* [179].

The solutions to the lid-driven cavity flow problem described above were obtained on a sequence of refined meshes to establish the grid-convergence of the solution. In particular, predictions have been obtained on a sequence of three adaptively refined grids, each consisting of blocks with 8×8 cells: 16 blocks (1,024 cells), 64 blocks (4,096 cells), and 97 blocks (6,656 cells). Figures 5.6(a) and 5.6(b) illustrate a mesh with 5-levels of refinement giving a refinement efficiency of 0.75 and as well as a mesh with 7-levels of refinement having a refinement efficiency of 0.976. Note that the refinement efficiency is defined as $\eta = 1 - N_{\text{cells}}/N_{\text{uniform}}$ where N_{cells} is the total number of cells and N_{uniform} is the total number of cells that would have been used on a uniform mesh composed of cells of a size corresponding to the finest level of refinement. Figures 5.6(c) and 5.6(d) show the u velocity contours corresponding to the meshes in Figures 5.6(a) and 5.6(b), respectively. The results from the refinement study are shown in Figures 5.7(a) and 5.7(b). The u-velocity profiles along a vertical line and the v-velocity profiles along a horizontal line along the geometric center of the cavity are shown and compared to the original numerical results of Ghia *et al.* [179] which are indicated by the square symbols in Figures 5.7(a) and 5.7(b). These profiles are in good agreement with the original numerical results. Moreover, it is apparent that the predicted solution does not change appreciably as the mesh is refined from 4,096 cells to 6,656 cells. In this sense, the numerical solutions can be said to be grid independent.

As mentioned in Section 3.3.2, multigrid convergence was significantly affected by

the prolongation operator for highly stretched meshes. The effect of four different prolongation operators on convergence rate for this laminar cavity flow was studied using three stretched meshes. The size of the three meshes was same: 4 10×10 cell blocks with 1,024 cells. The ranges of the cell aspect ratio for these three meshes were about 1 to 250, 10 to 5.0×10^3, and 100 to 1.0×10^6. Figures 5.8(a) and 5.8(b) show the mesh whose cell-aspect-ratio range is from 10 to 5.0×10^3 and the convergence comparisons for this mesh, respectively. The mesh shown has mesh points clustering toward the four corners. This is a feature of the other two meshes considered in the prolongation study.

The four prolongation operators of interest are simple injection, cell-area weighted interpolation, standard bi-linear interpolation, and standard bi-linear plus linear interpolation. The convergence results from the four prolongation operators provide similar performance for the mesh with cell aspect ratios of 1 to 250. For the case with mesh having a cell-aspect-ratio range of 100 to 1.0×10^6, the calculation failed to converge for both the bi-linear plus linear and the bi-linear operators, while the simple injection operator converged faster than the cell-area weighted operator. This indicates that the simple injection can be used for meshes having high cell-aspect-ratios. Figure 5.8(b) shows that, overall, both the standard bi-linear operator and the bi-linear plus linear operator result in the fastest convergence rate among the pro-

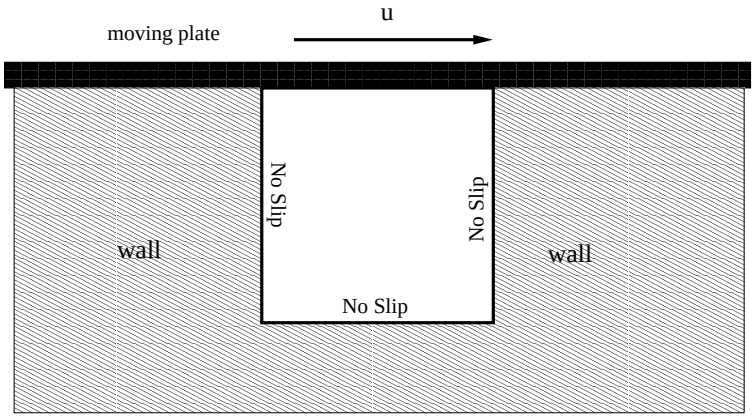
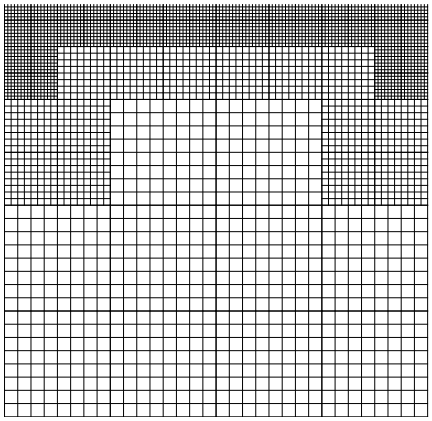
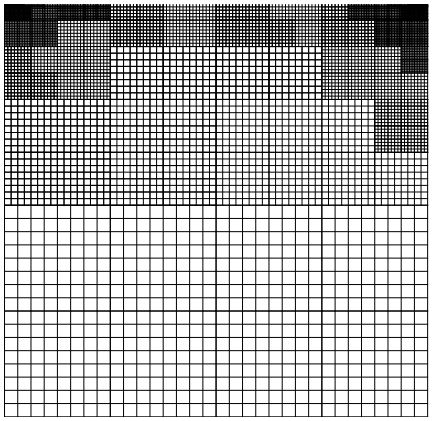


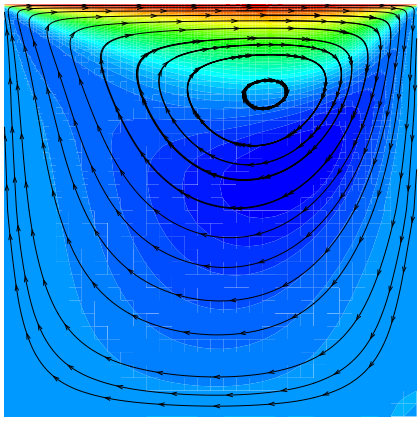
Figure 5.5: Schematic of a two-dimensional driven-cavity laminar flow.



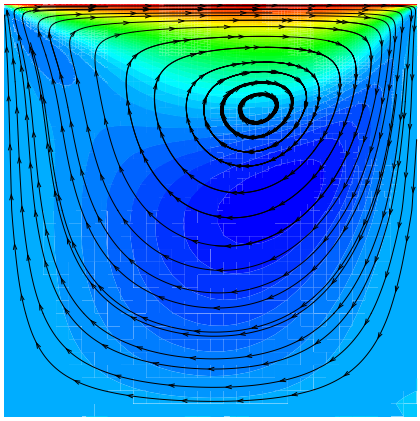
(a) Medium mesh consists of 64 8×8 cell blocks (4,096 cells): 5 levels of refinement with a refinement efficiency of 0.75.



(b) Fine mesh consists of 97 8×8 cell blocks (6,656 cells): 7 levels of refinement with a refinement efficiency of 0.976.



(c) u contour on the medium mesh

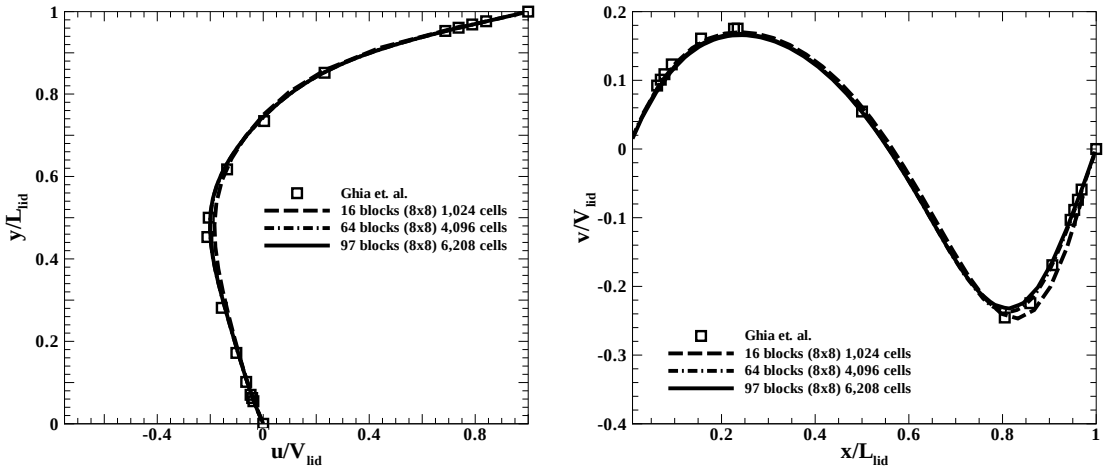


(d) u contour on the fine mesh

Figure 5.6: Two refined meshes used in the numerical solution of the laminar lid-driven cavity flow and the computed u velocity contours for the medium mesh (4,096 cells) and the fine mesh (6,656 cells).

longation operators, the simple injection seems sufficient, and the cell-area weighted operator is the slowest.

Based on this study, it is suggested that a cell-aspect-ratio-based sensor can be



(a) u-velocity profile

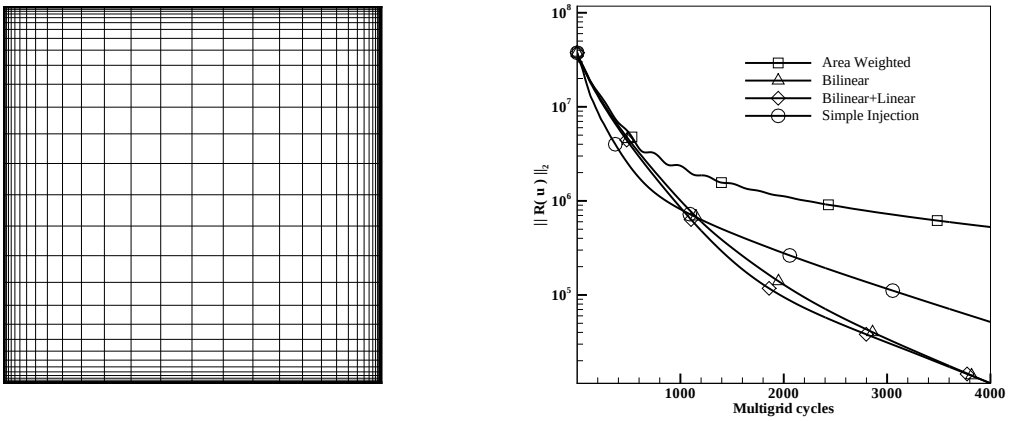
(b) v-velocity profile

Figure 5.7: Comparison of computed velocity profiles along the vertical and the horizontal center-line of the cavity with data of Ghia *et al.* (1982), Re = 100 and Ma = 0.1.

applied as a switch in the approach of the prolongation operator. For the two-dimensional cases with relatively complex geometry considered in Chapter 6, a cell-aspect-ratio-based sensor was implemented in the multigrid algorithm as follows: simple injection is used for cells with aspect ratio higher than a cutoff value, and standard bi-linear interpolation is then employed for cells with aspect ratios lower than that cutoff value.

5.2.4 Three-Dimensional Laminar Couette Flow

The computation of non-reacting laminar Couette flow in a channel with a moving wall was re-considered in order to demonstrate the accuracy of the viscous spatial discretization scheme in the three-dimensional case. Again, Couette flow with an upper wall velocity of 29.4 m/s and a favourable pressure gradient of $dp/dx = -3,177$ Pa/m was investigated on a 0.2 m by 0.001 m by 0.001 m computational grid of $60 \times 80 \times 2$ cells. The cell-face gradients are evaluated using the formula (Equation (3.36)) proposed by Mathur and Murthy [119]. The predicted velocity profile is given and compared to the exact analytic solution for this nearly incompressible isothermal



(a) Stretched mesh (b) Convergence

Figure 5.8: Influence of different prolongation operators on the multigrid convergence for the laminar cavity flow with a stretched mesh having cell aspect ratios from 10 to 5.0×10^3.

flow in Figure 5.9. Note that the exact solution is based on the assumption that the flow is incompressible ($\rho = \text{constant}$, $\text{Ma} \rightarrow 0$). It can be seen that the predicted results of the three-dimensional algorithm match well with the analytic solution and the two-dimensional predictions, providing support for that the implementation of the numerical viscous flux operators in the three-dimensional version of the proposed solution scheme is correct.

5.3 Verification of k-ω Turbulence Model

The classical problems of flow in a channel, or duct, and a pipe are excellent cases for verifying the implementation of the k-ω turbulence model considered here. The flow becomes fully-developed at locations sufficiently far from the inlet, i.e., properties no longer vary with distance along the channel/pipe.

The verification of the implementation of the k-ω turbulence model for non-reacting turbulent flows was performed by comparing numerical results to the experimental data of Laufer [1, 180] for non-reacting, fully-developed turbulent flow in

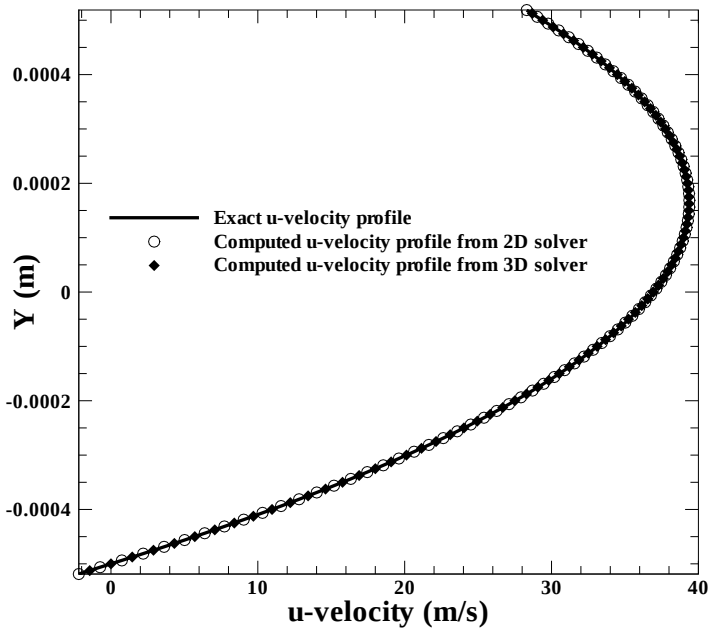
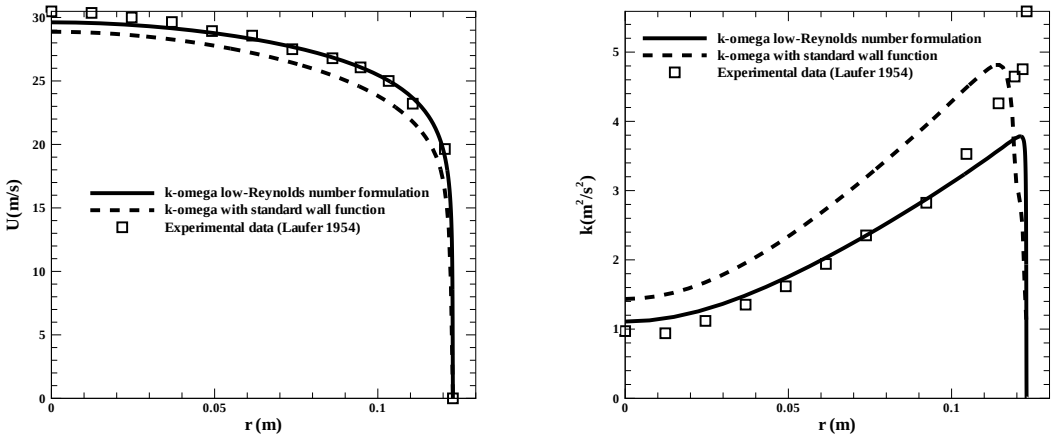


Figure 5.9: Computed u-velocity profile along vertical center-line of the laminar Couette flow compared to the analytic data and the two-dimensional calculated profile, Re$=1.98\times10^3$ and Ma$=0.086$.

a circular cross-section pipe and a rectangular channel. The working gas was air in both cases. For the pipe flow, the radius of the pipe was 0.123 m and the Reynolds number based on the center-line velocity, $U_o = 30.48$ m/s, was $Re = 5.0\times10^5$; for the channel flow, the width of the channel was 0.127 m and the Reynolds number based on the center-line velocity, $U_o = 7.4$ m/s, was $Re = 6.16\times10^4$. The results of the comparisons between the predictions of the proposed solution algorithm and experimental data are now discussed.

5.3.1 Two-Dimensional Fully-Developed Pipe Flow

Numerical solutions were first carried out for two-dimensional (axisymmetric) fully-developed turbulent pipe flow. Solutions for both the wall function and low-Reynolds-number formulation of the k-ω turbulence model are compared to measured mean axial velocity and turbulent kinetic energy profiles in Figures 5.10(a) and



(a) Mean axial velocity

(b) Turbulent kinetic energy

Figure 5.10: Comparison of predicted solutions with experimental data [1] for fully developed turbulent pipe flow, Re$=5.0\times10^5$ and Ma$=0.089$.

5.10(b). Calculations with the low-Reynolds-number formulation were performed using 80 cells in the radial direction with 3-4 of those cells lying within the laminar sublayer. The first cell off the wall was located at $y^+\approx0.6$. The result using the wall function was obtained using 32 cells in the radial direction with the first cell located at $y^+\approx43$. The agreement between the experimental data and numerical results for this case is generally quite good. As expected, it is evident that the k-ω model is able to reproduce the characteristic features of fully-developed pipe flow.

Multigrid Acceleration

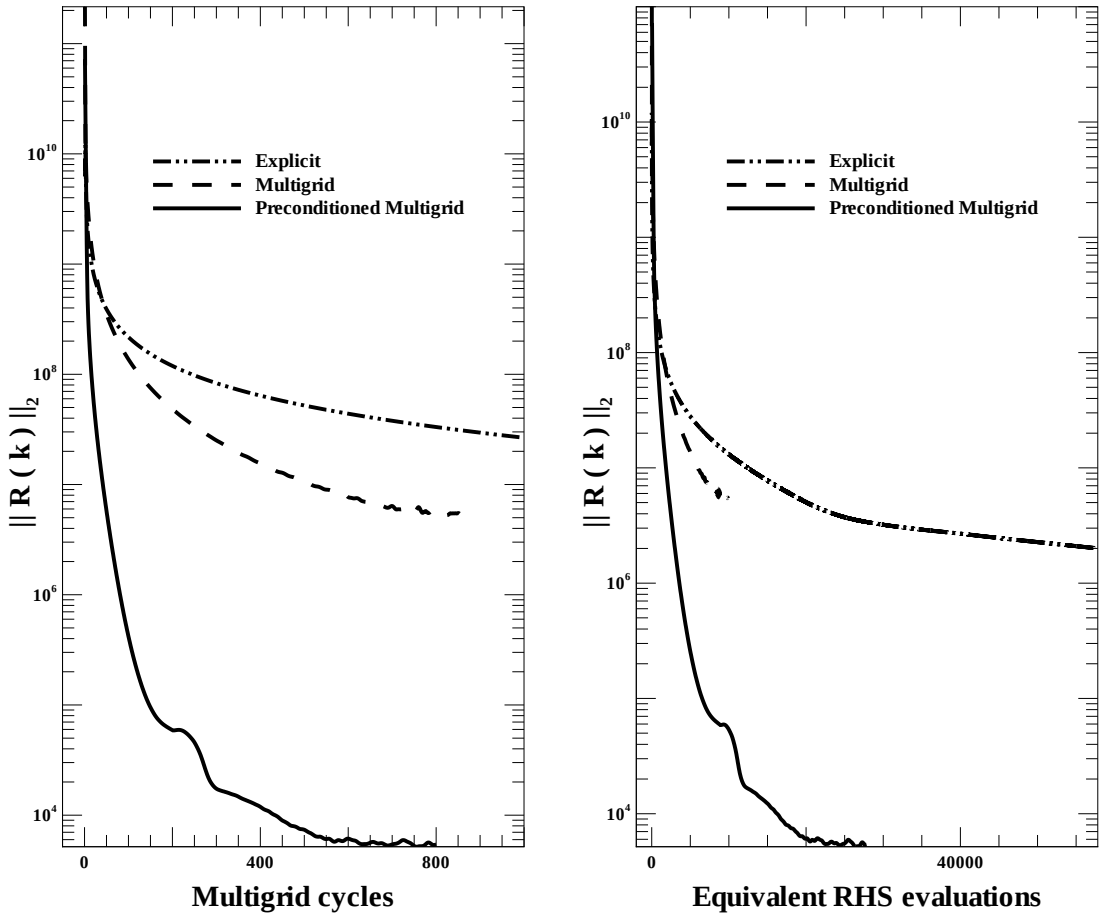
Convergence acceleration provided by the preconditioned multigrid algorithm was also examined for the fully developed turbulent pipe flow problem. A mesh of size 1,024 cells and having cell aspect ratios in the range of 10 to 2×10^5 and an off-wall spacing of 7.0×10^{-7} m was used in this study. There were 32 cells in the radial direction and an automatic wall boundary treatment was employed. The influence of using different multigrid levels and cycles on convergence features has also been investigated.

Figures 5.11(a) and 5.11(b) compare the convergence rates achieved for the turbulent pipe flow using the explicit time marching scheme with local time-stepping, the proposed multigrid algorithm with explicit smoother, and the preconditioned multigrid approach described in Chapter 3. The convergence rate is shown as a function of both the number of iterations and the number of equivalent right-hand side (RHS) evaluations. Clearly, the preconditioned multigrid results in a more efficient convergence rate than the others. Notice from the figures that the preconditioned multigrid algorithm exhibited a convergence stall after the residual in the turbulent kinetic energy dropped to be about 10^4. It is felt that this effect is due to the nonlinear nature of the slope limiters and their activation in smooth regions of the flow field [124]. This convergence stall can be alleviated by freezing the limiter after the residual has dropped to a predefined level; however, this technique was not employed here.

Theoretically, the work count of multigrid has a linear relationship with the number of unknowns and hence mesh size. Even for this relatively simple case, the ideal linear relationship is not being achieved. Figure 5.12 shows the increase in computational work as a function of mesh size for the fully-developed turbulent pipe flow. For this case, the work increases as the mesh size to the power of about 1.8 (i.e., work $\propto N^{1.8}$). Nevertheless, the current preconditioned multigrid is certainly effective and provides rather significant convergence acceleration.

Tables 5.1–5.3 summarize some convergence features for the fully developed turbulent pipe flow problem. Note that the maximum grid level for these cases was chosen to be 3, allowing for relatively smaller-sized solution blocks. Table 5.1 lists the numerical data from using both the regular and the preconditioned multigrid and the results for grid-level and multigrid-cycle effects are presented in Table 5.2 and Table 5.3. The term work unit (WU) is defined as the time for one right-hand-side evaluation on the finest mesh.

Table 5.1 indicates that the preconditioned multigrid with 5-stage optimal smoothing scheme produces a 14 times speedup over multigrid without a preconditioner and is shown in Figure 5.13(a). For both cases, the L_2 norms of the residual for the solution quantity of turbulent kinetic energy drop about six orders of magnitude. The data from Table 5.2 shows a speedup factor of four between the 3-level multigrid and



(a) Convergence rate as a function of multigrid cycles

(b) Convergence rate as a function of the number of equivalent RHS evaluations

Figure 5.11: Comparisons of 4-level V-cycle multigrid convergence between regular multigrid and preconditioned multigrid with a 5-stage optimal smoothing scheme for the fully-developed turbulent pipe flow.

the single-level computation. This grid-level influence on the convergence is shown in Figure 5.13(b). The convergence rate of 2-level is the same as that of a 3-level for this case, and both used the V-cycle. The same convergence rates for both 2- and 3-level might be due to the fact that the turbulence source terms were not recalculated on the coarse meshes. Figure 5.14 shows that the 3-level V- and W-cycle preconditioned multigrid algorithms, using a 5-stage optimal smoothing scheme, have nearly

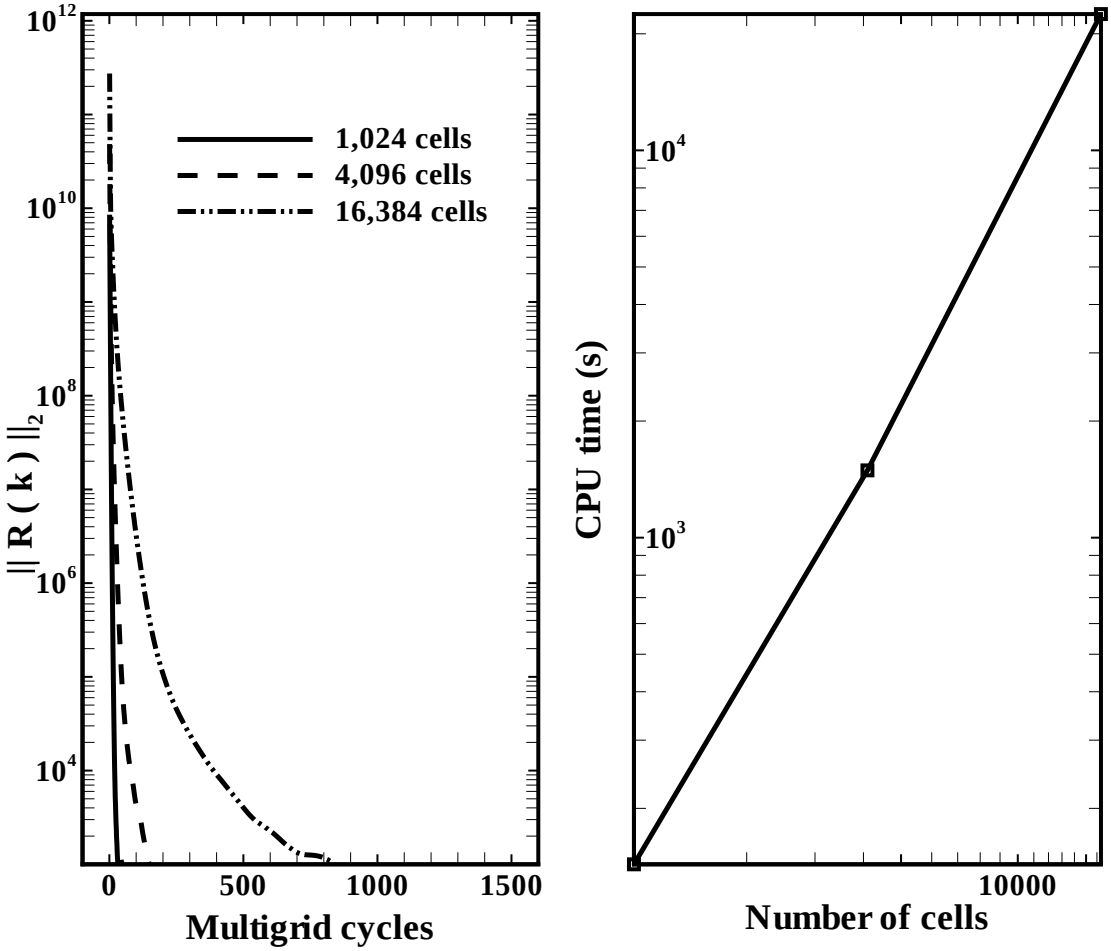


Figure 5.12: Convergence rate and computational work in terms of CPU time of a 3-level preconditioned multigrid with 5-stage optimal smoothing scheme for the fully-developed pipe flow showing effects of computational mesh size (3 mesh sizes: 1,024; 4,096; 16,384).

the same speedup in terms of multigrid cycles, while Table 5.3 indicates the V-cycle uses about half the computation time of the W-cycle. The reason might be that the W-cycle is expensive in a parallel algorithm when frequent coarse-level calculations lead to poor processor utilization. From these results, it appears that a 3-level V-cycle preconditioned multigrid should deliver an optimal speed up for computing turbulent flows. For this reason, the 3-level V-cycle preconditioned multigrid was employed

in the numerical predictions of the two-dimensional reactive and non-reactive flows described in next chapter.

Table 5.1: Matrix-preconditioner effects on convergence of the 4-level V-cycle multigrid for the fully-developed turbulent pipe flow.

Method	CPU time [min]	WUs	speedup CPU (WUs)
Multigrid	210.5	17542	1
Preconditioned multigrid	15	1250	14

Table 5.2: Grid-level effects on convergence of the V-cycle preconditioned multigrid for the fully-developed turbulent pipe flow.

Method	CPU time [min]	WUs	speedup CPU (WUs)
Single-level	35.8	2983	1
2-level	10.9	908	3.3
3-level	9.69	807.5	3.7

Table 5.3: The V- and W-cycle effects on convergence of the 3-level preconditioned multigrid for the fully-developed turbulent pipe flow.

Method	CPU time [min]	WUs	speedup CPU (WUs)
W-cycle	38	3166.7	1
V-cycle	22	1833.3	1.72

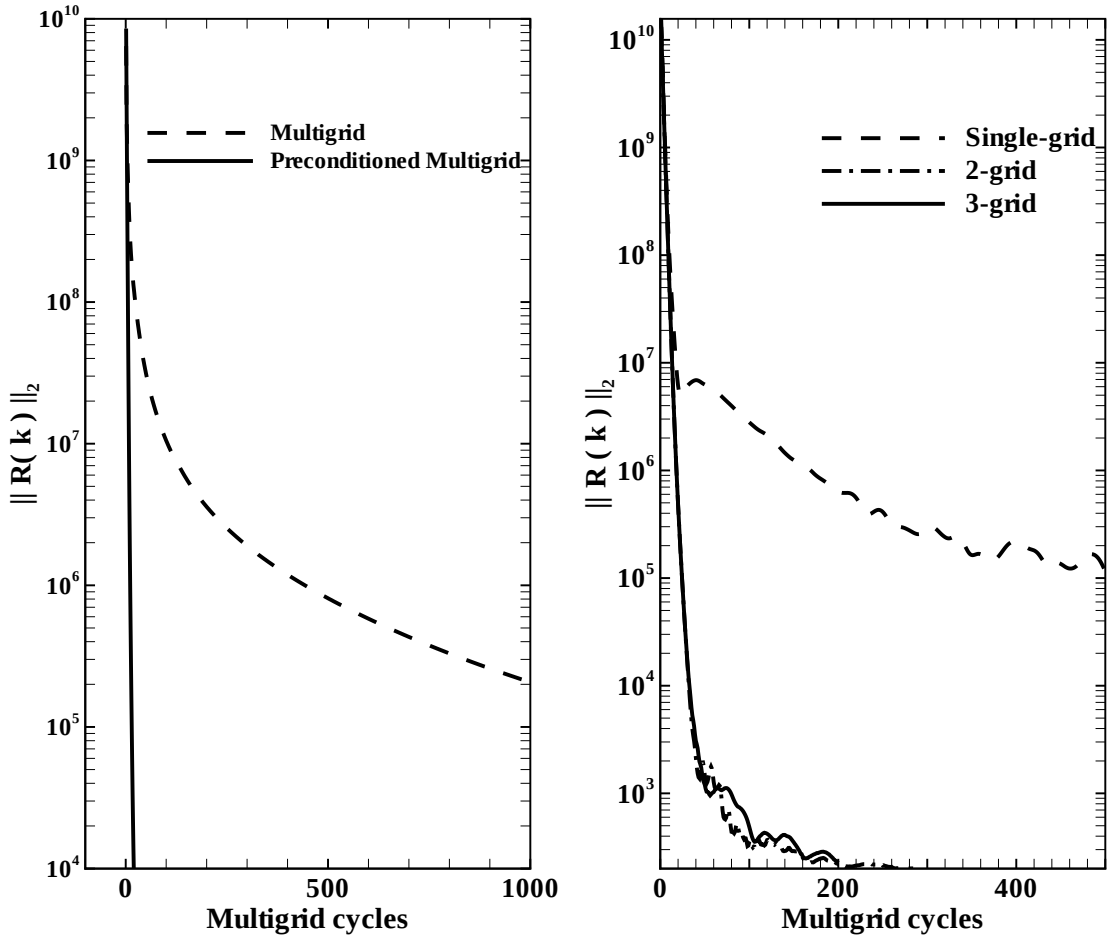
Parallel Performance

Parallel speedup, parallel scale-up, and parallel efficiency are often used to measure/evaluate the parallel performance of a parallel algorithm. The parallel speedup, S_p, is defined as

$$S_p = \frac{t_1}{t_p}\,, \tag{5.1}$$

the parallel scale-up, S_ϕ, defined as

$$S_\phi = \frac{t_1}{t_p}\,p\,, \tag{5.2}$$



(a) Effect of matrix preconditioner on multigrid convergence rate

(b) Effect of number of grid-levels on multigrid convergence rate

Figure 5.13: Convergence features for the fully-developed turbulent pipe flow: (a) comparisons between the preconditioned multigrid and regular multigrid both using 3-level V-cycle with a 5-stage optimal smoothing scheme; (b) comparisons between the grid-level effects on the preconditioned V-cycle multigrid with a 5-stage optimal smoothing scheme.

and the parallel efficiency, E_p is defined as

$$E_p = \frac{S_p}{p}\,, \tag{5.3}$$

where t_1 is the time required to solve the problem by a single processor, and t_p is the time required to solve the problem by p processors.

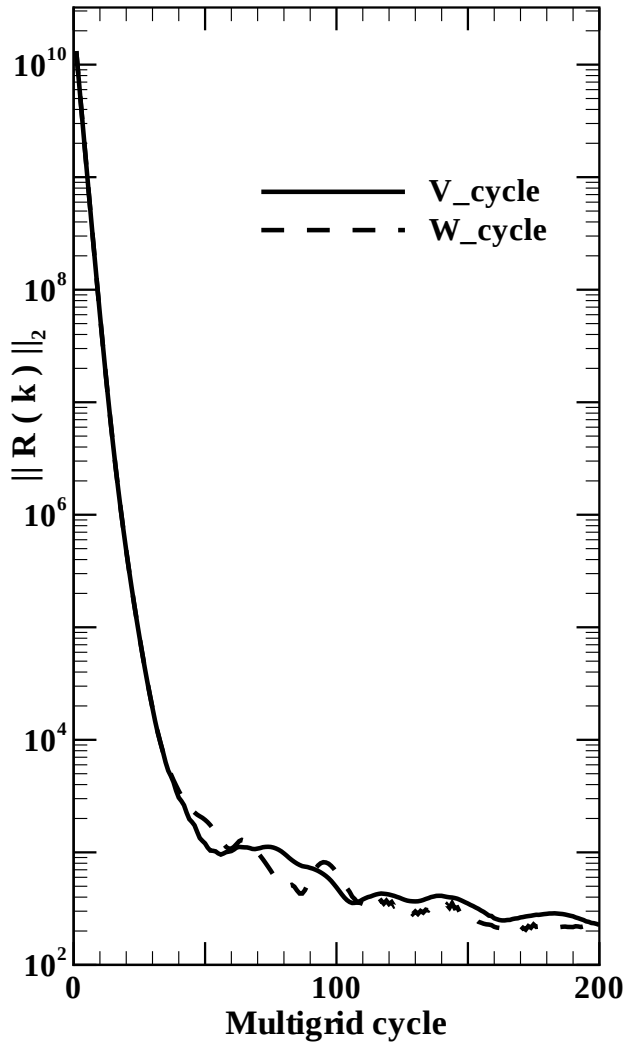


Figure 5.14: Comparisons between the V- and W-cycle preconditioned multigrid convergence rates of a 3-level preconditioned multigrid with 5-stage optimal smoothing scheme for the fully-developed pipe flow.

Parallel speedup, also know as strong scaling, is measured by taking a fixed size problem and solving the problem using an increasing number of processors. If the solver produces a result p times faster when p processors are used, then the algorithm has perfect speed up. Parallel scale-up or weak scaling is measured by increasing the

size of the problem as the number of processors is increased. Parallel performance in this case in then assessed by considering the time to solve the problem, which ideally should remain the same for any number of processors. While both the parallel speedup and the parallel scale-up are important to consider, the parallel speedup is probably more relevant for engineering problems of practical interest.

High efficiency for strong scaling is generally harder to achieve than for the weak scaling problem. Virtually, all programs contain both parallel and serial portions. The wall clock time of the parallel parts is reduced by using an increasing number of processors, but the time for sequential parts remains the same. Eventually, the time spent in the sequential parts can dominate the entire program execution time and puts an upper limit on the expected speedup. This effect, known as Amdahl's law, is represented by the formula [181]

$$S_p = \frac{1}{(f/p + (1-f))}\,, \tag{5.4}$$

where f is the parallel fraction of the program. In this thesis, by careful design, virtually entire algorithm can be executed in parallel and theoretically $f = 1$. Therefore, a perfect speedup should be expected. Nevertheless, a small serial fraction is introduced by some minor non-parallel aspects of the implementation and by the message passing and MPI library and/or other components of the operating system. Therefore, a perfect speedup is not (always) expected according to the Amdahl's law. This is illustrated by the results given below.

The parallel speedup and efficiency of the proposed parallel solution-adaptive algorithm applied to the two-dimensional turbulent pipe flow problem with a fixed size grid (64 blocks) has been assessed. Figure 5.15 shows that the parallel speedup of the block-based AMR scheme (without multigrid acceleration) is nearly linear and is at least 90% efficient for up to 32 processors using the larger (10×10) solution blocks. For the smaller (8×8) blocks, the efficiency drops slightly down to 87% efficient. Table 5.4 illustrates the parallel fraction of the program (the problem of 64 10×10 solution blocks) based on the measured S_p, for different numbers of processors. An estimate of the parallel fraction provides an indication of how Amdahl's law will affect the strong scaling. Interestingly, the serial fraction of the algorithm seems to increase

Table 5.4: The parallel fraction of the program varies with increasing the number of processors

CPU(s)	f
4	100%
8	100%
16	99.84%
32	99.52%

somewhat with the number of processors. However, this sequential fraction remains extremely low (0.16%–0.48%) and, based on this analysis, the proposed algorithm can be seen to be well suited for scaling to relatively larger numbers of processors.

This rather high level of performance should generally be expected for the two-dimensional version of the algorithm with the explicit time marching scheme. In this case, the communication overhead is low (on the order of 5–7%) and high parallel efficiency is achieved for a well load-balanced problem.

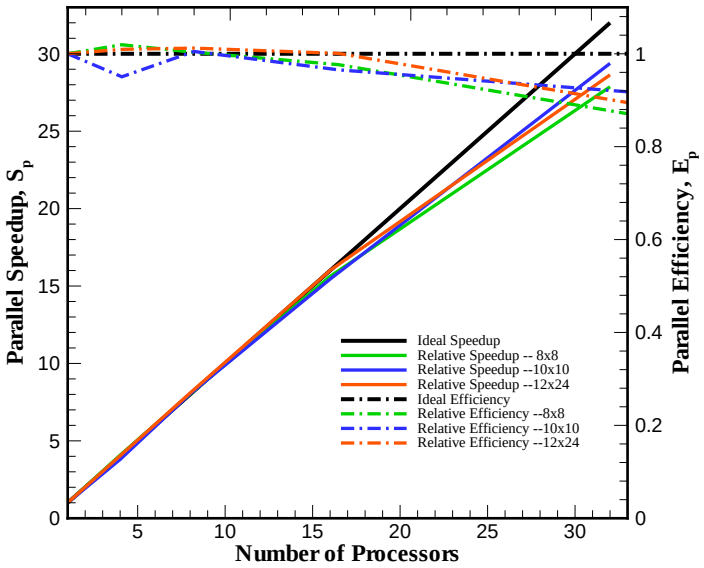


Figure 5.15: Parallel speedup (strong scaling), S_p and the parallel efficiency, E_p, for a fixed size problem using up to 32 processors.

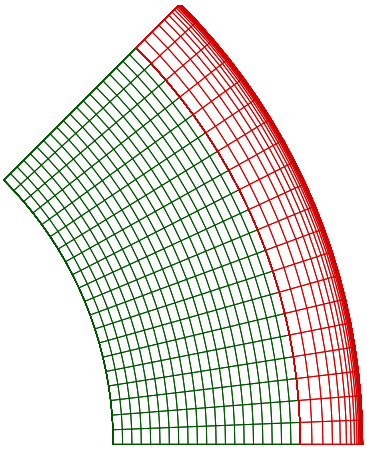
The parallel performance of the proposed solution-adaptive method for both two- and three-dimensional reactive-flow cases is considered later in Chapter 6 of this thesis.

5.3.2 Three-Dimensional Turbulent Channel and Pipe Flows

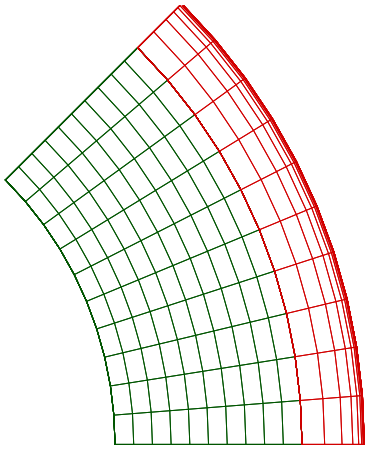
Numerical solutions obtained using the proposed parallel AMR scheme in three dimensions for both the turbulent channel and pipe flows are now investigated with the results given in Figures 5.17(b)–5.18(a). The numerical predictions for the pipe flow were obtained by using three types of near wall turbulence treatment: (1) low-Reynolds number formulation, i.e., integrating the transport equations for the two-equation turbulent model through the laminar sublayer directly up to the solid wall; (2) standard wall function; and (3) automatic near-wall treatment with a switching procedure (Equation (2.29)). The calculations were performed on a quarter of a pipe geometry. Numerical results were obtained from three meshes. There were 80 cells in the radial direction for the calculation with the low-Reynolds number formulations, 40 cells for the calculation with the automatic switching function, and 32 cells for the standard wall function.

Two cross-sections of the meshes used in the pipe-flow calculations along with colour contours of the values of y_1^+ in each computational cell are shown in Figures 5.16(a)–5.16(d). The xy-plane views of the two meshes as used in both the low Reynolds number formation and the automatic near-wall treatment, are shown in Figures 5.16(a) and 5.16(b), respectively. The meshes are stretched and clustered toward the solid wall and the first y_1^+ values are 0.45 and 16.6 for the two cases, respectively. There were two cells in the viscous sublayer for the calculation using the low-Reynolds formulation.

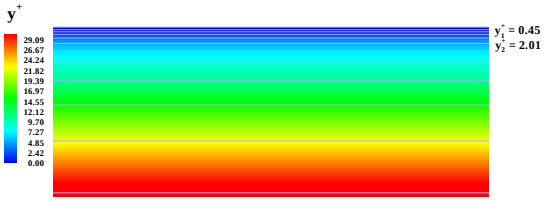
The predictions of the mean axial velocity and turbulent kinetic energy from the three types of near-wall turbulence treatment are compared to the experimental data of Laufer [1,180] in Figures 5.17(a) and 5.17(b). In general, the numerical results for the three different wall treatments are very close to one another. Good agreement



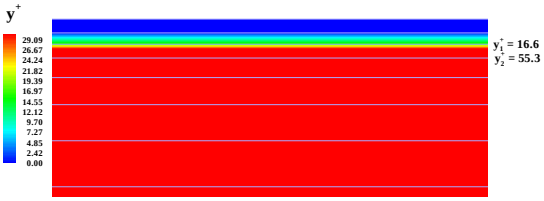
(a) Low Reynolds number formulation



(b) Automatic near-wall treatment



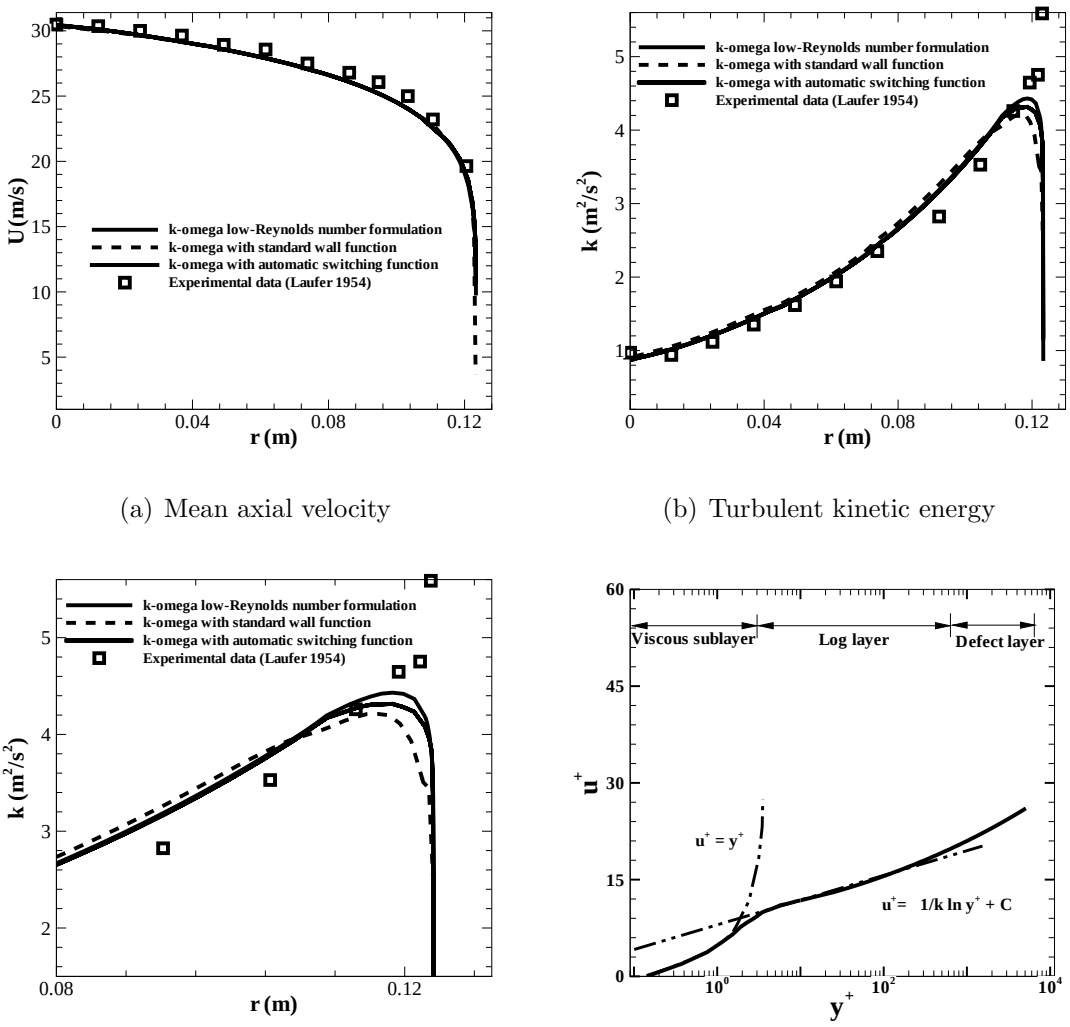
(c) y^+ for low Reynolds number formulation



(d) y^+ for the automatic near-wall treatment

Figure 5.16: Comparisons of cross-section of meshes (only showing two blocks close to the wall) and different y_1^+ for fully developed turbulent pipe flow, $\mathrm{Re} = 5.0 \times 10^5$ and $\mathrm{Ma} = 0.089$.

between the experimental data and numerical results is also observed. Figure 5.17(b) indicates that predictions of the turbulent kinetic energy from the three approaches are almost exactly the same except in the region close to the solid wall, which is not surprising because of the different near-wall treatments. Figure 5.17(c) shows a close-up of this feature. Clearly, the low-Reynolds number formulation has the best prediction of the peak value of turbulent kinetic energy. The wall function under-predicts the turbulent kinetic energy more than the others. The profile computed from using the automatic switching function falls in between and indicates that the automatic near-wall treatment appears to work well. Figure 5.17(d) shows the nu-



(a) Mean axial velocity

(b) Turbulent kinetic energy

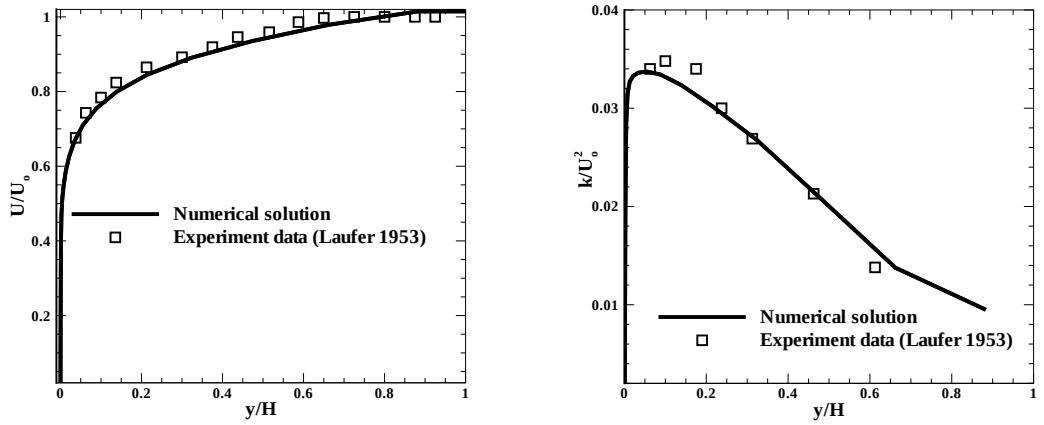
(c) Close-up of turbulent kinetic energy near the wall

(d) Predicted velocity profile for the turbulent boundary layer of the pipe flow with the low-Reynolds number formulation

Figure 5.17: Comparison of predicted solutions with experimental data [1] for fully developed turbulent pipe flow, $\mathrm{Re} = 5.0 \times 10^5$ and $\mathrm{Ma} = 0.089$.

merical prediction of the typical velocity profile for the turbulent boundary layer of the pipe flow.

For the turbulent channel flow calculation, a grid consisting of 32 cells in the cross-channel with stretching towards the wall and with 2-3 of these computational



(a) Mean axial velocity

(b) Turbulent kinetic energy

Figure 5.18: Comparison of predicted solutions with experimental data [1] for fully developed turbulent channel flow, Re$=6.16\times10^4$ and Ma$=0.022$.

cells within the laminar sublayer was used. The first cell from the wall was located at y_1^+=0.07. Figures 5.18(a) and 5.18(b) compare the predictions of the mean axial velocity and turbulent kinetic energy to the experimental data of Laufer [1]. As for the turbulent pipe-flow, good agreement between the numerical and the experimental data is again observed. It is evident that the k-ω model is able to reproduce the characteristic features of these two fully-developed non-reacting turbulent flows.

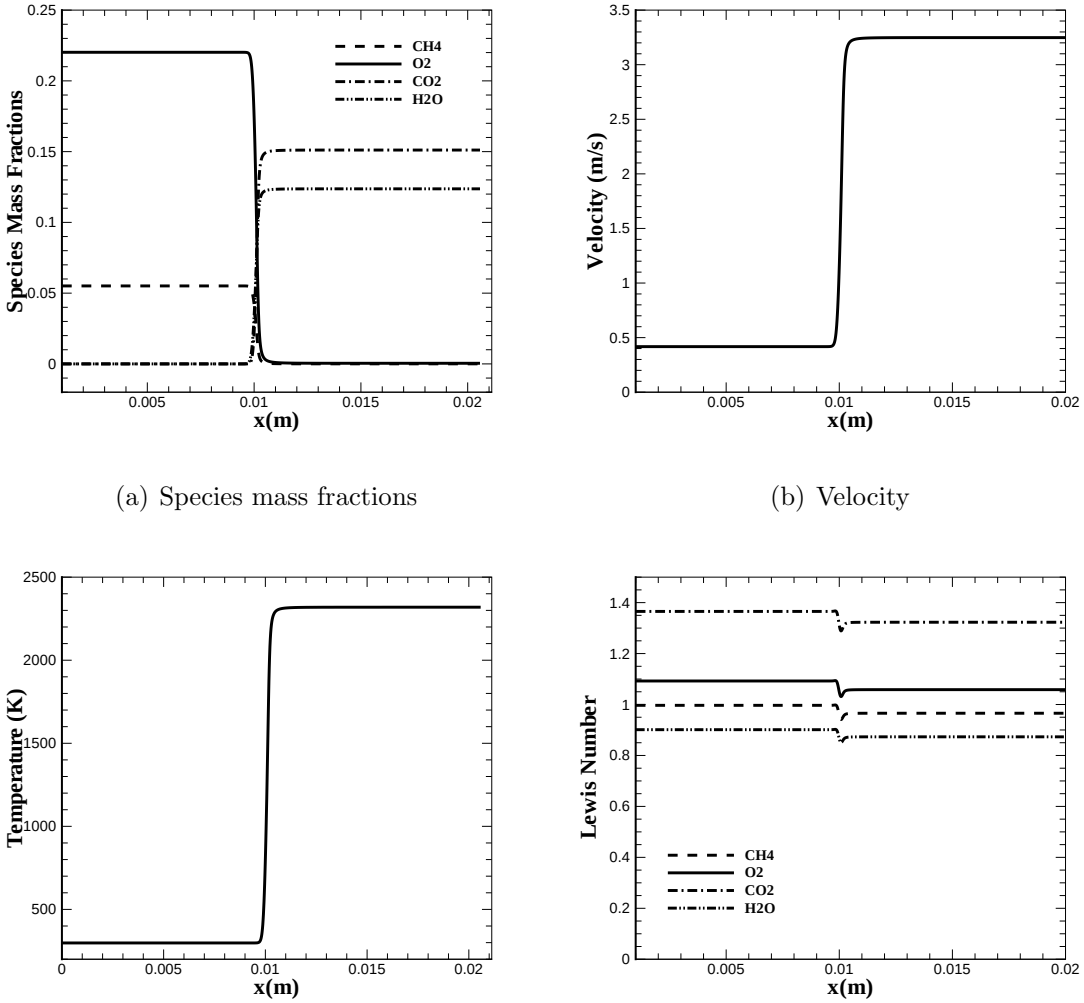
5.4 Verification of Chemical Kinetics

Partial verification of the implementation of the chemical kinetics and thermodynamic and transport models used herein for describing methane-air combustion has been carried out by performing laminar flame speed and flame structure computations for a one-dimensional steady-state premixed flame. Note that similar one-dimensional premixed flame calculations were performed in earlier work of Northrup [94]. In particular, Northrup considered the six-species two-step chemical kinetics scheme of methane-air combustion. The predictions were in good agreement with the predicted

values of a 17-species, 58-reaction scheme provided by CHEMKIN. As noted earlier in Chapter 2, a five-species one-step reduced chemical-kinetics scheme for methane oxidization is used for all of the reactive flow calculations described in this thesis. The structure of the stoichiometric premixed flame and the laminar flame speed was determined for this one-step chemical kinetics model and compared to the values obtained using the CHEMKIN program PREMIX.

The computational domain consists of 100×2 cells on a solution domain of $0.02\,\mathrm{m} \times 0.0002\,\mathrm{m}$ with a mesh clustered near the flame front located at the center of the domain. Initially, a stoichiometric mixture of premixed fuel and air was established on one side of the computational domain and the burnt products on the other side. The inlet and outlet boundary velocity and pressure were then adjusted to ensure the constant mass flux throughout domain (see Northrup [94] for details on the boundary conditions). The Roe flux function with the Venkatakrishnan limiter was used together with a four-stage optimally smoothing time-marching method and a CFL number of 0.25 to achieve a steady-state solution. Low-Mach number preconditioning was also employed for convergence acceleration and to avoid introducing excessive numerical dissipation for this low-Mach-Number flow.

The numerical predictions of the laminar flame structure is well represented by the profiles of the mass fraction, velocity, and temperature of Figure 5.19. Figures 5.19(a)–5.19(c) show the variation of these quantities through the flame. For the five-species, one-step model, the predicated laminar flame speed was about 45 cm/s and the flame temperature was 2300 K. Both values are slightly higher than the laminar flame speed and flame temperature provided by CHEMKIN, which were 41 cm/s and 2234 K, respectively. In general, the predictions of the one-step model are are thought to be acceptable considering the simplified one-step chemical kinetics. Figure 5.19(d) shows the small variations in Lewis number for each of the species across the flame. This also agrees with published data [11] and provides additional confidence in the modelling of mixture transport properties used herein.



(a) Species mass fractions

(b) Velocity

(c) Temperature

(d) Lewis number

Figure 5.19: Predicted profiles for a stoichiometric, $\phi = 1.0$, premixed one-dimensional methane and air flame solution.

Chapter 6

Numerical Results

This chapter considers the application of the proposed parallel AMR scheme to the prediction of both two-dimensional (axisymmetric) and three-dimensional turbulent non-reacting flows and turbulent non-premixed flames. The numerical simulations were performed for a bluff-body burner and, for the reacting cases, gaseous methane fuel was considered. In each case, the numerical predictions are compared to the experimental data provided by the International Workshop on Measurement and Computation of Turbulent Non-premixed Flames [182].

As for the results discussed in Chapter 5, the parallel AMR scheme was implemented on a parallel cluster of 4-way Hewlett-Packard ES40, ES45, and Integrity rx4640 servers with a total of 244 Alpha and Itanium 2 processors. The interconnection between the servers in the cluster was achieved using a low-latency Myrinet network and switch. All of the numerical results reported here were obtained using this parallel cluster.

6.1 Bluff-Body Burner Flows

6.1.1 Bluff-Body Burner Flow Geometry

The International Workshop on Measurement and Computation of Turbulent Non-premixed Flames [182] has established an internet library of well-documented experi-

mental database for turbulent non-premixed flames that are appropriate for combustion model verification and validation. The Sydney bluff-body burner configuration that forms part of this experimental database has data available for both non-reacting and reacting cases. The configuration for the Sydney bluff-body burner is shown in Figure 6.1. The bluff-body has a diameter of $D_b = 50$ mm and is located in a co-axial flow of air. Various gases can be injected through an orifice of diameter 3.6 mm at the base of the cylindrical bluff body. The bluff-body stabilized flames have a recirculation zone close to the base of the bluff body. This burner configuration produces a relatively extensive and complex turbulent field and causes intense mixing between the reactants and combustion products. The stabilization mechanisms resemble those of industrial combustors and yet the boundary conditions for the bluff-body flames are simple and well-defined, making them well suited for investigating in great detail the capabilities of models for turbulent non-premixed diffusion flames.

The Sydney bluff-body burner has been investigated and/or used for verification and validation purposes in several recent studies (e.g., Dally *et al.* [183–189] and Turpin and Troyes [190]). In this thesis research, the proposed parallel AMR algorithm has been applied to the solutions of two non-reacting flow cases and one reacting flow case. The three cases investigated herein are a non-reacting (cold) air flow case for understanding the re-circulating flow field structure at the base of the burner; a non-reacting (cold) mixing flow case for analyzing the mixing of the fuel and co-flow streams; and finally, a reacting (hot) case. In the cold non-reacting bluff-body burner flow case, air is injected through the orifice at the base of the cylindrical bluff body with a temperature of 300 K and a bulk velocity of 61 m/s. The bulk velocity and temperature of the co-flowing air are 20 m/s and 300 K, respectively. The Reynolds and Mach numbers based on the high-speed jet are $\mathrm{Re} = 193,000$ and $\mathrm{Ma} = 0.18$. For the mixing (ethylene jet) case, ethylene (C_2H_4) is injected at the base of the bluff-body with a bulk velocity of 50 m/s and a temperature of 300 K. The velocity and temperature of the co-flowing air are the same as the cold air case. In the mixing case, the Reynolds and Mach numbers based on the ethylene flow are $\mathrm{Re} = 145,000$ and $\mathrm{Ma} = 0.11$. For the reacting case, methane (CH_4) is injected through the orifice at the base of the cylindrical bluff body with a temperature of 300 K. The bulk velocities

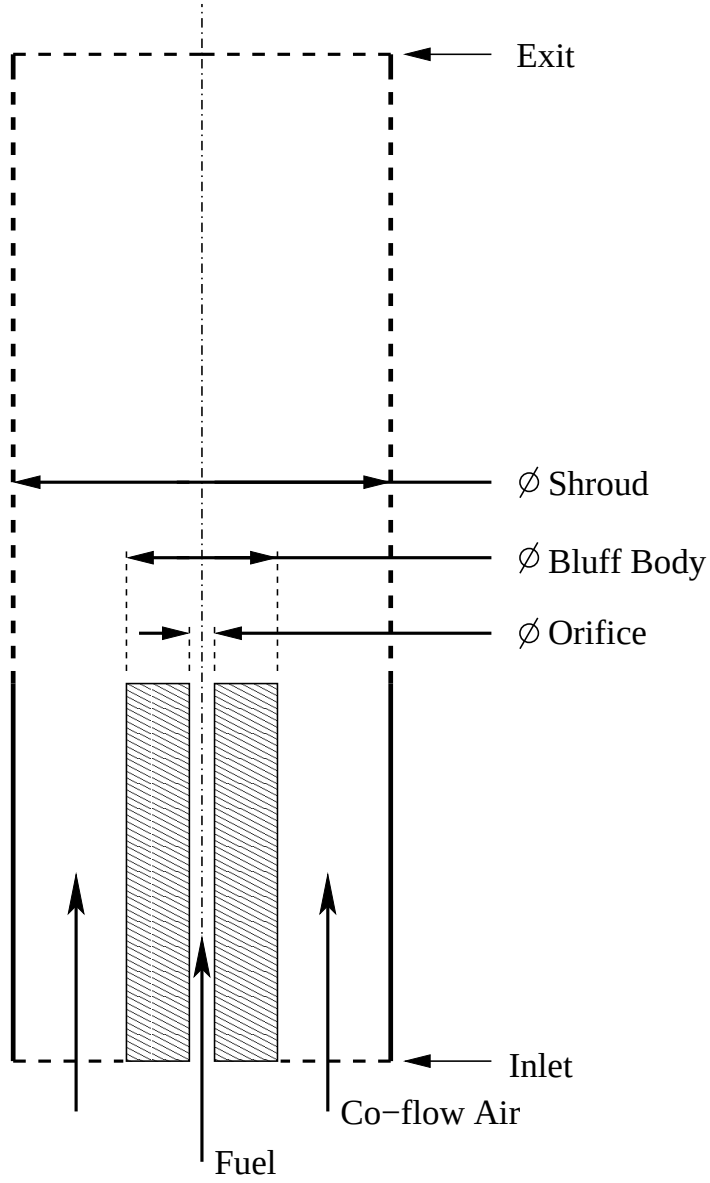


Figure 6.1: Schematic of the Sydney bluff-body burner showing the fuel jet, co-flow, and bluff-body geometry.

of the co-flowing air and methane fuel are 25 m/s and 108 m/s, respectively. The Reynolds and Mach numbers of the methane jet are Re$=315,000$ and Ma$=0.24$.

6.1.2 Boundary Conditions

At the inflow planes of the computational domain for both the fuel jet and the co-flow air, Dirichlet conditions are used for flow quantities such as density, velocities, turbulent kinetic energy, the dissipation rate of the turbulent kinetic energy, and species concentrations. A Neumann condition is used for the pressure. The symmetry condition is applied on both the centre line and the shroud boundary. A no-slip boundary condition is applied to the wall of the annulus pipe for the co-flow air. At the outflow plane, the ambient pressure is specified and Neumann type boundary conditions are applied for all other flow quantities. The co-flow air boundaries were set wide enough to include the full boundary layer on the co-flow side and to ensure that the boundary specifications do not influence the jet. The diameter of the co-flow inlet pipe is about 3 D_b and the axial length of the computational domain is 6 D_b, where D_b is the bluff-body diameter.

In order to intentionally avoid sharp changes in velocity for computational purposes, a power law is used to specify the initial axial velocity profiles in both the jet and the co-flow and thereby approximate a fully-developed turbulent pipe flow. Additionally, the initial radial velocities are taken as zero [182]. The axial distance of the annulus pipe was extended further upstream to ensure that fully developed pipe flow conditions prevail at the exit plane of the annulus pipe. Note that in the case of the three-dimensional simulations, the flow fields were initialized with the steady-state numerical solutions obtained from the two-dimensional calculations.

All three of the cases described above were solved in a two-dimensional axisymmetric coordinate frame, and two of these cases, the cold-air flow case and the hot case, were also considered for a general three-dimensional coordinate frame. The three-dimensional calculations were performed on a quarter section of the bluff-body burner as shown in Figure 6.2. In what follows, the predicted flow fields for the bluff-body burner in both two dimensions and three dimensions are analyzed and compared to the available experimental data.

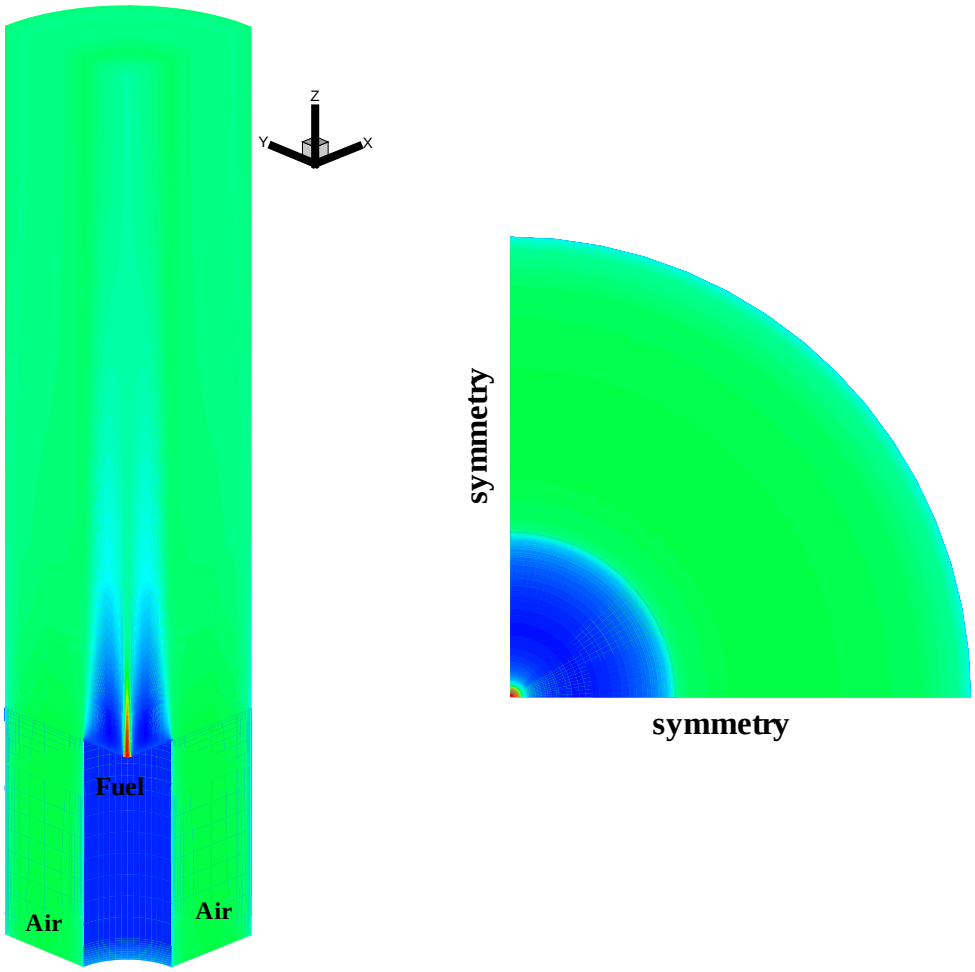


Figure 6.2: Depiction of the quarter section geometry used in three-dimensional numerical simulations of Sydney bluff-body burner.

6.2 Axisymmetric Turbulent Diffusion Flame

6.2.1 Non-Reacting Cold Flow

The first case considered is the two-dimensional axisymmetric simulation of the non-reacting cold flow. In this case, the air exits the fuel jet at a bulk velocity of 61 m/s into a co-flow of air flowing at 20 m/s. These flow conditions are classified as "jet-dominant", since the jet penetrates the re-circulation zone behind the wall of the bluff-body and propagates in a jet-like manner further downstream. The solutions

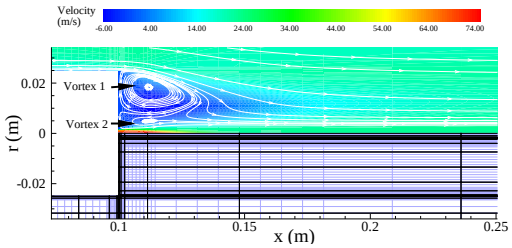


Figure 6.3: Predicted mean axial velocity field of the bluff-body for non-reacting bluff-body burner with air jet with 53 16×16 cell blocks (13,568 cells).

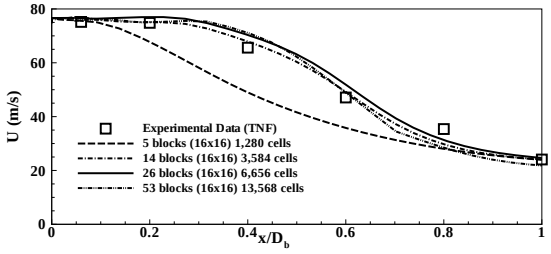
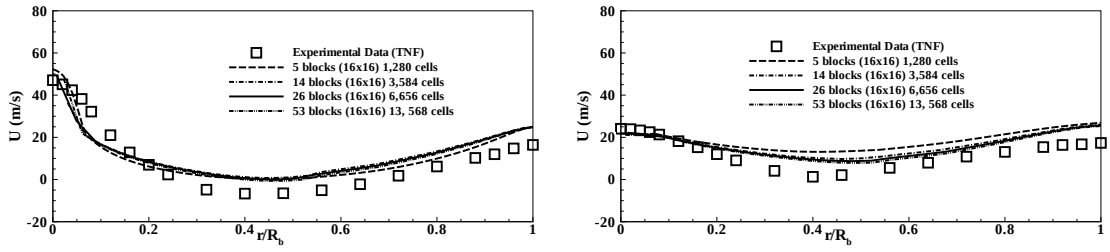


Figure 6.4: Comparison of predicted and measured on-axis axial profiles of the mean axial velocity component downstream from the base of the bluff-body burner for non-reacting flow with air jet.

were determined for a sequence of refined meshes to establish the grid-convergence of the solution. In particular, the flow-field predictions were carried out on a sequence of four adaptively refined grids, each consisting of a number of 16×16 cell blocks: 5 blocks (1,280 cells); 14 blocks (3,584 cells); 26 blocks (6,656 cells); and 53 blocks (13,568 cells). The refinement efficiency for the three sequential fine grids was 0.53, 0.675, and 0.834. The mesh resolution was such that the typical size of the computational cells nearest the wall was in the range $0.2 < y^+ < 1$.

The numerical predictions for this case and results of the refinement study are shown in Figures 6.3–6.8. It is apparent that the majority of the solution values do not change significantly as the mesh is refined from 6,656 cells to 13,568 cells. For this reason it can be argued that the final numerical solution on the finest mesh is virtually independent of the grid. Figure 6.3 shows the predicted mean axial velocity and streamlines and reveals the formation of a double-vortex structure in the re-circulation zone. The two vortices are important in controlling fuel/oxidizer mixing. The calculations indicate that the re-circulation zone extends to $x/D_b \approx 0.8$. This is slightly less than the experimentally observed value of $x/D_b \approx 1.0$. The agreement between the predictions and the experiments is further confirmed by a comparison of the predicted axial (center-line) profile of the mean axial velocity component to the experimental results as depicted in Figure 6.4 and by comparisons of the predicted radial profiles of the mean axial velocity component at both $x/D_b = 0.6$ and $x/D_b =$



(a) $x/D_b = 0.6$ (b) $x/D_b = 1.0$

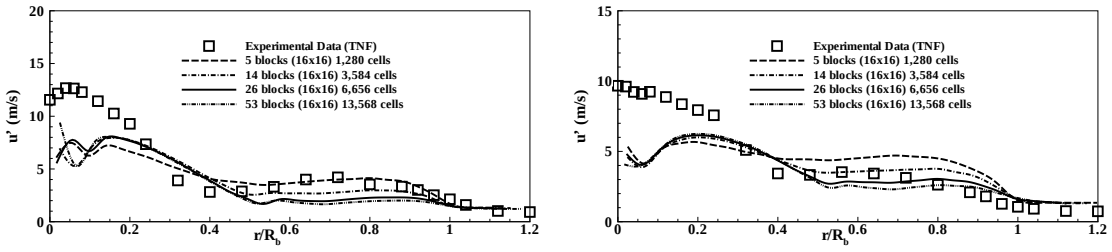
Figure 6.5: Comparison of predicted and measured radial profiles of mean axial velocity at various locations downstream from the base of the bluff-body burner for non-reacting flow with air jet.

1.0 downstream from the base of the bluff-body to the measured data as shown in Figures 6.5(a) and 6.5(b). For each velocity profile, rather good agreement between the numerical results and the experiments can be observed.

Figures 6.6 and 6.7 show a comparison of the root mean square (RMS) fluctuations of the velocity components at the two downstream stations $x/D_b = 0.6$ and $x/D_b = 1.0$. The predicted and measured specific Reynold stress $\overline{u'v'}$ profiles are also compared in Figure 6.8. It can be seen that there are under- and/or over-predicted regions ($r/R_{\rm b} < 0.2$). These regions encompass the inner vortex and the vicinity of the outer vortex of a double-vortex structure in the re-circulation zone. Re-circulation zones with complex turbulent structures are quite sensitive to the turbulence modelling and a variety of RANS simulations have addressed this sensitivity to the turbulence model and/or combustion models [188, 191–193]. The overall agreement between the numerical solution and the experimental data for these turbulence quantities is quite reasonable and is comparable to other results reported in the literature [188, 190, 192].

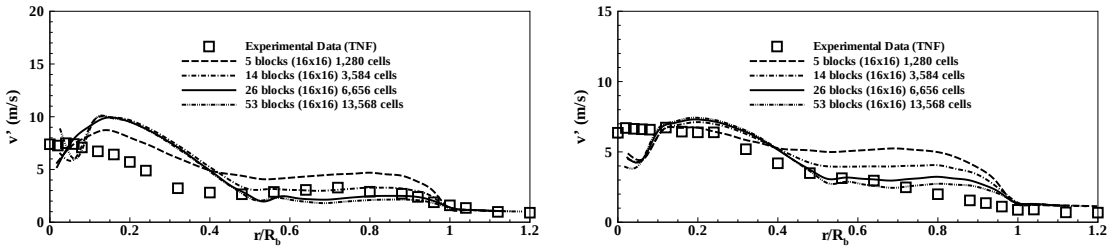
6.2.2 Mixing Flow

Numerical results for the two-dimensional axisymmetric simulations of the mixing-flow bluff-body burner problem are considered next. Numerical results for the cold flow with an ethylene fuel jet are depicted in Figure 6.9, where the predicted mass



(a) $x/D_b = 0.6$ (b) $x/D_b = 1.0$

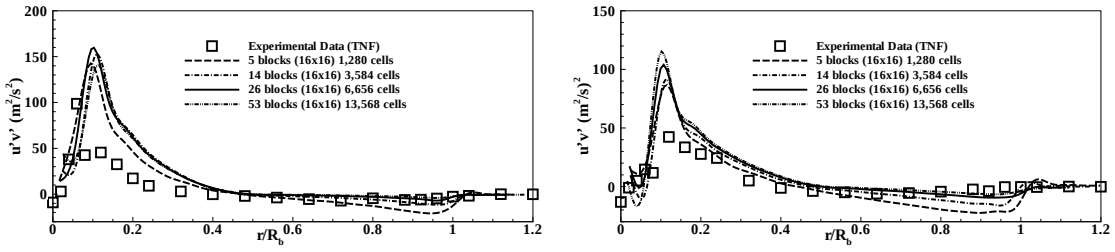
Figure 6.6: Comparison of predicted and measured radial profiles of $\sqrt{\overline{u'^2}}$ at various locations downstream from the base of the bluff-body burner for non-reacting flow with air jet.



(a) $x/D_b = 0.6$ (b) $x/D_b = 1.0$

Figure 6.7: Comparison of predicted and measured radial profiles of $\sqrt{\overline{v'^2}}$ at various locations downstream from the base of the bluff-body burner for non-reacting flow with air jet.

fraction of C_2H_4 obtained using a mesh consisting of 479 6×6 cell blocks (17,244 cells), with five levels of refinement, is compared to measured C_2H_4 concentrations. The mesh resolution was again such that the typical size of the computational cells nearest the wall was in the range $0.2 < y^+ < 1$. Qualitatively and quantitatively, the numerical results appear to be quite reasonable when compared to the experimental data. Additional quantitative comparisons of the numerical on-axis axial and radial distributions (at $x/D_b = 0.6$ and $x/D_b = 1.0$) of the C_2H_4 mass fraction to experimental values are given in Figures 6.10 and 6.11. Figure 6.10 shows the center-line mass fraction of C_2H_4 as a function of the axial distance and indicates that the overall



(a) $x/D_b = 0.6$ (b) $x/D_b = 1.0$

Figure 6.8: Comparison of predicted and measured radial profiles of $\overline{u'v'}$ at various locations downstream from the base of the bluff-body burner for non-reacting flow with air jet.

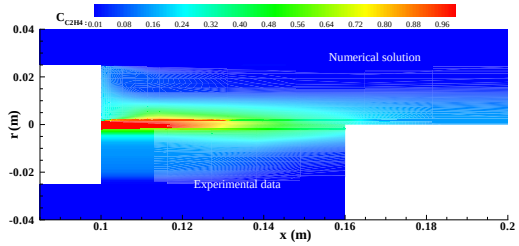


Figure 6.9: Mean C_2H_4 mass fraction for non-reacting flow field of bluff-body burner.

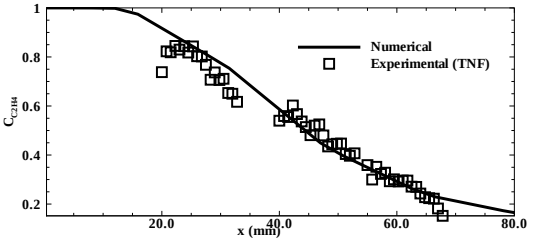
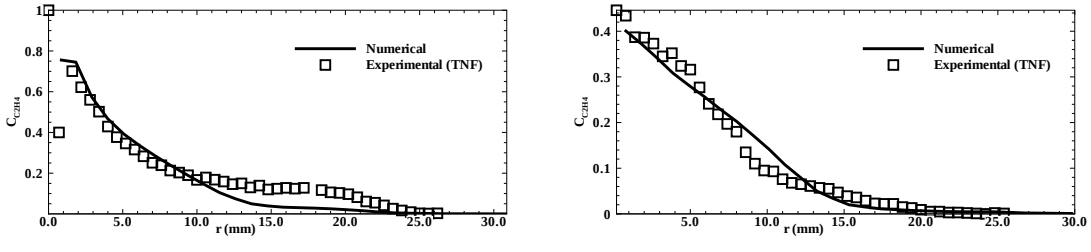


Figure 6.10: Comparison of predicted and measured on-axis axial profiles of mean C_2H_4 mass fraction downstream from the base of the bluff-body burner for non-reacting flow with C_2H_4 jet.

agreement is rather good, except for a slight over-prediction of the mass fraction near $x =$ 25-30 mm, which is close to the inner vortex of the double-vortex structure. The mass fraction is also slightly under-predicted near $r =$ 15-20 mm in Figure 6.11(a), and over-predicted near $r =$ 8-12 mm in Figure 6.11(b). Dally *et al.* [188] indicate that numerical predictions of the mixture fraction show strong sensitivity to the turbulence modelling in these regions. In general, it is felt that the present numerical solutions of the mixing field indicate that the fuel and oxidizer mixing process is in fact quite well reproduced by the two-equation RANS turbulence modelling approach adopted in this thesis research.



(a) $x/D_b = 0.6$ (b) $x/D_b = 1.0$

Figure 6.11: Comparison of predicted and measured radial profiles of mean C_2H_4 mass fraction at various locations downstream from the base of the bluff-body burner for non-reacting flow with C_2H_4 jet.

6.2.3 Reacting Hot Flow

The final set of two-dimensional axisymmetric simulations considers a reacting flow case for the bluff-body burner with methane fuel injected into the flow at the base of the bluff body. The five-species, one-step, reduced kinetic scheme described in Chapter 2 is used in this case to represent the oxidation of the methane fuel. As with the cold flow case, the computations were carried out on a sequence of adaptively refined grids in order to assess the grid convergence of the solutions. Three computational grids were considered with each grid consisting of a number of 16×16 cell blocks: 28 blocks (1,792 cells); 43 blocks (2,752 cells); and 55 blocks (3,520 cells). The refinement efficiencies for the medium and fine grids were 0.616 and 0.8772, respectively. Also in this case, the mesh resolution was such that the typical size of the computational cells nearest the wall was in the range $0.2 < y^+ < 1$.

Figures 6.12(a) and 6.12(b) show the predicted distributions of mean mass fraction of CO_2 and mean temperature for the turbulent non-premixed bluff-body burner flame. The predicted flame structure is generally in agreement with the experimentally observed structure. The flame is quite elongated and three zones can be identified: the re-circulation, neck, and jet-like propagation zones. A vortex structure is formed in the re-circulation zone and acts to stabilize the flame. The maximum flame temperature is about 2180 K. The predicted values of the mean temperature, 1620 K, and mass fraction of CO_2, 0.116, at location of ($x/D_b = 1.92$, $r/R_b = 0.4$) can

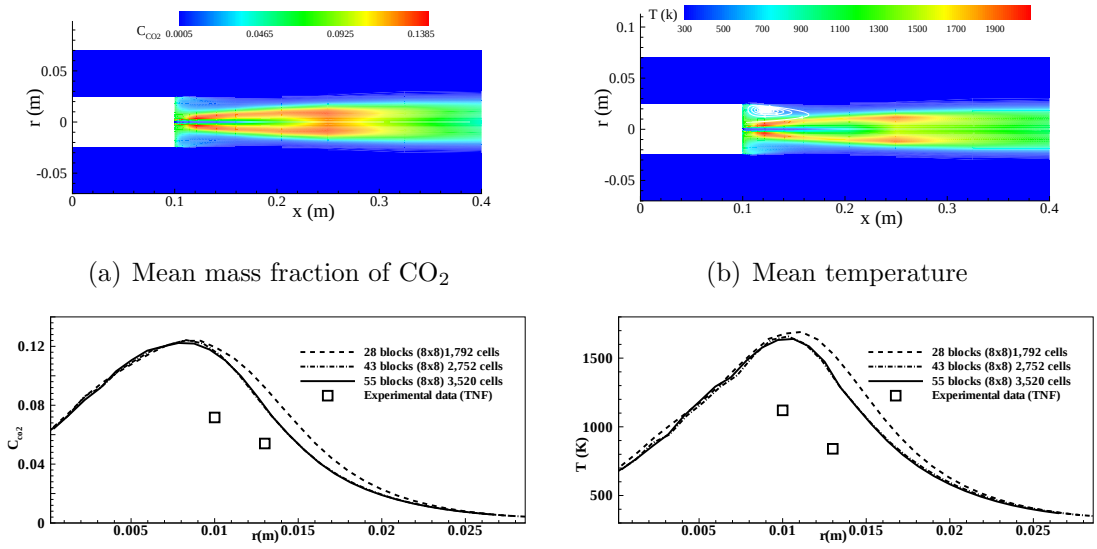
be compared to the measured values of the flame temperature, 1120 K, and CO_2 concentration, 0.07. At the location of ($x/D_b = 1.92$, $r/R_b = 0.52$), the computed mean temperature, 1380 K, and mass fraction of CO_2, 0.076, are over predicted compared to the measured values of the flame temperature, 810 K, and CO_2 concentration, 0.055. The results of the mesh refinement study are shown in Figures 6.12(c) and 6.12(d), and the majority of the solution values, such as axial velocity, temperature, and major species CO_2, do not change appreciably as the mesh is refined from 2,752 cells to 3,520 cells, providing confidence that a grid independent solution has been achieved.

It is noted above that the temperature and hence carbon dioxide concentration are somewhat over-predicted by the proposed parallel AMR scheme for this case. However, it is felt that the agreement with the experimental values is actually quite reasonable considering the limitations of the simplified reduced chemical kinetics scheme and turbulence/chemistry interaction model used herein, as well as the fact that radiation transport has not been taken into account in the simulation. Note that while radiation effects may influence the predicted temperature for this case; Merci *et al.* argue that, since the flame is unconfined and very little soot is formed, radiation effects should be relatively small [192, 194].

6.2.4 Multigrid Acceleration

The numerical solutions for both the cold and hot cases of the two-dimensional axisymmetric bluff-body burner flows were obtained using the preconditioned multigrid technique described in Section 3.3. The 3-level V-cycle preconditioned multigrid with a 3-stage optimally smoothing scheme was employed. A prolongation operator, based on a cell-aspect-ratio sensor as discussed in Section 5.2.3, was also applied. For those cells with aspect ratios greater than a value of 1000, a simple injection was used. Standard bi-linear interpolation was employed for cells with aspect ratios smaller than this value. The CFL number was 0.2.

Figures 6.13(a) and 6.13(b) indicate the preconditioned multigrid acceleration for the non-reacting and reacting cases, respectively. Both the multigrid and precondi-



(a) Mean mass fraction of CO_2

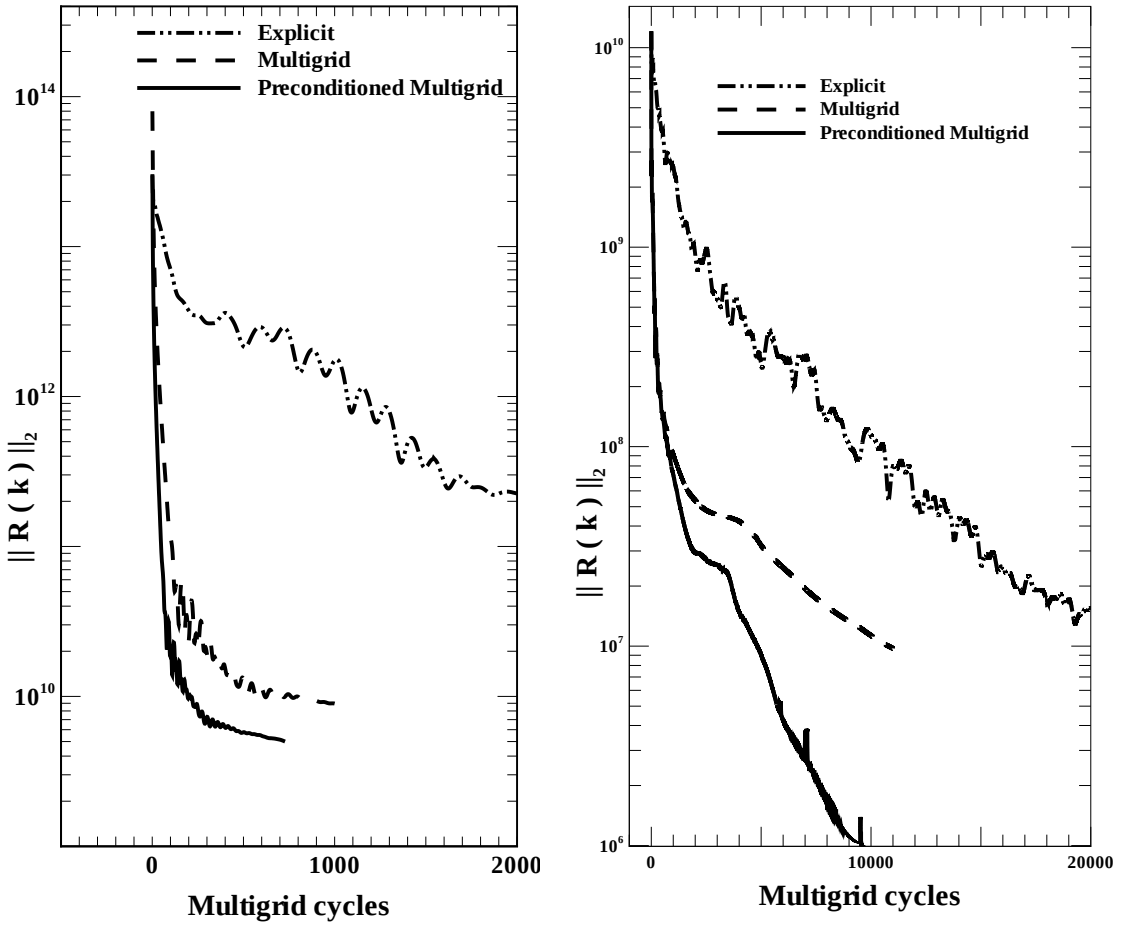
(b) Mean temperature

(c) Mean mass fraction of CO_2 radial profile

(d) Mean temperature radial profile

Figure 6.12: Predicted mean mass fraction of CO_2 and mean temperature contours and comparison of predicted radial profiles of mean mass fraction of CO_2 and mean temperature to the measured data at $x/D_b = 1.92$ downstream from the base of the bluff-body burner for reacting flow with CH_4 jet.

tioned multigrid algorithms speed up the convergence rates to the steady-state solutions quite significantly for both cases as compared to the convergence rate achieved using the semi-implicit time-marching method alone (i.e., smoother alone without the multigrid procedure). The preconditioned multigrid seems to have a more positive effect for the reacting case than for the non-reacting problem. Although the convergence rate slows after the residual has been reduced by 4–5 orders of magnitude, and this slow down may be associated with limitations of the proposed multigrid method when AMR is used, overall it is felt that satisfactory convergence rates have been achieved. The preconditioned multigrid provides a speedup of about two in terms of CPU time over regular multigrid for both cases.



(a) non-reacting (b) reacting

Figure 6.13: Convergence features of preconditioned multigrid for cold and hot bluff-body burner flows.

6.2.5 Parallel Performance

The parallel performance of the proposed solution-adaptive algorithm was assessed for the two-dimensional turbulent pipe flow problem in the previous chapter. Here, the parallel performance of the proposed algorithm, in terms of its strong scaling and parallel efficiency, is further evaluated for a fixed-size turbulent bluff-body diffusion flame problem having 112 solution blocks with two different mesh sizes: 28,672 cells (112 16×16 cell blocks) and 64,512 cells (112 24×24 cell blocks). The performance

was assessed using up to 64 processors. An added difference of this parallel performance assessment from the previous one is that the 3-level V-cycle preconditioned multigrid technique was used in the computations, whereas the previous estimation of the parallel performance was carried out for the semi-implicit time-marching scheme alone.

Figure 6.14 shows the parallel speedup (strong scaling) for this case. It can be observed that the proposed scheme provides a nearly linear speedup and is about 76% efficient for up to 64 processors using the larger 24×24 cell solution blocks. For the smaller 16×16 cell solution blocks, the parallel efficiency drops to 68%. Compared to the estimation shown in Figure 5.15, the parallel efficiency is somewhat reduced. The performance is affected by the coarse grid calculations of the multigrid algorithm. In particular, the number of inter-block messages on the coarse meshes is often nearly the same as that on the fine meshes, thereby decreasing the ratio of computation to communication and adversely affecting the parallel performance. For three-dimensional applications, it is felt that it will be generally easier to keep the computation work to communication overhead ratio high using the proposed parallel AMR approach and thereby maintain high parallel efficiency.

6.3 Three-Dimensional Turbulent Diffusion Flame

This section provides the fully three-dimensional simulations of the bluff-body burner, for which the two-dimensional axisymmetric numerical results were discussed above. Three-dimensional simulations of both non-reacting and reacting bluff-body burner flows have been considered.

A cylindrical-shaped computational domain of the bluff-body burner can be used for the three-dimensional simulations as shown in Figure 6.15. However, to reduce the computational expense, a quarter section mesh of the bluff-body configuration as illustrated in Figure 6.16 was used. Figure 6.17 depicts the mesh topology near the fuel inlet for this case. The three-dimensional numerical results were obtained using an explicit 3-stage optimally-smoothing time marching scheme with a CFL number of 0.1. Typically, the momentum residual reduction was found to be about

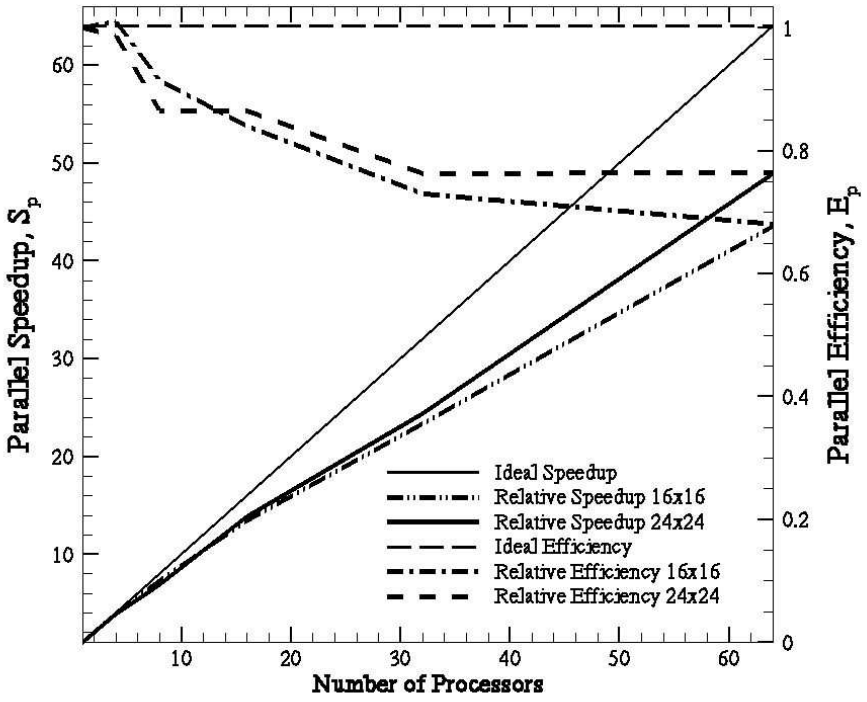


Figure 6.14: Parallel speedup (strong scaling) and efficiency for computation of Sydney bluff-body burner two-dimensional flame problem with 3-level V-cycle preconditioned multigrid using up to 64 processors.

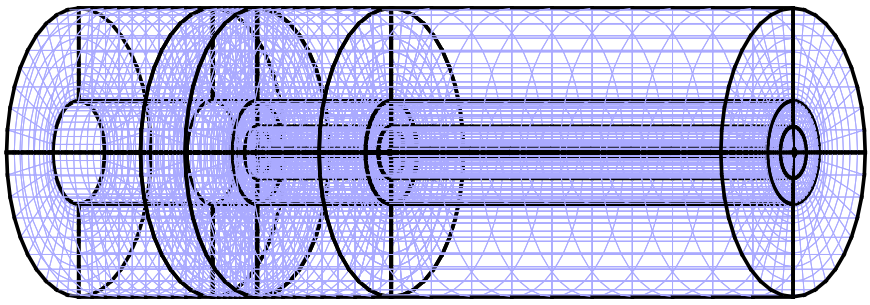


Figure 6.15: An illustration of the mesh for the complete bluff-body burner configuration.

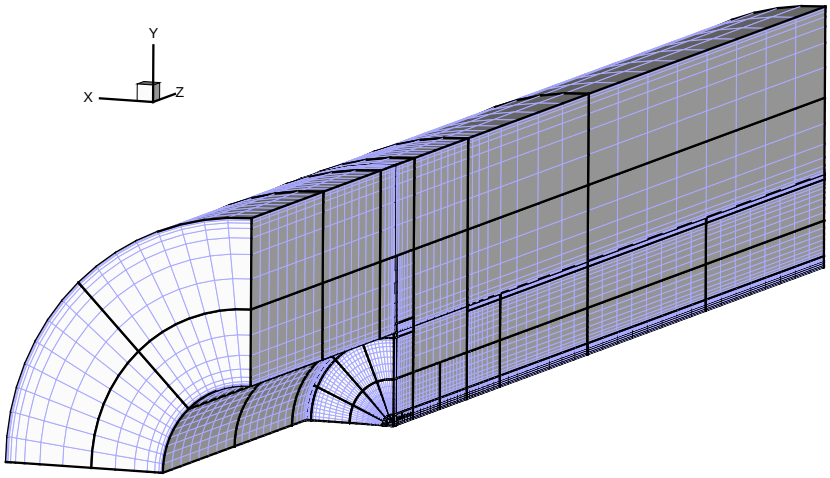


Figure 6.16: A quarter of the bluff-body burner mesh with 392 (8×8×8) cell blocks (200,704 cells).

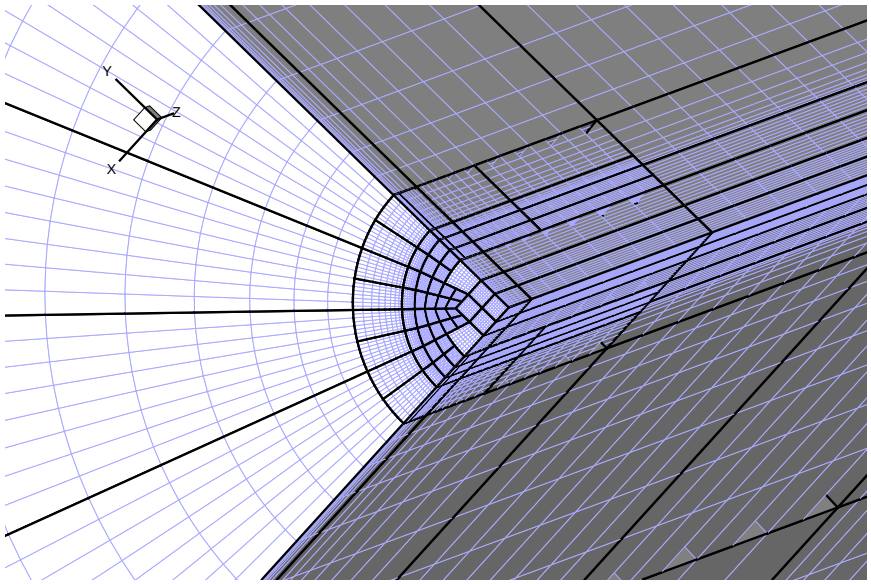


Figure 6.17: A magnified view of the mesh near the fuel inlet.

3–4 order of magnitude within 200,000–250,000 iterations for all the three-dimensional cases considered here. Note that the multigrid algorithm is not employed for three-dimensional calculations.

6.3.1 Non-Reacting Cold Flow

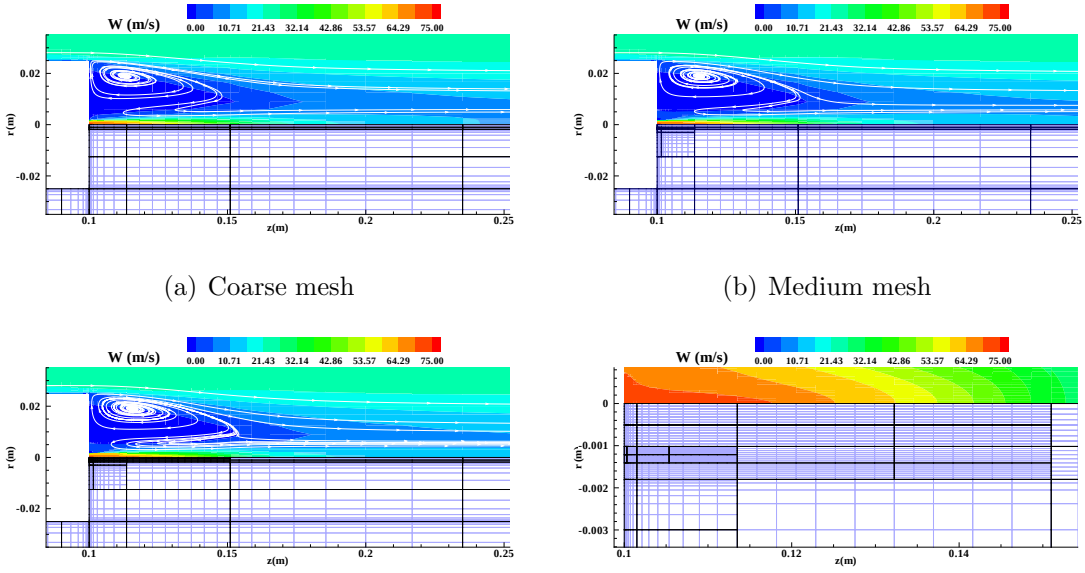
As with the axisymmetric simulations, the three-dimensional computations were carried out on a sequence of three adaptively refined grids, each consisting of a number of $8 \times 8 \times 8$ cell blocks: 56 blocks (28,672 cells); 84 blocks (43,008 cells); and 140 blocks (71,680 cells). In this way, grid convergence of the numerical solution could be assessed. The cells of the computational grid are clustered in regions of strong gradients of the mean mixture solution quantities, outer and inner shear layers, and re-circulation zones near the walls. The mesh resolution typically provides for off-wall spacings for the first cells nearest the wall in the range of $0.47 < y^+ < 1.2$.

Figures 6.18(a)–6.18(d) show colour contours of the predicted mean axial velocity with superimposed mean streamlines in the xz-plane ($y = 0$) obtained from coarse, medium and fine meshes for the three-dimensional cold-flow solutions of the bluff-body burner. The figures reveal the formation of a vortex structure in the re-circulation zone which is important in controlling fuel/oxidizer mixing. Th flow structure is qualitatively and quantitatively very similar to the two-dimensional findings described in previous section of this chapter. Clearly, Figure 6.18(c) indicates that the re-circulation zone extends to $z/D_b \approx 1.0$ as the mesh was refined. This is very close to the experimentally observed value.

Figures 6.19(a)–6.22(b) compares numerical distributions of mean axial velocity, the intensity of velocity fluctuations, and the Reynolds shear stress, $\overline{v'w'}$, to experiments. Moreover, the results of the mesh refinement study are also shown in these figures. From these profiles, it can be seen that the majority of those numerical solution values do not change appreciably as the mesh is refined from the 43,008 cells to 71,680 cells. This indicates that the numerical solution is converging toward a grid-independent result.

Figure 6.19(a) depicts the axial (center-line) profile of the mean axial velocity

component and a close-up of the region $0 < z/D_b \leq 1.0$ is shown in Figure 6.19(b). In addition, Figures 6.19(c) and 6.19(d) show comparisons of radial profiles of the mean axial velocity component at $z/D_b = 0.6$ and $z/D_b = 1.0$ downstream from the base of the bluff-body burner. The predicted mean axial velocity on the axis is slightly over-predicted for $0 < z/D_b < 1.0$; otherwise, good agreement between the numerical solutions and experiment can be observed.



(a) Coarse mesh

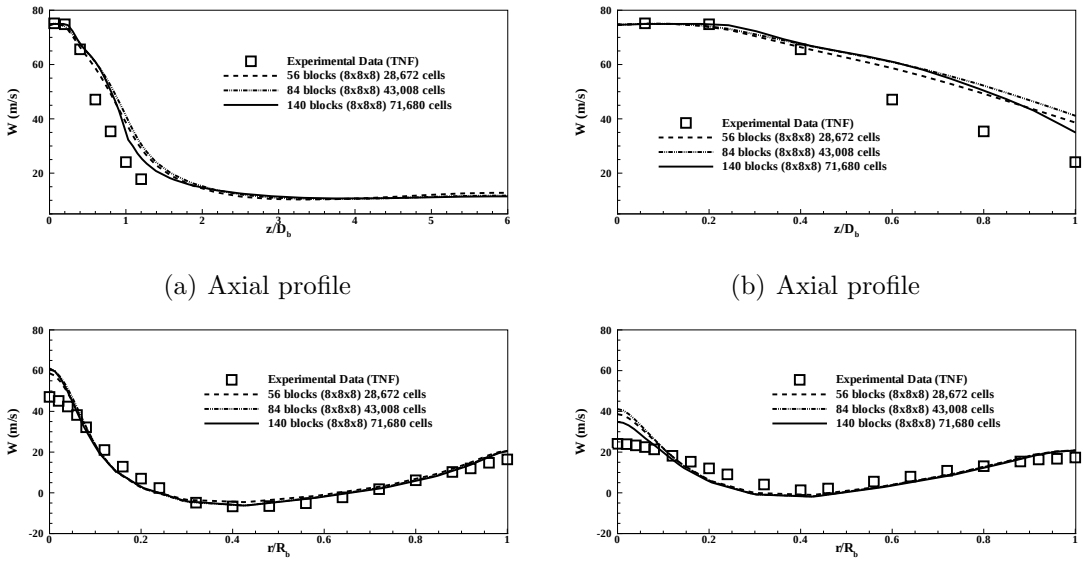
(b) Medium mesh

(c) Fine mesh

(d) A zoom-in of fine mesh

Figure 6.18: Predicted mean axial velocity contours on coarse, medium and fine meshes: (a) coarse mesh consists of 56 solution blocks ($8 \times 8 \times 8$) 28,672 cells, (b) medium mesh consists of 84 solution blocks (8×8×8) 43,008 cells of 2-level refinement with a refinement efficiency of 0.8125; (c) fine mesh consists of 140 solution blocks (8×8×8) 71,680 cells of 3-level refinement with a refinement efficiency of 0.9609, and (d) shows the zoom-in view of the refined mesh in the region close to the fuel jet and bluff-body solid wall in the fine mesh.

Figures 6.20 and 6.21 show comparisons of the predicted and measured values of the RMS fluctuations in the velocity components (axial and radial velocity components). The predicted intensity of the fluctuations in the axial velocity component is under-predicted overall, while the predicted intensity of fluctuations in the radial velocity component is generally over-predicted, as compared to the experimental data.



(a) Axial profile

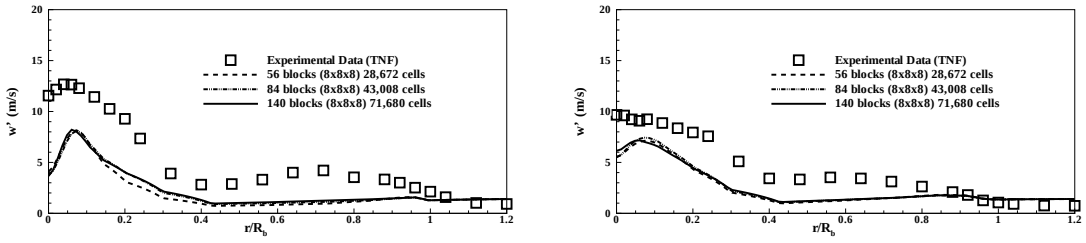
(b) Axial profile

(c) $z/D_b = 0.6$

(d) $z/D_b = 1.0$

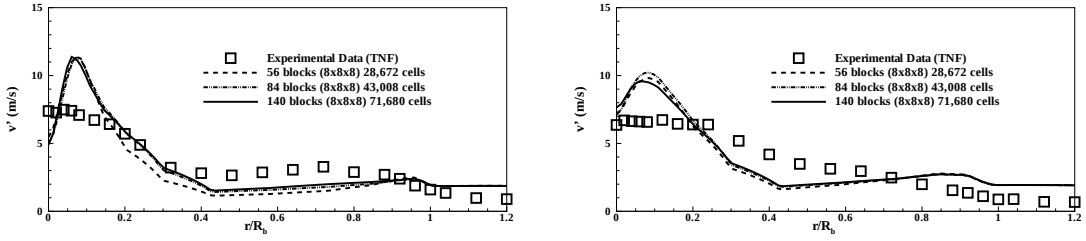
Figure 6.19: Predicted mean axial velocity radial profiles at $z/D_b = 0.6$ and $z/D_b = 1.0$ downstream from the base of the bluff-body burner for non-reacting flow with air jet.

The computed and measured specific Reynolds stress $\overline{v'w'}$ profiles are also compared in Figure 6.22. The numerical values of $\overline{v'w'}$ are somewhat under-predicted in the relatively high mean-velocity regions, but closer to the measured data in the low mean-velocity regions. It can be seen that there are under- and/or over-predicted regions close to the center-line and the solid wall boundaries. These regions either encompass or are in the vicinity of the re-circulation zone. As noted earlier, re-circulation zones with complex turbulent structures are quite sensitive to the turbulence modelling. A variety of RANS simulations have addressed the sensitivity of the results to the turbulence model and/or combustion models [95, 96, 188, 191–193]. The overall agreement between the predicted results from the current study and the experimental data is thought to be quite reasonable and is comparable to other similar results found in the literature [188, 190, 192].



(a) $z/D_b = 0.6$ (b) $z/D_b = 1.0$

Figure 6.20: Comparison of predicted and measured radial profiles of $\sqrt{\overline{w'^2}}$ at various locations downstream from the base of the bluff-body burner for non-reacting flow with air jet.

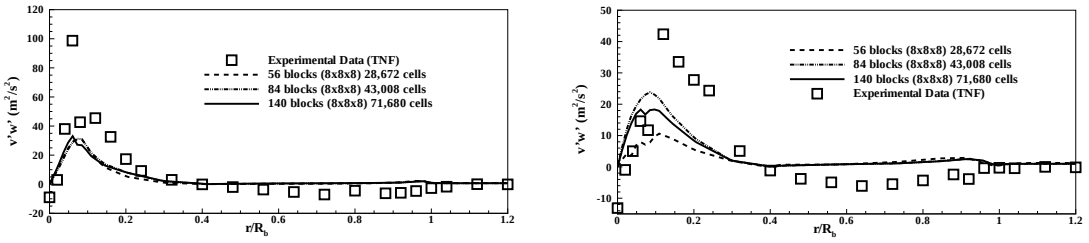


(a) $z/D_b = 0.6$ (b) $z/D_b = 1.0$

Figure 6.21: Comparison of predicted and measured radial profiles of $\sqrt{\overline{v'^2}}$ at various locations downstream from the base of the bluff-body burner for non-reacting flow with air jet.

6.3.2 Reacting Hot Flow

As a final case, a fully three-dimensional simulations of the reactive bluff-body burner flow with methane fuel injection was considered. A quarter section of the three-dimensional grid was again used. Reactive flow-field predictions using the same chemical kinetics as in the axisymmetric simulations have been performed on a sequence of three successively refined meshes, with each grid consisting of a number of 8×8×8 cell blocks. The three computational grids consist of 56 blocks (28,672 cells), 133 blocks (68,096 cells), and 210 blocks (107,520 cells), respectively. The refinement efficiency for the grids ranged from 0.7 to 0.9414. The sequence of the refined meshes



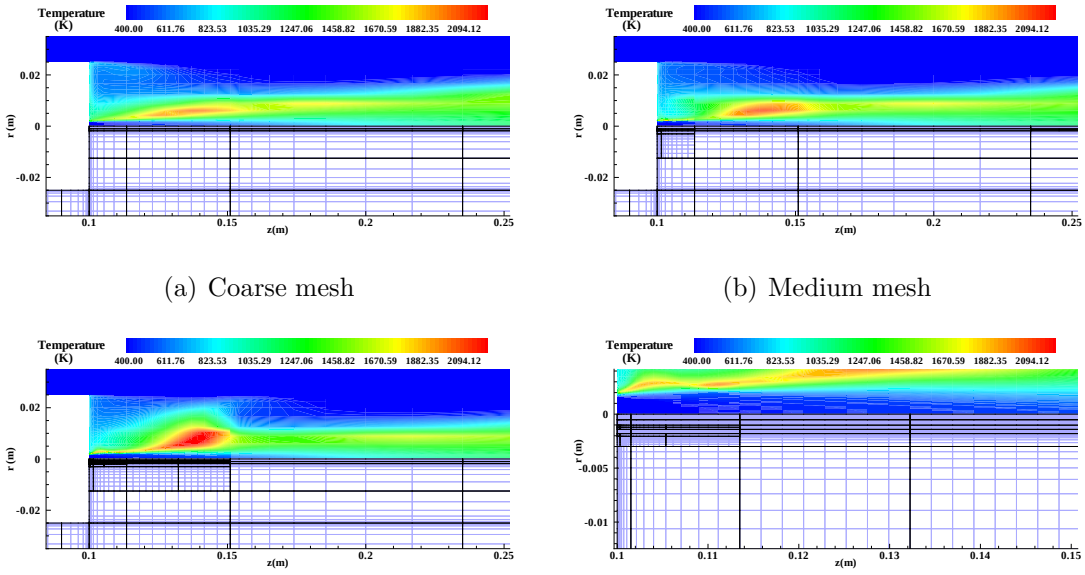
(a) $z/D_b = 0.6$ (b) $z/D_b = 1.0$

Figure 6.22: Comparison of predicted and measured radial profiles of $\overline{v'w'}$ at various locations downstream from the base of the bluff-body burner for non-reacting flow with air jet.

in xz-planes ($y = 0$) is shown in Figures 6.23(a)–6.23(d). Similar to the non-reacting case, the mesh resolution has a typical off-wall spacing of the first computational cells nearest the wall in the range $0.47 < y^+ < 1.2$.

The experimental and numerical bluff-body flame structure are shown in Figures 6.24(a) and 6.24(b), respectively. Figures 6.25(a) and 6.25(b) provide the computed distributions of mean mass fraction of CO_2 and temperature in the xz-plane ($y = 0$) for this turbulent non-premixed flame, respectively. The predicted flame structure is generally in very good agreement with the experimental observations and the previous two-dimensional axisymmetric results. Like the experimental flame, the numerical flame is quite elongated and three zones can be identified: the re-circulation, neck, and jet-like propagation zones. As noted previously, a vortex structure is formed in the re-circulation zone and acts to stabilize the flame. The maximum computed flame temperature is about 2100 K, which is close to the value of 2180 K observed in previous axisymmetric studies of this bluff-body hot-flow case.

Figures 6.25(c) and 6.25(d) compare the radial profiles of the predicted mean temperature and mass fraction of CO_2 to the experiments at a location of $z/D_b = 1.92$ downstream from the base of the bluff body. Additionally, the figures also show the results of the mesh refinement study and indicate that the predicted mean quantities of CO_2 mass fraction and temperature do not change significantly as the mesh is refined from 68,096 cells to 107,520 cells. This provides confidence that the



(a) Coarse mesh

(b) Medium mesh

(c) Fine mesh

(d) A zoom-in of fine mesh

Figure 6.23: Predicted mean axial velocity contours on coarse, medium and fine meshes, each consists of a number of $8\times8\times8$ cell blocks: (a) coarse mesh consists of 56 solution blocks 28,672 cells, (b) medium mesh consists of 133 solution blocks 68,096 cells of 2-level refinement with a refinement efficiency of 0.70; (c) fine mesh consists of 210 solution blocks 107,520 cells of 3-level refinement with a refinement efficiency of 0.9414, and (d) shows the magnified view of the refined mesh in the region close to the fuel jet and bluff-body solid wall in the fine mesh.

numerical solution is converging toward a grid-independent result that agrees well with experiments. The predicted mean temperature, 1628 K, and mass fraction of CO_2, 0.095, at the location of $z/D_b=1.92$, $r/R_b=0.4$ are comparable to the measured values of the flame temperature, 1120 K, and carbon dioxide concentration, 0.07. At the location of $z/D_b=1.92$, $r/R_b=0.52$, the predicted mean temperature, 940 K, and mass fraction of CO_2, 0.04, are quite close compared to the measured values of the flame temperature, 810 K, and carbon dioxide concentration, 0.055. The comparisons shown in the figure indicate that both the mean temperature and mass fraction of CO_2 are generally well predicted considering the simplified chemical kinetics and eddy dissipation model used for prescribing the interaction between turbulence and chemistry.

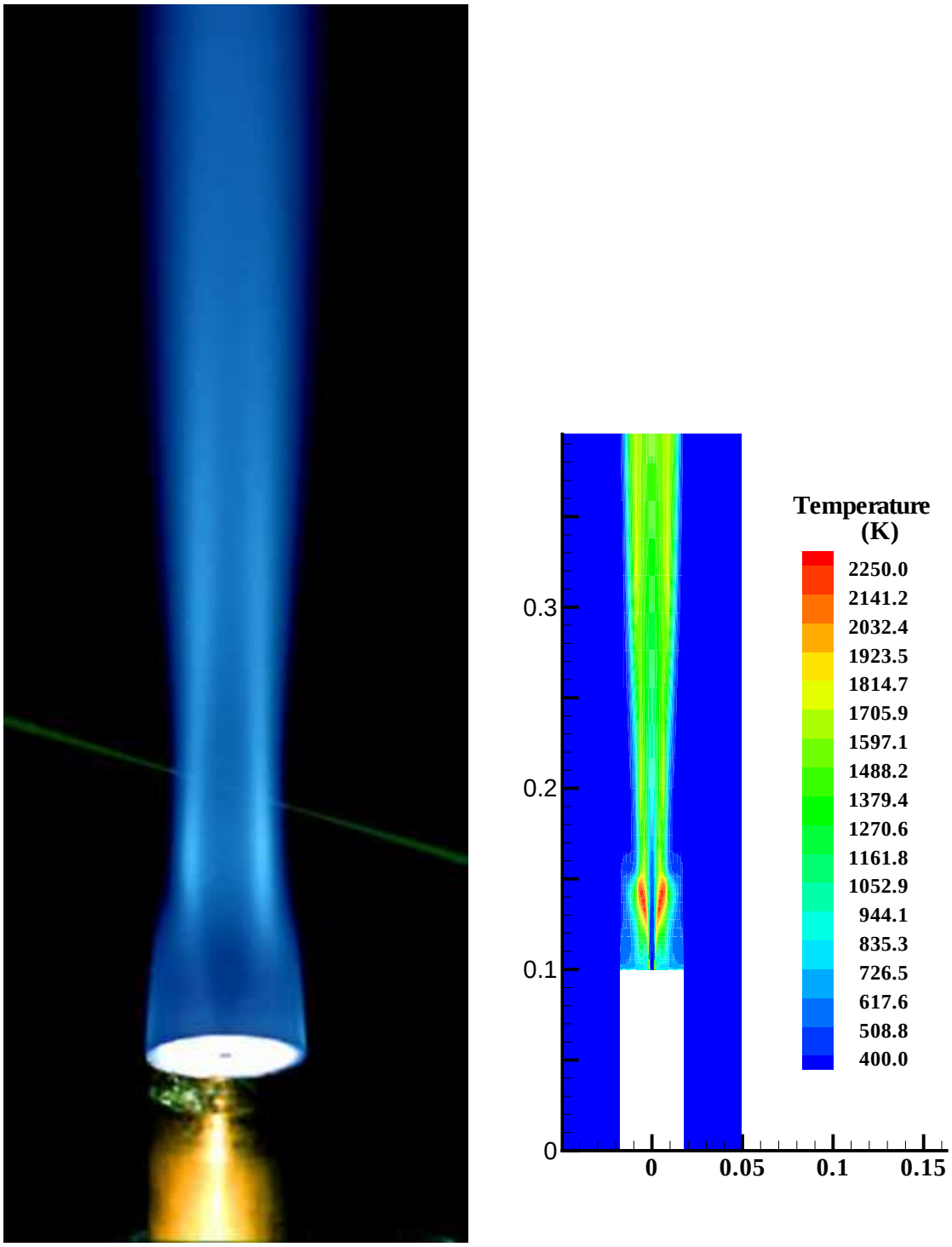
Table 6.1: The parallel fraction of the program varies with increasing the number of processors

CPU(s)	f
2	100%
6	100%
14	99.96%
21	100%
42	99.93%

6.3.3 Parallel Performance

The parallel performance of the three-dimensional algorithm has also been assessed for the Sydney bluff-body burner flame. For this purpose, a fixed size problem was considered with a multi-block hexahedral computational mesh consisting of 42 solution blocks $(8 \times 8 \times 8)$ 21,504 cells. The parallel performance was then measured using up to 42 processors. The strong scaling performance results are shown in Figure 6.26. The results indicate that the parallel speedup remains essentially linear with an efficiency of 98% up to 42 processors. Again, Table 6.1 provides an estimate of the parallel fraction of the program using Amdahl's law. The serial fraction remains particularly low (0.04%–0.07%) and the proposed algorithm can be seen to be well suited for scaling to larger numbers of processors.

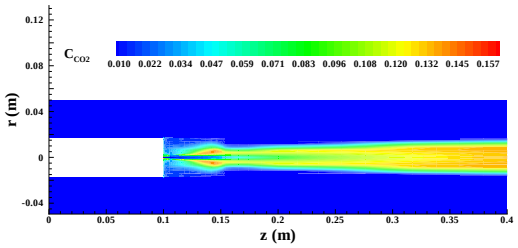
The parallel efficiency for the three dimensional case is particularly good as compared to the two-dimensional results discussed earlier due to the high ratio of computational work to communication time for each processor. This feature will be generally true for most three-dimensional problems and provides strong support for the use of the proposed block-based AMR approach for predicting three-dimensional combusting flows using parallel computer architectures.



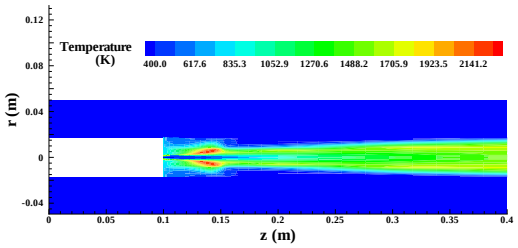
(a) Experimental bluff-body flame

(b) Numerical bluff-body flame

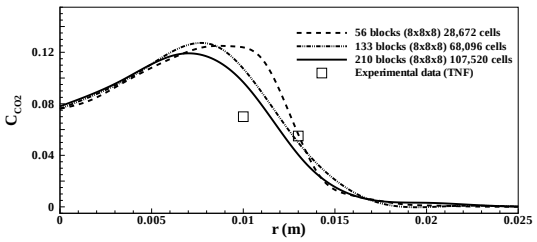
Figure 6.24: Comparison of the experimental and numerical bluff-body flames.



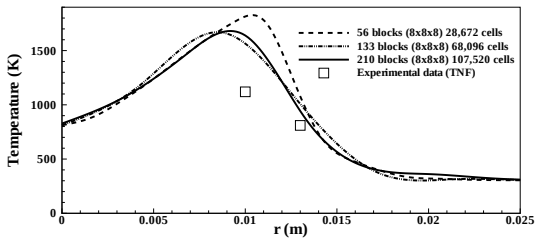
(a) CO_2 contour in the $y=0$ xz-plane



(b) Temperature contour in the $y=0$ xz-plane



(c) CO_2 radial profile at $x/D_b=1.92$



(d) Temperature radial profile at $x/D_b=1.92$

Figure 6.25: Predicted mean mass fraction of CO_2 and temperature contour on the final fine mesh; and comparison of predicted mean mass fraction of CO_2 and temperature profiles at the location of $z/D_b = 1.92$ downstream for the reacting flow with CH_4 jet.

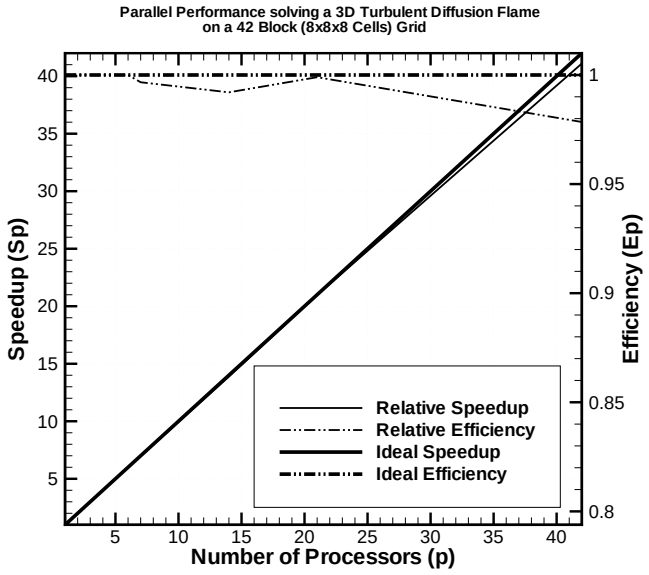


Figure 6.26: Parallel speedup and efficiency for computation of Sydney bluff-body burner flame problem using up to 42 processors.

Chapter 7

Conclusions

7.1 Conclusions

A new highly parallelized AMR scheme has been described for obtaining steady-state solutions of two- and three-dimensional turbulent non-premixed combusting flows. The parallel AMR algorithm solves the system of PDEs governing turbulent compressible flows of reactive thermally perfect gaseous mixtures using a fully coupled finite-volume formulation on body-fitted multi-block quadrilateral and hexahedral mesh. This compressible formulation can readily accommodate large density variations and thermo-acoustic phenomena. As the primary focus of the thesis research has been the efficient solution of non-premixed combusting flows and not the improved modelling of such flows, somewhat simplified models for the turbulence/chemistry interaction and chemical kinetics were employed. In particular, an eddy-dissipation model, one-step chemical kinetics for gaseous fuels, and a RANS formulation based on a two-equation model of turbulence were considered. These models afforded a level of simplicity which was very beneficial to the algorithm development yet provided sufficient realism to assess the performance of the solution method. The combination of a block-based AMR strategy, hierarchical tree data structure, and parallel implementation has resulted in a highly scalable computational tool.

Following a partial verification of the proposed parallel algorithm, the scheme was applied to the numerical prediction of a bluff-body stabilized turbulent diffusion flame.

Both two-dimensional axisymmetric and fully three-dimensional configurations were considered and the numerical results for the cold- and hot-flow cases were compared to available experimental data, including mean velocity and turbulence quantities. Given the complexity of the combusting flow field, limitations of the reduced chemical reaction mechanism, and simplified turbulence/chemistry interaction model adopted here, the results reported herein are quite encouraging and indicate the potential of the algorithm for predicting more complex combusting flows.

The primary goal of this research was to establish a computational framework for predicting complex turbulent non-premixed reacting flows in practical combustor geometries and this goal has generally been achieved. The extension and application of the proposed algorithm to RANS and LES modelling of other non-premixed flames including more sophisticated turbulence chemistry interaction model, more detailed chemical kinetic schemes, soot and radiation models, and multi-phase flow treatment for liquid fuels will be the subject of other follow-on research in the future.

7.2 Original Contributions

This thesis research has contributed to the development of a parallel AMR algorithm for performing simulations of both two- and three-dimensional turbulent combusting flows. The key original features of the proposed parallel AMR algorithm are highlighted below.

1. A new parallel AMR framework has been developed for performing solution-direction mesh adaptation of multi-block body-fitted grids and applied to the prediction of turbulent reactive flows. The block-based AMR technique allows for anisotropic grids and makes use of a flexible hierarchical tree data structure for the treatment of complex grid topologies having unstructured block connectivity. By design, the block-based approach also leads to a highly scalable and efficient parallel implementation of the finite-volume solution scheme on multi-processor parallel clusters.

   Note that the proposed block-based AMR technique has the potential to both

reduce the time/cost associated with mesh generation, as much of the mesh generation is carried out in an automated fashion, as well as reduce the time/cost associated with obtaining a solution, as the AMR mesh is more efficient.

2. A novel low-cost and computationally efficient technique has also been proposed for the generation of refined body-fitted or curvilinear grid blocks which must be determined as part of the AMR process. The grid refinement procedure makes use of standard grid metrics to preserve the original mesh topology, smoothness of the grid lines, and grid point clustering of the body-fitted mesh.

3. For complex turbulent combusting flows, dealing with the near-wall turbulence is challenging within an AMR procedure. A somewhat novel automatic and smooth switching procedure for computing wall turbulence has been proposed, and that is well suited to the AMR scheme considered here. This automatic near-wall treatment readily accommodates situations during AMR where the mesh resolution may not be sufficient for directly calculating near-wall turbulence using the low-Reynolds-number formulation.

4. In order to provide enhanced convergence for steady-state problems, a preconditioned (matrix preconditioner) multigrid strategy has been proposed and developed for two-dimensional turbulent combustion calculations. It is thought to be one of the first applications of a parallel AMR scheme with multigrid to turbulent-combusting flow calculations in the literature, and significantly improved convergence was achieved by devising a cell-aspect-ratio-based prolongation operator for treating highly stretched meshes.

5. Finally, a quantitative evaluation of the parallel AMR algorithm with the features described above has been carried out for a complex turbulent combusting flow having a relatively complex physical geometry (the bluff-body burner) and the numerical predictions were compared to experimental data. This numerical study demonstrates the validity and potential of the parallel AMR approach for predicting fine-scale features of complex turbulent non-premixed flames.

7.3 Future Research

There are a number of avenues for future research that have arisen from or during the course of this work. They are as follows:

- Investigation of Newton-Krylov-Schwarz (NKS) strategies to improve the efficiency of the time integration procedure while maintaining high parallel efficiency. Note that the current time-marching scheme as described is certainly not optimal, particularly for the three-dimensional case, but the proposed block-based AMR scheme is well suited to a NKS treatment. The NKS approach could be combined with a multigrid or multi-level method.

- Inclusion of more sophisticated combustion modelling to produce more accurate flame predictions including improved modelling of turbulence/chemistry interaction, chemical kinetics, soot formation and transport, radiation transport and liquid fuels [11, 108–110].

- Investigation of refinement criteria based on reconstruction error estimation.

- Investigation of anisotropic grid refinement based on directional dependent refinement and a binary tree data structure to enhance the efficiency of the AMR algorithm.

References

[1] J.Laufer, "The Structure of Turbulence in Fully Developed Pipe Flow," Report 1174, NACA, 1954.

[2] AQ, "Air Quality," http://www.env.gov.bc.ca/air/airregs.html/ http://www.europeansealing.com/workinggroups/Emission http://www.epa.gov/ebtpages/air.html.

[3] Vervisch, L., Hauguel, R., Domingo, P., and Rullaud, M., "Three Facets of Turbulent Combustion Modelling: DNS of Premixed V-Flame, LES of Lifted Nonpremixed Flame and RANS of Jet-Flame," *Journal of Turbulence*, Vol. 5, 2004, pp. 1–36.

[4] Cant, S., "High-Performance Computing in Computational Fluid Dynamics: Progress and Challenges," *Philosophical Transactions: Mathematical, Physical and Engineering Sciences*, Vol. 360, No. 1795, 2002, pp. 1211–1225.

[5] Vervisch, L., "DNS and LES of Turbulent Combustion," Computational Fluid Dynamics in Chemical Reaction Engineering IV, June 19-24, 2005, Barga, Italy.

[6] Piomelli, U., "Large-Eddy Simulation: Achievements and Challenges," *Prog. Aerospace Sci.*, Vol. 35, 1999, pp. 335–362.

[7] Veynante, D. and Poinsot, T., "Large Eddy Simulation of Combustion Instabilities in Turbulent Premixed Burners," Annual Research Briefs 253–274, Center for Turbulence Research, NASA Ames/Stanford Univ., 2000.

[8] Selle, L., Lartigue, G., Poinsot, T., Krebs, P. K. W., and Veynante, D., "Large-Eddy Simulation of Turbulent Combustion for Gas Turbines with Reduced Chemistry," *Proceedings of the Summer Program*, 2002, pp. 333 – 344.

[9] Pitsch, H., "Large-Eddy Simulation of Turbulent Combustion," *Annual Review of Fluid Mechanics*, Vol. 38, 2006, pp. 453–482.

[10] Davis, S., Freund, L., Leibovich, S., and Tvergaard, V., editors, *Turbulent Combustion*, Cambridge University Press, 2000.

[11] Poinsot, T. and Veynante, D., *Theoretical and Numerical Combustion*, R.T. Edwards Inc., 2001.

[12] Fox, R. O., editor, *Computational Models for Turbulent Reacting Flows*, Cambridge University Press, Cambridge, 2003.

[13] Berger, M. J. and Oliger, J., "Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations," *Journal of Computational Physics*, Vol. 53, 1984, pp. 484–512.

[14] Berger, M. J. and Colella, P., "Local Adaptive Mesh Refinement for Shock Hydrodynamics," *Journal of Computational Physics*, Vol. 82, 1989, pp. 67–84.

[15] Thompson, M. and Ferziger, J., "An Adaptive Multigrid Technique for the Incompressible Navier-Stokes Equations," *Journal of Computational Physics*, Vol. 82, No. 1, 1989, pp. 94–121.

[16] Quirk, J. J., *An Adaptive Grid Algorithm for Computational Shock Hydrodynamics*, Ph.D. thesis, Cranfield Institute of Technology, January 1991.

[17] Powell, K. G., Roe, P. L., and Quirk, J., "Adaptive-Mesh Algorithms for Computational Fluid Dynamics," *Algorithmic Trends in Computational Fluid Dynmaics*, edited by M. Y. Hussaini, A. Kumar, and M. D. Salas, Springer-Verlag, New York, 1993, pp. 303–337.

[18] De Zeeuw, D. and Powell, K. G., "An Adaptively Refined Cartesian Mesh Solver for the Euler Equations," *Journal of Computational Physics*, Vol. 104, 1993, pp. 56–68.

[19] Quirk, J. J. and Hanebutte, U. R., "A Parallel Adaptive Mesh Refinement Algorithm," Report 93-63, ICASE, August 1993.

[20] Berger, M. J. and Saltzman, J. S., "AMR on the CM-2," *Applied Numerical Mathematics*, Vol. 14, 1994, pp. 239–253.

[21] Almgren, A. S., Buttke, T., and Colella, P., "A Fast Adaptive Vortex Method in Three Dimensions," *Journal of Computational Physics*, Vol. 113, 1994, pp. 177–200.

[22] Bell, J., Berger, M., Saltzman, J., and Welcome, M., "A Three-Dimensional Adaptive Mesh Refinement for Hyperbolic Conservation Laws," *SIAM Journal on Scientific Computing*, Vol. 15, 1994, pp. 127–138.

[23] Pember, R. B., Bell, J. B., Colella, P., Crutchfield, W. Y., and Welcome, M. L., "An Adaptive Cartesian Grid Method for Unsteady Compressible Flow in Irregular Regions," *Journal of Computational Physics*, Vol. 120, 1995, pp. 278–304.

[24] Aftomis, M. J., Berger, M. J., and Melton, J. E., "Robust and Efficient Cartesian Mesh Generation for Component-Base Geometry," *AIAA Journal*, Vol. 36, No. 6, 1998, pp. 952–960.

[25] Almgren, A. S., Bell, J. B., Colella, P., Howell, L. H., and Welcome, M. J., "A Conservative Adaptive Projection Method for the Variable Density Incompressible Navier-Stokes Equations," *Journal of Computational Physics*, Vol. 142, 1998, pp. 1–46.

[26] Pember, R. B., Howell, L., Bell, J., Colella, P., Crutchfield, W. Y., Fiveland, M. A., and Jessee, J., "An Adaptive Projection Method for Unsteady, Low Mach Number Combustion," *Combustion Science and Technology*, Vol. 140, 1998, pp. 123–168.

[27] Jessee, J., Fiveland, W., Howell, L., Colella, P., and Pember, R., "An Adaptive Mesh Refinement Algorithm for the Radiative Transport Equation," *Journal of Computational Physics*, Vol. 139, No. 2, 1998, pp. 380–398.

[28] Colella, P., Dorr, M., and Wake, D., "Numerical Solution of Plasma-Fluid Equations Using Locally Refined Grids," *Journal of Computational Physics*, Vol. 152, 1999, pp. 550–583.

[29] Howell, L., Pember, R., Colella, P., Jessee, J., and Fiveland, W. A., "A Conservative Adaptive-Mesh Algorithm for Unsteady, Combined-Mode Heat Transfer Using the Discrete Ordinates Method," *Numerical Heat Transfer*, Vol. 35, 1999, pp. 407–430.

[30] Groth, C. P. T., Zeeuw, D. L. D., Powell, K. G., Gombosi, T. I., and Stout, Q. F., "A Parallel Solution-Adaptive Scheme for Ideal Magnetohydrodynamics," Paper 99-3273, AIAA, June 1999.

[31] Groth, C. P. T., De Zeeuw, D. L., Gombosi, T. I., and Powell, K. G., "Global Three-Dimensional MHD Simulation of a Space Weather Event: CME Formation, Interplanetary Propagation, and and Interaction with the Magnetosphere," *Journal of Geophysical Research*, Vol. 105, No. A11, 2000, pp. 25,053–25,078.

[32] Sachdev, J., Groth, C., and Gottlieb, J., "A Parallel Solution-Adaptive Scheme for Predicting Multi-Phase Core Flows in Solid Propellant Rocket Motors," *International Journal of Computational Fluid Dynamics*, Vol. 19, No. 2, 2005, pp. 159–177.

[33] Berger, M. J., "Data Structures for Adaptive Grid Generation," *SIAM Journal for Scientific and Statistical Computing*, Vol. 904, 1986.

[34] Group, A. N. A., "CHOMBO," National Energy Research Scientific Computing Center, Lawrence Berkeley National Laboratory, http://seesar.lbl.gov/anag/chombo.

[35] Group, A. N. A., "Structured Adaptive Mesh Refinement Application Infrastructure," The Center for Applied Scientific Computing (CASC) at Lawrence Livermore National Laboratory https://computation.llnl.gov/casc/SAMRAI.

[36] De Zeeuw, D. L., *A Quadtree-Based Adaptively-Refined Cartesian-Grid Algorithm for Solution of the Euler Equations*, Ph.D. thesis, University of Michigan, September 1993.

[37] Berger, M. J. and LeVeque, R. J., "An Adaptive Cartesian Mesh Algorithm for the Euler Equations in Arbitrary Geometries," Paper 89-1930, AIAA, June 1989.

[38] Aftomis, M. J., Berger, M. J., and Adomavicius, G., "A Parallel Multilevel Method for Adaptively Refined Cartesian Grids with Embedded Boundaries," Paper 2000-0808, AIAA, January 2000.

[39] Coirier, W. J., *An Adaptively-Refined, Cartesian, Cell-Based Scheme for the Euler and Navier-Stokes Equations*, Ph.D. thesis, University of Michigan, 1994.

[40] Aftosmis, M., Berger, M., and Nemec, M., "Cart3D," http://people.nas.nasa.gov/ aftosmis/cart3d/.

[41] Lin, C.-W., Percival, S., and Gotimer, E. H., "Chimera Composite Grid Scheme," http://www.nswccd.navy.mil/hyd/tec-rep/chi-com-gri/index.html.

[42] E.P.Boden and Toro, E. F., "A Combined Chimera-AMR Technique for Computing Hyperbolic PDEs," *Proceedings of the Fifth Annual Conference of the CFD Society of Canada, Toronto, Ontario, Canada*, CFD Society of Canada, 1997, pp. 5.13–5.18.

[43] Brislawn, K. D., Brown, D. L., Chesshire, G., and Saltzman, J., "Adaptively-Refined Overlapping Grids for the Numerical Solution of Hyperbolic Systems of Conservation Laws," Report LA-UR-95-257, Los Alamos National Laboratory, 1995.

[44] Chesshire, G. and Henshaw, W., "Composite Overlapping Meshes for the Solution of Partial Differential Equations," *Journal of Computational Physics*, Vol. 90, 1990, pp. 1–64.

[45] Henshaw, W. D., "Adaptive Mesh Refinement on Overlapping," *Adaptive Mesh Refinement - Theory and Applications*, Springer Berlin Heidelberg, 2005.

[46] Henshaw, W. D. and Schwendeman, D. W., "An adaptive numerical scheme for high-speed reacting flow on overlapping grids," *Journal of Computational Physics*, Vol. 191, 2003, pp. 420–447.

[47] Henshaw, W., "OverBlown: A Fluid Flow Solver for Overlapping Grids, User Guide," Technical Report UCRL-MA-134288, LLNL, 1999.

[48] Henshaw, W., "Ogen: An Overlapping Grid Generator for Overture," Technical Report UCRL-MA-132237, LLNL, 1998.

[49] Henshaw, W. D., "A Fourth-Order Accurate Method for the Incompessible Navier-Stokes Equations on Overlapping Grids," *Journal of Computational Physics*, Vol. 113, 1994, pp. 13–25.

[50] Powell, K. G., Roe, P. L., Myong, R. S., Gombosi, T. I., and De Zeeuw, D. L., "An Upwind Scheme for Magnetohydrodynamics," Paper 95-1704-CP, AIAA, June 1995.

[51] Stout, Q. F., De Zeeuw, D. L., Gombosi, T. I., Groth, C. P. T., Marshall, H. G., and Powell, K. G., "Adaptive Blocks: A High-Performance Data Structure," *Proceedings of SC97, San Jose, California, U.S.A., November 12–15, 1997*, 1997.

[52] Gombosi, T. I., De Zeeuw, D. L., Groth, C. P. T., and Powell, K. G., "Magnetospheric Configuration for Parker-Spiral IMF Conditions: Results of a 3D AMR MHD Simulation," *Advances in Space Research*, Vol. ?, No. ?, 1999, pp. ???–???

[53] Liu, Y., Nagy, A. F., Groth, C. P. T., De Zeeuw, D. L., Gombosi, T. I., and Powell, K. G., "3D Multi-fluid MHD Studies of the Solar Wind Interaction with Mars," *Geophysical Research Letters*, Vol. 26, No. 17, 1999, pp. 2689–2692.

[54] Volberg, O., Gombosi, T. T., Powell, K. G., Ridley, A. J., Hansen, K. C., Toth, G., Stout, Q. F., Zeeuw, D. D., Kane, K., Chesney, D. R., and Oehmke, R., "A High-Performance Framework for Sun-to-Earth Space Weather Modeling," Workshop 19th, IEEE International Parallel and Distributed Processing Symposium, 2005.

[55] Groth, C. P. T. and Northrup, S. A., "Parallel Implicit Adaptive Mesh Refinement Scheme for Body-Fitted Multi-Block Mesh," Paper 2005-5333, AIAA, June 2005.

[56] Groth, C. P. T., De Zeeuw, D. L., Gombosi, T. I., and Powell, K. G., "Three-Dimensional MHD Simulation of Coronal Mass Ejections," *Advances in Space Research*, Vol. 26, No. 5, 2000, pp. 793–800.

[57] van der Holst, B. and Keppens, R., "Hybrid Block-AMR in Cartesian and Curvilinear Coordinates:MHD Applications," *Journal of Computational Physics*, Vol. 226, 2007, pp. 925–946.

[58] McHugh, P. R. and Knoll, D. A., "Comparison of Standard and Matrix-Free Implementations of Several Newton-Krylov Solvers," *AIAA Journal*, Vol. 32, No. 12, 1994, pp. 2394–2400.

[59] McHugh, P. R., Knoll, D. A., and Keyes, D. E., "Application of Newton-Krylov-Schwarz Algorithm to Low-Mach-Number Compressible Combustion," *AIAA Journal*, Vol. 36, No. 2, 1998, pp. 290–292.

[60] Gropp, W. D., Keyes, D. E., McInnes, L. C., and Tidriri, M. D., "Globalized Newton-Krylov-Schwarz Algorithms and Software for Parallel Implicit CFD," Technical Report 98-24, ICASE, July 1998.

[61] Gropp, W. D., Kaushik, D. K., Keyes, D. E., and Smith, B. F., "High-Performance Parallel Implicit CFD," *Parallel Computing*, Vol. 27, 2001, pp. 337–362.

[62] Coelho, P. J. and Pereira, J., "Calculation of a Confined Axisymmetric Laminar Diffusion Flame Using a Local Grid Refinement Technique," *Combustion Science and Technology*, Vol. 92, 1993, pp. 243–264.

[63] de Lange, H. and de Goey, L., "Numerical Modelling in a Locally Refined Grid," *International Journal for Numerical Mechanical Engineering*, Vol. 37, 1994, pp. 497–515.

[64] Giovangigli, V. and M.D.Smooke, "Adaptive Continuation Algorithms with Application to Combustion Problems," *Applied Numerical Mathematics*, Vol. 5, 1989, pp. 305–331.

[65] Mallens, R., de Lange, H., van de Ven, C., and de Goey, L., "Modeling of Confined and Unconfined Laminar Premixed Flames on Slit and Tube Burners," *Combustion Science and Technology*, Vol. 107, 1995, pp. 387–401.

[66] L.T.Somers and de Goey, L., "A Numerical Study of a Premixed Flame on a Slit Burner," *Combustion Science and Technology*, Vol. 108, 1995, pp. 121–132.

[67] Day, M. S. and Bell, J. B., "Numerical Simulation of Laminar Reacting Flows with Complex Chemistry," *Combustion Theory and Modelling*, Vol. 4, No. 4, 2000, pp. 535–556.

[68] Bennett, B. and Smooke, M., "Local Rectangular Refinement with Application to Axisymmetric Laminar Flames," *Combustion Theory and Modelling*, Vol. 2, No. 3, 1998, pp. 221–258.

[69] Bennett, B. and Smooke, M., "Local Rectangular Refinement with Application to Nonreacting and Reacting Fluid Flow Problems," *Journal of Computational Physics*, Vol. 151, 1999, pp. 684–727.

[70] Bennett, B., Fielding, J., Mauro, R., Long, M., and Smooke, M., "A Comparison of the Structures of Lean and Rich Axisymmetric Laminar Bunsen Flames: Application of Local Rectangular Refinement Solution-Adaptive Gridding," *Combustion Theory and Modelling*, Vol. 3, 1999, pp. 657–687.

[71] Valdati, B. A., *Solution-Adaptive Gridding Methods with Application to Combustion Problems*, Ph.D. thesis, Yale University, 1997.

[72] da Silva, L., Azevedo, J. L. F., and Korzenowski, H., "Unstructured Adaptive Grid Flow Simulations of Inert and Reactive Gas Mixtures," *Journal of Computational Physics*, Vol. 160, 2000, pp. 522–540.

[73] Sullivan, N., Jensen, A., Glarborg, P., Day, M. S., Grcar, J., Bell, J., Pope, C. J., and Kee, R. J., "Ammonia Conversion and NO_x Formation in Laminar Coflowing Nonpremixed Methane-Air Flames," *Combustion and Flame*, Vol. 131, 2002, pp. 285–298.

[74] Reynolds, W. and Fatica, M., "Stanford Center for Integrated Turbulence Simulations," Technical Paper 54–63, Stanford University, Marh/April 2000.

[75] Medic, G., Kalitzin, G., You, D., Herrmann, M., Ham, F., van der Weide, E., Pitsch, H., and Alonso, J., "Integrated RANS/LES Computations of Turbulent Flow Through a Turbofan Jet Engine," Annual Research Briefs 275–285, Stanford University, 2006.

[76] Douglas, C. C., Ern, A., and Smooke, M. D., "Numerical Simulation of Flames Using Multigrid Methods," *Iterative Methods in Scientific Computation*, edited by J. Wang, M. B. Allen, B. M. Chen, and T. Mathew, Vol. 4 of *IMACS Series in Computational and Applied Mathematics*, New Brunswick, 1998, pp. 149–154.

[77] Ern, A., *Vorticity-Velocity Modeling of Chemically Reacting Flows*, Ph.D. thesis, Yale University, 1994.

[78] Ern, A., Douglas, C., and Smooke, M., "Detailed Chemistry Modeling of Laminar Diffusion Flames on Parallel Computers," *International Journal of Supercomputer Applications and High Performance Computing*, Vol. 10, 1995, pp. 225–236.

[79] Desprez, F., "Practical Aspects and Experiences Numerical Simulation of a Combustion problem on a Paragon Machine," *Parallel Computing*, Vol. 21, 1995, pp. 495–508.

[80] Nkonga, B. and Charrier, P., "Generalized Parcel Method for Dispersed Spray and Message Passing Strategy on Unstructured Meshes," *Parallel Computing*, Vol. 28, 2002, pp. 369–398.

[81] Yasar, O., "A Scalable Model for Complex Flows," *Computers & Mathematics with Applications*, Vol. 35, No. 7, 1998, pp. 117–128.

[82] Yasar, O. and Moses, G., "Explicit Adaptive Grid Radiation Magnetohydrodynamics," *Journal of Computational Physics*, Vol. 100, No. 1, 1992.

[83] Yasar, O. and Zacharia, T., "Distributed Implementation of KIVA-3 on the Intel Paragon," in *Parallel Computational Fluid Dynamics*, 1996.

[84] Yasar, O. and Zacharia, T., "Distributed Implementation of KIVA-3 on the Intel Paragon," *Journal of Computational Physics*, Vol. 100, No. 1, 1992.

[85] E.S.Oran, Weber, J., Stefaniw, E. I., M.H.Lefebvre, and Anderson, J., "A Numerical Study of a Two-Dimensional H_2-O_2-Ar Detonation Using a Detailed Chemical Reaction Model," *Combustion and Flame*, Vol. 113, 1998, pp. 147–163.

[86] Lepper, J., Schnell, U., and Hein, K. G., "Parallelization of a Simulation Code for Reactive Flows on the Intel Paragon," *Computers& Mathematics with Applications*, Vol. 35, No. 7, 1998, pp. 101–109.

[87] Huang, Y., Sung, H., Hsieh, S., and Yang, V., "Large-Eddy Simulation of Combustion Dynamics of Lean-Premixed Swirl-Stabilized Combustor," *Journal of Propulsion and Power*, Vol. 19, No. 5, 2003, pp. 782–794.

[88] SNL, "Sandia National Laboratory," http://pmw.org/ ravi/work/sandia/.

[89] Bell, J., Day, M., Almgren, A., Lijewski, M. J., and Rendleman, C. A., "Adaptive Numerical Simulation of Turbulent Premixed Combustion," *Proceedings of the First MIT Conference on Computational Fluid and Solid Mechanics*, June 2001.

[90] Bell, J., Day, M., Almgren, A., Lijewski, M. J., and Rendleman, C. A., "A Parallel Adaptive Projection Method for Low Mach Number Flows," *International Journal for Numerical Methods in Fluids*, Vol. 40, 2002, pp. 209–216.

[91] Bell, J., "AMR for Low Mach Number Reacting Flow," http://repositories.cdlib.org/lbnl/ LBNL-54351, 2004.

[92] Bell, J. B., Day, M., Rendleman, C., Woosley, S., and M.A.Zingale, "Adaptive Low Mach Number Simulations of Nuclear Flame Microphysics," *Journal of Computational Physics*, Vol. 195, 2004, pp. 677–694.

[93] Day, M. S. and Bell, J. B., "Simulation of Premixed Turbulent Flames," *Journal of Physics*, Vol. 46, 2006, pp. 43–47.

[94] Northrup, S. A. and Groth, C. P. T., "Solution of Laminar Diffusion Flames Using a Parallel Adaptive Mesh Refinement Algorithm," Paper 2005-0547, AIAA, January 2005.

[95] Gao, X. and Groth, C. P. T., "Parallel Adaptive Mesh Refinement Scheme for Turbulent Non-Premixed Combusting Flow Prediction," Paper 2006-1448, AIAA, January 2006.

[96] Gao, X. and Groth, C. P. T., "A Parallel Adaptive Mesh Refinement Algorithm for Predicting Turbulent Non-Premixed Combusting Flows," *International Journal of Computational Fluid Dynamics*, Vol. 20, No. 5, 2006, pp. 349–357.

[97] Gao, X. and Groth, C. P. T., "Parallel Adaptive Mesh Refinement Scheme for Three-Dimensional Turbulent Non-Premixed Combustion," Paper 2008-1017, AIAA, January 2008.

[98] Wilcox, D. C., *Turbulence Modeling for CFD*, DCW Industries, La Cañada, 2002.

[99] Brinckman, K. W., Kenzakowski, D. C., and Dash, S. M., "Progress in Practical Scalar Fluctuation Modeling for High-Speed Aeropropulsive Flows," Paper 1–18, AIAA, January 2005.

[100] Pope, S., *Turbulent Flows*, Cambridge University Press, 2000.

[101] Spalart, P. R. and Allmaras, S. R., "A one-equation turbulence model for aerodynamic flows," Paper 92-0439, AIAA, 1992.

[102] Anderson, J. D., *Modern Compressible Flow with Historical Perspective*, McGraw-Hill, New York, 1990.

[103] Gordon, S. and McBride, B. J., "Computer Program for Calculation of Complex Chemical Equilibrium Compositions and Applications I. Analysis," Reference Publication 1311, NASA, 1994.

[104] McBride, B. J. and Gordon, S., "Computer Program for Calculation of Complex Chemical Equilibrium Compositions and Applications II. Users Manual and Program Description," Reference Publication 1311, NASA, 1996.

[105] Wilke, C. R., "A Viscosity Equation for Gas Mixtures," *Journal of Chemical Physics*, Vol. 18, 1950, pp. 517–519.

[106] W. C. Gardiner, J., *Combustion Chemistry*, Springer-Verlag New York Inc., Boca Raton, 1984.

[107] Westbrook, C. K. and Dryer, F. L., "Simplified Reaction Mechanisms for the Oxidation of Hydrocarbon Fuels in Flames," *Combustion Science and Technology*, Vol. 27, 1981, pp. 31–43.

[108] Vervisch, L. and D.Veynante, "Turbulent combustion modeling," *Introduction to turbulent combustion Lecture Series 1999-04*, von Karman Institute for Fluid Dynamics, Rhode Saint Genese, Belgium, 1999.

[109] Eaton, A. M., Smoot, L. D., Hill, S. C., and Eatough, C. N., "Components, Formulations, Solutions, Evaluation, and Application of Comprehensive Combustion Models," *Progress in Energy and Combustion Science*, Vol. 25, 1999, pp. 387–436.

[110] Veynante, D. and Vervisch, L., "Turbulent Combustion Modeling," *Progress in Energy and Combustion Science*, Vol. 28, 2002, pp. 193–266.

[111] Bray, K. N. C., "The Challenge of Turbulent Combustion," Paper, Twenty-Sixth Symposium (International) on Combustion/The Combustion Institute, 1996.

[112] Magnussen, B. and Hjertager, B., "On mathematical modeling of turbulent combustion with special emphasis on soot formation and combustion," *Sixteenth Symposium (International) on Combustion*, The Combustion Institute, 1976, pp. 719–729.

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

[114] Hirsch, C., *Numerical Computation of Internal and External Flows, Volume 1, Fundamentals of Numerical Discretization*, John Wiley & Sons, Toronto, 1989.

[115] Hirsch, C., *Numerical Computation of Internal and External Flows, Volume 2, Computational Methods for Inviscid and Viscous Flows*, John Wiley & Sons, Toronto, 1990.

[116] Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics*, Vol. 43, 1981, pp. 357–372.

[117] Einfeldt, B., "On Godunov-Type Methods for Gas Dynamics," *SIAM Journal on Numerical Analysis*, Vol. 25, 1988, pp. 294–318.

[118] Coirier, W. J. and Powell, K. G., "An Accuracy Assessment of Cartesian-Mesh Approaches for the Euler Equations," *Journal of Computational Physics*, Vol. 117, 1995, pp. 121–131.

[119] Mathur, S. R. and Murthy, J. Y., "A Pressure-Based Method for Unstructured Meshes," *Numerical Heat Transfer*, Vol. 31, 1997, pp. 191–215.

[120] van Leer, B., "Upwind and High-Resolution Methods for Compressible Flow: From Donor Cell to Residual Distribution Schemes," *Communications in Computational Physics*, Vol. 1, No. 2, 2006, pp. 192–206.

[121] Boris, J. P. and Book, D. L., "Flux-Corrected Transport. I. SHASTA, A Fluid Transport Algorithm That Works," *Journal of Computational Physics*, Vol. 11, 1973, pp. 38–69.

[122] van Leer, B., "Towards the Ultimate Conservative Difference Scheme. II. Monotonicity and Conservation Combined in a Second Order Scheme," *Journal of Computational Physics*, Vol. 14, 1974, pp. 361–370.

[123] Harten, A., "High Resolution Schemes for Hyperbolic Conservation Laws," *Journal of Computational Physics*, Vol. 49, 1983, pp. 357–393.

[124] Venkatakrishnan, V., "On the Accuracy of Limiters and Convergence to Steady State Solutions," Paper 93-0880, AIAA, January 1993.

[125] Barth, T. J., "Recent Developments in High Order K-Exact Reconstruction on Unstructured Meshes," Paper 93-0668, AIAA, January 1993.

[126] Mavriplis, D. J., "Revisiting the Least-squares Procedure for Gradient Reconstruction on Unstructured Meshes," Report 2003-06, NIA, 2003.

[127] Anderson, W. and Bonhaus, D. L., "An Implicit Upwind Algorithm for Computing Turbulent Flows on Unstructured Grids," *Computers Fluids*, Vol. 23, No. 1, 1994, pp. 1–21.

[128] Haselbacher, A. and Blazek, J., "On the Accurate and Efficient Discretization of the Navier-Stokes Equations on Mixed Grids," Paper 1999-3363, AIAA, June 1999.

[129] Gottlieb, J. J. and Groth, C. P. T., "Assessment of Riemann Solvers for Unsteady One-Dimensional Inviscid Flows of Perfect Gases," *Journal of Computational Physics*, Vol. 78, 1988, pp. 437–458.

[130] Osher, S. and Solomon, F., "Upwind Difference Schemes for Hyperbolic Systems of Conservation Laws," *Mathematics of Computation*, Vol. 38, No. 158, 1982, pp. 339–374.

[131] van Leer, B., "On the Relation Between the Upwind-Difference Schemes of Godunov's Engquist Osher and Roe," *SIAM Journal for Scientific and Statistical Computing*, Vol. 5, 1984, pp. 1–20.

[132] Harten, A., Lax, P. D., and van Leer, B., "On Upstream Differencing and Godunov-Type Schemes for Hyperbolic Conservation Laws," *SIAM Review*, Vol. 25, No. 1, 1983, pp. 35–61.

[133] Powell, K. G., Toth, G., Zeeuw, D. D., Roe, P., and Gombosi, T., "Development and Validation of Solution-Adaptive, Parallel Methods for Compressible Plasmas," Paper 2001–2525-CP, AIAA, June 2001.

[134] van Leer, B., "Flux-Vector Splitting for the Euler Equations," Report 82-30, ICASE, September 1982.

[135] Deconinck, H., Sermeus, K., and Abgrall, R., "Status of Multidimensional Upwind Residual Distribution Schemes and Applications in Aeronautics," Paper 2000-33804, AIAA, June 2000.

[136] Abgrall, R. and Mezine, M., "Construction of Second-Order Accurate Monotone and Stable Residual Distribution Schemes for Steady Problems," *Journal of Computational Physics*, Vol. 195, 2004, pp. 474–507.

[137] Guzik, S. M. J. and Groth, C. P. T., "Comparison of Solution Accuracy of Multidimensional Residual Distribution and Godunov-Type Finite-Volume Methods," *International Journal of Computational Fluid Dynamics*, Vol. 22, No. 1–2, 2008, pp. 61–83.

[138] LeVeque, R. J., *Numerical Methods for Conservation Laws*, Birkhäuser Verlag, Boston, 1992.

[139] Harten, A., "On a Class of High Resolution Total-Variation-Stable Finite-Difference Schemes," *SIAM Journal on Numerical Analysis*, Vol. 21, 1984, pp. 1–23.

[140] van Leer, B., Tai, C. H., and Powell, K. G., "Design of Optimally-Smoothing Multi-Stage Schemes for the Euler Equations," Paper 89-1933-CP, AIAA, June 1989.

[141] Kleb, B. and B. van Leer, B. W., "Mathcing Multistage Schemes to Viscous Flow," Paper 2005–4708, AIAA, June 2005.

[142] Mavriplis, D. J., "Multigrid Strategies for Viscous Flow Solvers on Anisotropic Unstructured Meshes," *Journal of Computational Physics*, Vol. 145, 1998, pp. 141–165.

[143] Trottenberg, U., Oosterlee, C. W., and Schüller, A., *Multigrid*, Academic Press, London, 2001.

[144] Brandt, A., "Multi-Level Adaptive Solutions to Boundary-Value Problems," *Mathematics of Computation*, Vol. 31, 1977, pp. 333–390.

[145] Pierce, N. A. and Giles, M. B., "Preconditioning Compressible Flow Calculations on Stretched Meshes," Paper 96-0889, AIAA, January 1996.

[146] Pierce, N. A. and Giles, M., "Preconditioned multigrid methods for compressible flow calculations on stretched meshes," *Journal of Computational Physics*, Vol. 136, 1997, pp. 425–445.

[147] Liu, F. and Zheng, X., "A Strongly Coupled Time-Marching Method for Solving the Navier-Stokes and $k - \omega$ Turbulence Model Equations with Multigrid," *Journal of Computational Physics*, Vol. 128, No. 2, 1996, pp. 289–300.

[148] Park, S. H. and Kwon, J. H., "Implementation of $k - \omega$ Turbulence Models in An Implicit Multigrid Method," *AIAA Journal*, Vol. 42, No. 7, 2004, pp. 1348–1357.

[149] Gerlinger, P., Stoll, P., and Brüggemann, D., "An Implicit Multigrid Method for the Simulation of Chemically Reacting Flows," *Journal of Computational Physics*, Vol. 146, 1998, pp. 322–345.

[150] Maplesoft^{TM}, "Maple," http://www.maplesoft.com/.

[151] Gerlinger, P., Möbus, H., and Brüggemann, D., "An Implicit Multigrid Method for Turbulent Combustion," *Journal of Computational Physics*, Vol. 167, 2001, pp. 247–276.

[152] Slomski, J., Anderson, J. D., and Gorski, J. J., "Effectiveness of Multigrid in Accelerating Convergence of Multidimensional Flows in Chemical Nonequilibrium," Paper 90-1575, AIAA, June 1990.

[153] Edwards, J. R., "An Implicit Multigrid Algorithm for Computing Hypersonic, Chemically Reacting Viscous Flows," *Journal of Computational Physics*, Vol. 123, No. 0007, 1996, pp. 84–95.

[154] Sheffer, S. G., Martinelli, L., and Jameson, A., "Simulation of Supersonic Reacting Hydrocardon Flows with Detailed Chemistry," *Combustion Science and Technology*, Vol. 136, 1998, pp. 55–80.

[155] Bellucci, V. and Bruno, C., "Incompressible Flows with Combustion Simulated by a Preconditioning Method Using Multigrid Acceleration and MUSCL Reconstruction," *International Journal for Numerical Methods in Fluids*, Vol. 36, 2001, pp. 619–637.

[156] Davis, R. L. and Dannenhoffer, J. F., "Decomposition and Parallelization Strategies for Adaptive Grid-Embedding Techniques," *International Journal of Computational Fluid Dynamics*, Vol. 1, 1993, pp. 79–93.

[157] Sun, M. and Takayama, K., "Conservative Smoothing on an Adaptive Quadrilateral Grid," *Journal of Computational Physics*, Vol. 150, 1999, pp. 143–180.

[158] Kautsky, J. and Nichols, K., "Equidistributing Meshes with Constraints," *SIAM Journal for Scientific and Statistical Computing*, Vol. 1, No. 4, 1980, pp. 499–511.

[159] Chen, K., "Error Equidistribution and Mesh Adaptation," *SIAM Journal on Scientific Computing*, Vol. 15, No. 4, 1994, pp. 798–818.

[160] Burman, E., Ern, A., and Giovangigli, V., "An Adaptive Finite Element Method with Crosswind Diffusion for Low Mach, Steady, Laminar Combustion," *Journal of Computational Physics*, Vol. 188, 2003, pp. 472–492.

[161] Venditti, D. A. and Darmofal, D. L., "Adjoint Error Estimation and Grid Adaptation for Functional Outputs: Application to Quasi-One-Dimensional Flow," *Journal of Computational Physics*, Vol. 164, 2000, pp. 204–277.

[162] Venditti, D. A. and Darmofal, D. L., "Grid Adaptation for Functional Outputs: Application to Two-Dimensional Inviscid Flows," *Journal of Computational Physics*, Vol. 176, 2002, pp. 40–69.

[163] Venditti, D. A. and Darmofal, D. L., "Anisotropic Grid Adaptation for Functional Outputs: Application to Two-Dimensional Viscous Flows," *Journal of Computational Physics*, Vol. 187, 2003, pp. 22–46.

[164] Ham, F., Lien, F.-S., and Strong, A. B., "A Cartesian Grid Method with Transient Anisotropic Adaption," *Journal of Computational Physics*, Vol. 179, 2002, pp. 469–494.

[165] Keats, W. and Lien, F.-S., "Two-Dimensional Anisotropic Cartesian Mesh Adaptation for the Compressible Euler Equations," *IJNMF*, Vol. 46, 2004, pp. 1099–1125.

[166] CGNS, "CFD General Notation System," http://www.grc.nasa.gov/www/cgns/.

[167] Smith, B., Bjorstad, P., and Gropp, W., *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge University Press, 1996.

[168] Ou, C.-W. and Ranka, S., "Thoughts on the chimera method of simulation of three-dimensional viscous flow," *Proceedings of the 5th Symposium on the Frontiers of Massively Parallel Computation*, IEEE Computer Society Press, McLean, VA, February 1995, 1995, pp. 367–374.

[169] Pilkington, J. and Baden, S. B., "Dynamic Partitioning of Non-Uniform Structured Workloads with Space-Filling Curves," Paper 3, IEEE Transaction on Parallel and Distributed Systems, 1996.

[170] Aftomis, M. J., Berger, M. J., and Murman, S. M., "Application of Space-Filling Curves to Cartesian Methods for CFD," Paper 2000-1232, AIAA, January 2004.

[171] Stroustrup, B., *The C++ Programming Language*, Addison-Wesley, 2001.

[172] Gropp, W., Lusk, E., and Skjellum, A., *Using MPI*, MIT Press, Cambridge, Massachussets, 1999.

[173] Sachdev, J. and Groth, C., "A Mesh Adjustment Scheme for Embedded Boundaries," *Communications in Computational Physics*, 2007, accepted in April 2007.

[174] McDonald, J. G. and Groth, C. P. T., "Numerical modeling of micron-scale flows using the Gaussian moment closure," Paper 2005-5035, AIAA, June 2005.

[175] Ivan, L. and Groth, C. P. T., "High-Order Central ENO Finite-Volume Scheme with Adaptive Mesh Refinement," Paper 2007-4323, AIAA, June 2007.

[176] AIAA, "Guide for the Verification and Validation of Computational Fluid Dynamics Simulations," AIAA G-077-1998, 1998.

[177] Sachdev, J., *Parallel Solution-Adaptive Method for Predicting Solid Propellant Rocket Motor Core Flows*, Ph.D. thesis, University of Toronto, March 2007.

[178] Schlichting, H., *Boundary-Layer Theory*, McGraw-Hill, Toronto, 7th ed., 1979.

[179] Ghia, U., Ghia, K., and Shin, C., "High-Re Solutions for Incompressible Flow Using the Navier-Stokes Equations and a Multigrid method," *Journal of Computational Physics*, Vol. 48, 1982, pp. 387–411.

[180] J.Laufer, "Investigation of Turbulent Flow in A Two-Dimensional Channel," Report 1053, NACA, 1951.

[181] Chapman, B., Jost, G., and Pas, R. V. D., *Using OpenMP*, MIT Press, Cambridge, Massachussets, 2007.

[182] TNF, "International Workshop on Measurement and Computation of Turbulent Nonpremixed Flames," http://www.ca.sandia.gov/TNF/.

[183] Masri, A. R., Dibble, R. W., and Barlow, R. S., "The Structure of Turbulent Nonpremixed Flames of Methanol over a Range of Mixing Rates," *Combustion and Flame*, Vol. 89, 1992, pp. 167–185.

[184] Masri, A. R., Dibble, R. W., and Barlow, R. S., "Raman-Rayleigh Measurements in Bluff Body Stabilised Flames of Hydrocarbon Fuels," *Twenty-fourth Symposium (International) on Combustion*, The Combustion Institute, 1992, pp. 317–324.

[185] Masri, A. R., Dally, B. B., and Barlow, R. S., "The Structure of the Recirculation Zone of a Bluff-Body Combustor," *Twenty-fifth Symposium (International) on Combustion*, The Combustion Institute, 1994, pp. 1301–1308.

[186] Fallot, L., Gonzalez, M., Elamraoui, R., and Obounou, M., "Modelling Finite-Rate Chemistry Effects in Nonpremixed Turbulent Combustion: Test on the Bluff-Body Stabilized Flame," *Combustion and Flame*, Vol. 110, 1997, pp. 298–318.

[187] Masri, A. R., Kelman, J. B., and Dally, B. B., "The Instantaneous Spatial Structure of the Recirculation Zone in Bluff-Body Stabilised Flames," *Proceedings of the Combustion Institute*, The Combustion Institute, 1998, pp. 1031–1038.

[188] Dally, B. B., Fletcher, D. F., and Masri, A. R., "Modelling of Turbulent Flames Stabilised on a Bluff-Body," *Combustion Theory and Modelling*, Vol. 2, 1998, pp. 193–219.

[189] Dally, B. B., Masri, A. R., Barlow, R. S., and Fiechtner, G. J., "Instantaneous and Mean Compositional Structure of Bluff-Body Stabilised Nonpremixed Flames," *Combustion Theory and Modelling*, Vol. 114, 1998, pp. 119–148.

[190] Turpin, G., , and Troyes, J., "Validation of a Two-Equation Turbulence Model for Axisymmetric Reacting and Non-Reacting Flows," Paper 2000-3463, AIAA, July 2000.

[191] Xu, J. and Pope, S. B., "PDF Calculations of Turbulent Nonpremixed Flames with Local Extinction," *Combustion and Flame*, Vol. 123, 200, pp. 281–307.

[192] Merci, B., Dick, E., Vierendeels, J., Roekaerts, D., and Peeters, T. W. J., "Application of a New Cubic Turbulence Model to Piloted and Bluff-Body Diffusion Flames," *Combustion and Flame*, Vol. 126, No. 1-2, 2001, pp. 1533–1556.

[193] Liu, K., Pope, S., and Caughey, D., "Calculations of Bluff-Body Stabilized Flames Using a Joint Probability Density Function Model with Detailed Chemistry," *Combustion and Flame*, Vol. 141, 2005, pp. 89–117.

[194] Merci, B., *An Unstructured Parallel Algorithm for Large Eddy and Conjugate Heat Transfer Simulations*, Ph.D. thesis, Ghent University, 2000.

Appendix A

Governing Equation Systems

Two-Dimensional Axisymmetric Formulation

The divergence form of Equation(3.1) is obtained by applying Gauss's theorem to the flux integral, leading to

$$\frac{\partial \mathbf{U}}{\partial t} + \left(\vec{\nabla} \cdot \vec{\mathbf{F}}\right) = \frac{\partial \mathbf{U}}{\partial t} + \left(\vec{\nabla} \cdot \vec{\mathbf{F}}_{\mathrm{I}}\right) + \left(\vec{\nabla} \cdot \vec{\mathbf{F}}_{\mathrm{V}}\right) = \mathbf{S}\,, \tag{A.1}$$

where ∇ is the well-known gradient operator. The complete set of equations is reformulated in conservation form for two dimensional axisymmetric coordinate frames as

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial (\mathbf{F} - \mathbf{F}_{\mathrm{v}})}{\partial r} + \frac{\partial (\mathbf{G} - \mathbf{G}_{\mathrm{v}})}{\partial z} = -\frac{(\mathbf{S}_{\mathrm{a}} - \mathbf{S}_{\mathrm{av}})}{r} + \mathbf{S}_{\mathrm{t}} + \mathbf{S}_{\mathrm{c}}\,, \tag{A.2}$$

where r and z denote the radial and axial coordinates, $\mathbf{U}$ the cell-averaged solution vector, $\mathbf{F}_I$ and $\mathbf{F}_V$ the inviscid and viscous flux vectors in radial direction, $\mathbf{G}_I$ and $\mathbf{G}_V$ the inviscid and viscous flux vectors in axial direction, $\mathcal{S}_{\mathbf{a}I}$, $\mathcal{S}_{\mathbf{a}V}$ are the source terms due to the axisymmetric coordinates, and $\mathbf{S}_{\mathrm{t}}$ and $\mathbf{S}_{\mathrm{c}}$ are the source terms due to turbulence and chemical reactions. Each of this term is given below.

$$\mathbf{U} = \left[\ \rho,\ \rho v_{\mathrm{r}},\ \rho v_{\mathrm{z}},\ \rho e,\ \rho k,\ \rho\omega, \rho c_1,\ ...,\ \rho c_{\mathrm{N}}\ \right]^{\mathrm{T}}$$

$$\mathbf{F} = \begin{bmatrix} \rho v_{\mathrm{r}} \\ \rho v_{\mathrm{r}^2} + p \\ \rho v_{\mathrm{r}} v_{\mathrm{z}} \\ (\rho e + p) v_{\mathrm{r}} \\ \rho k v_{\mathrm{r}} \\ \rho\omega v_{\mathrm{r}} \\ \rho c_1 v_{\mathrm{r}} \\ \vdots \\ \rho c_{\mathrm{N}} v_{\mathrm{r}} \end{bmatrix},\ \mathbf{F}_{\mathrm{v}} = \begin{bmatrix} 0 \\ \tau_{\mathrm{rr}} + \lambda_{\mathrm{rr}} \\ \tau_{\mathrm{rz}} + \lambda_{\mathrm{rz}} \\ -q_{\mathrm{r}} - q_{\mathrm{t_r}} + (\mu + \mu_{\mathrm{t}}\sigma^*)\frac{\partial k}{\partial r} + v_{\mathrm{r}}(\tau_{\mathrm{rr}} + \lambda_{\mathrm{rr}}) + v_{\mathrm{z}}(\tau_{\mathrm{rz}} + \lambda_{\mathrm{rz}}) \\ (\mu + \mu_{\mathrm{t}}\sigma^*)\frac{\partial k}{\partial r} \\ (\mu + \mu_{\mathrm{t}}\sigma)\frac{\partial \omega}{\partial r} \\ -\mathcal{J}_{1\mathrm{r}} + \mathcal{J}_{\mathrm{t}1\mathrm{r}} \\ \vdots \\ -\mathcal{J}_{\mathrm{N}r} + \mathcal{J}_{\mathrm{tNr}} \end{bmatrix}, \tag{A.3}$$

$$\mathbf{G}=\begin{bmatrix} \rho v_{\mathrm{z}} \\ \rho v_{\mathrm{r}} v_{\mathrm{z}} \\ \rho {v_{\mathrm{z}}}^2 + p \\ (\rho e + p) v_{\mathrm{z}} \\ \rho k v_{\mathrm{z}} \\ \rho \omega v_{\mathrm{z}} \\ \rho v_{\mathrm{z}} c_1 \\ \vdots \\ \rho v_{\mathrm{z}} c_{\mathrm{N}} \end{bmatrix}, \ \mathbf{G}_{\mathrm{v}}=\begin{bmatrix} 0 \\ \tau_{\mathrm{zr}} + \lambda_{\mathrm{zr}} \\ \tau_{\mathrm{zz}} + \lambda_{\mathrm{zz}} \\ -q_{\mathrm{z}} - q_{T_z} + (\mu + \mu_{\mathrm{t}}\sigma^*)\frac{\partial k}{\partial z} + v_r(\tau_{zr} + \lambda_{\mathrm{zr}}) + v_{\mathrm{z}}(\tau_{\mathrm{zz}} + \lambda_{\mathrm{zz}}) \\ (\mu + \mu_{\mathrm{t}}\sigma^*)\frac{\partial k}{\partial z} \\ (\mu + \mu_{\mathrm{t}}\sigma)\frac{\partial \omega}{\partial z} \\ -(\mathcal{J}_{T_1 z} + \mathcal{J}_{L_{1_z}}) \\ \vdots \\ -(\mathcal{J}_{T_N z} + \mathcal{J}_{L_{N_z}}) \end{bmatrix} \quad \text{(A.4)}$$

$$\mathbf{S}_{\mathrm{aI}}=\begin{bmatrix} \rho v_{\mathrm{r}} \\ \rho {v_{\mathrm{r}}}^2 \\ \rho v_{\mathrm{r}} v_{\mathrm{z}} \\ (\rho e + P) v_{\mathrm{z}} \\ \rho k v_{\mathrm{r}} \\ \rho \omega v_{\mathrm{r}} \\ \rho c_1 v_{\mathrm{r}} \\ \vdots \\ \rho c_{\mathrm{N}} v_{\mathrm{r}} \end{bmatrix}, \ \mathbf{S}_{\mathrm{aV}}=\begin{bmatrix} 0 \\ (\tau_{\mathrm{rr}} + \lambda_{\mathrm{rr}}) - (\tau_{\theta\theta} + \lambda_{\theta\theta}) \\ (\tau_{\mathrm{rz}} + \lambda_{\mathrm{rz}}) \\ -q_{\mathrm{r}} - q_{\mathrm{t_r}} + \mathcal{W} + \left(\mu + \mu_{\mathrm{t}}\sigma^*\right)\frac{\partial k}{\partial r} \\ (\mu + \mu_{\mathrm{t}}\sigma^*)\frac{\partial k}{\partial r} \\ (\mu + \mu_{\mathrm{t}}\sigma)\frac{\partial \omega}{\partial r} \\ \mathcal{J}_{1\mathrm{r}} + \mathcal{J}_{t1\mathrm{r}} \\ \vdots \\ \mathcal{J}_{N\mathrm{r}} + \mathcal{J}_{tN\mathrm{r}} \end{bmatrix}, \mathbf{S}=\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \mathcal{P} - \beta^* \rho k \omega \\ \alpha\frac{\omega}{k}\mathcal{P} - \beta\rho\omega^2 \\ \rho\dot{\omega_1} \\ \vdots \\ \rho\dot{\omega_{\mathrm{N}}} \end{bmatrix} . \quad \text{(A.5)}$$

In above equations, v_{r} and v_{z} are the r and z components of the mass-averaged velocity of the mixture, and the work done by the molecular stresses, $\mathcal{W}$ and production term, P, are given by $\mathcal{W} = v_{\mathrm{r}}(\tau_{\mathrm{rr}} + \lambda_{\mathrm{rr}}) + v_{\mathrm{z}}(\tau_{\mathrm{rz}} + \lambda_{\mathrm{rz}})$ and $\mathcal{P} = \lambda_{\mathrm{rr}}\frac{\partial v_{\mathrm{r}}}{\partial r} + \lambda_{\mathrm{rz}}\left(\frac{\partial v_{\mathrm{r}}}{\partial z} + \frac{\partial v_{\mathrm{z}}}{\partial r}\right) + \lambda_{\mathrm{zz}}\frac{\partial v_{\mathrm{z}}}{\partial z} + \lambda_{\theta\theta}\frac{v_{\mathrm{r}}}{r}$

Three-Dimensional Formulation

For three-dimensional flows, Equations (2.8)–(2.10), (2.12), (2.21), and (2.22) can be re-expressed using vector notation as

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial}{\partial x}\left(\mathbf{F} - \mathbf{F}_v\right) + \frac{\partial}{\partial y}\left(\mathbf{G} - \mathbf{G}_v\right) + \frac{\partial}{\partial z}\left(\mathbf{H} - \mathbf{H}_v\right) = \mathbf{S}\,, \quad \text{(A.6)}$$

where $\mathbf{U}$ is the vector of conserved variables given by

$$\mathbf{U} = \left[\ \rho,\ \rho v_{\mathrm{x}},\ \rho v_{\mathrm{y}},\ \rho v_{\mathrm{z}},\ \rho e,\ \rho k,\ \rho\omega,\ \rho c_1,\ \ldots,\ \rho c_{\mathrm{N}}\ \right]^{\mathrm{T}}\,, \quad \text{(A.7)}$$

and the inviscid and viscous x-direction flux vectors, $\mathbf{F}$ and $\mathbf{F}_{\mathrm{v}}$, can be written as

$$\mathbf{F}=\begin{bmatrix} \rho v_{\mathrm{x}} \\ \rho v_{\mathrm{x}}^2 + p \\ \rho v_{\mathrm{x}} v_{\mathrm{y}} \\ \rho v_{\mathrm{x}} v_{\mathrm{z}} \\ (\rho e + p) v_{\mathrm{x}} \\ \rho k v_{\mathrm{x}} \\ \rho \omega v_{\mathrm{x}} \\ \rho c_1 v_{\mathrm{x}} \\ \vdots \\ \rho c_{\mathrm{N}} v_{\mathrm{x}} \end{bmatrix}, \ \mathbf{F}_{\mathrm{v}}=\begin{bmatrix} 0 \\ \tau_{\mathrm{xx}} + \lambda_{\mathrm{xx}} \\ \tau_{\mathrm{xy}} + \lambda_{\mathrm{xy}} \\ \tau_{\mathrm{xz}} + \lambda_{\mathrm{xz}} \\ \mathcal{W}_{\mathrm{x}} - q_{\mathrm{x}} - q_{t_{\mathrm{x}}} + (\mu + \mu_{\mathrm{t}}\sigma^*)\frac{\partial k}{\partial x} \\ (\mu + \mu_{\mathrm{t}}\sigma^*)\frac{\partial k}{\partial x} \\ (\mu + \mu_{\mathrm{t}}\sigma)\frac{\partial \omega}{\partial x} \\ -\mathcal{J}_{1\mathrm{x}} - \mathcal{J}_{t1\mathrm{x}} \\ \vdots \\ -\mathcal{J}_{N\mathrm{x}} - \mathcal{J}_{tN\mathrm{x}} \end{bmatrix}, \tag{A.8}$$

$$\mathbf{G}=\begin{bmatrix} \rho v_{\mathrm{y}} \\ \rho v_{\mathrm{x}} v_{\mathrm{y}} \\ \rho v_{\mathrm{y}}^2 + p \\ \rho v_{\mathrm{y}} v_{\mathrm{z}} \\ (\rho e + p) v_{\mathrm{y}} \\ \rho k v_{\mathrm{y}} \\ \rho \omega v_{\mathrm{y}} \\ \rho c_1 v_{\mathrm{y}} \\ \vdots \\ \rho c_{\mathrm{N}} v_{\mathrm{y}} \end{bmatrix}, \ \mathbf{G}_{\mathrm{v}}=\begin{bmatrix} 0 \\ \tau_{\mathrm{yx}} + \lambda_{\mathrm{yx}} \\ \tau_{\mathrm{yy}} + \lambda_{\mathrm{yy}} \\ \tau_{\mathrm{yz}} + \lambda_{\mathrm{yz}} \\ \mathcal{W}_{\dagger} - q_{\mathrm{y}} - q_{t_{\mathrm{y}}} + (\mu + \mu_{\mathrm{t}}\sigma^*)\frac{\partial k}{\partial y} \\ (\mu + \mu_{\mathrm{t}}\sigma^*)\frac{\partial k}{\partial y} \\ (\mu + \mu_{\mathrm{t}}\sigma)\frac{\partial \omega}{\partial y} \\ -\mathcal{J}_{1\mathrm{y}} - \mathcal{J}_{t1\mathrm{y}} \\ \vdots \\ -\mathcal{J}_{N\mathrm{y}} - \mathcal{J}_{tN\mathrm{y}} \end{bmatrix}, \tag{A.9}$$

$$\mathbf{H}=\begin{bmatrix} \rho v_{\mathrm{z}} \\ \rho v_{\mathrm{z}} v_{\mathrm{x}} \\ \rho v_{\mathrm{z}} v_{\mathrm{y}} \\ \rho v_{\mathrm{z}}^2 + p \\ (\rho e + p) v_{\mathrm{z}} \\ \rho k v_{\mathrm{z}} \\ \rho \omega v_{\mathrm{z}} \\ \rho c_1 v_{\mathrm{z}} \\ \vdots \\ \rho c_{\mathrm{N}} v_{\mathrm{z}} \end{bmatrix}, \ \mathbf{H}_{\mathrm{v}}=\begin{bmatrix} 0 \\ \tau_{\mathrm{zx}} + \lambda_{\mathrm{zx}} \\ \tau_{\mathrm{zy}} + \lambda_{\mathrm{zy}} \\ \tau_{\mathrm{zz}} + \lambda_{\mathrm{zz}} \\ \mathcal{W}_{\mathrm{z}} - q_{\mathrm{z}} - q_{t_{\mathrm{z}}} + (\mu + \mu_{\mathrm{t}}\sigma^*)\frac{\partial k}{\partial z} \\ (\mu + \mu_{\mathrm{t}}\sigma^*)\frac{\partial k}{\partial z} \\ (\mu + \mu_{\mathrm{t}}\sigma)\frac{\partial \omega}{\partial z} \\ -\mathcal{J}_{1\mathrm{z}} - \mathcal{J}_{t1\mathrm{z}} \\ \vdots \\ -\mathcal{J}_{N\mathrm{z}} - \mathcal{J}_{tN\mathrm{z}} \end{bmatrix}, \tag{A.10}$$

where $\mathcal{W}_{\mathrm{x}} = v_{\mathrm{x}}(\tau_{\mathrm{xx}} + \lambda_{\mathrm{xx}}) + v_{\mathrm{y}}(\tau_{\mathrm{xy}} + \lambda_{\mathrm{xy}}) + v_{\mathrm{z}}(\tau_{\mathrm{xz}} + \lambda_{\mathrm{xz}})$, $\mathcal{W}_{\mathrm{y}} = v_{\mathrm{x}}(\tau_{\mathrm{xy}} + \lambda_{\mathrm{xy}}) + v_{\mathrm{y}}(\tau_{\mathrm{yy}} + \lambda_{\mathrm{yy}}) + v_{\mathrm{z}}(\tau_{\mathrm{yz}} + \lambda_{\mathrm{yz}})$, and $\mathcal{W}_{\mathrm{z}} = v_{\mathrm{x}}(\tau_{\mathrm{xz}} + \lambda_{\mathrm{xz}}) + v_{\mathrm{y}}(\tau_{\mathrm{yz}} + \lambda_{\mathrm{yz}}) + v_{\mathrm{z}}(\tau_{\mathrm{zz}} + \lambda_{\mathrm{zz}})$. The y- and z-direction flux vectors $\mathbf{G}$, $\mathbf{G}_v$, $\mathbf{H}$, and $\mathbf{H}_v$ have similar forms. The source vector, $\mathbf{S}$ contains terms related to the finite rate chemistry, body force due to gravity, and turbulence modeling and has the form

$$\mathbf{S} = \left[\ 0,\, 0,\, 0,\, 0,\, 0,\, \mathcal{P} - \beta^* \rho k \omega,\, \alpha\tfrac{\omega}{k}\mathcal{P} - \beta\rho\omega^2,\, \rho\dot{\omega}_1,\, \ldots,\, \rho\dot{\omega}_{\mathrm{N}} \ \right], \tag{A.11}$$

with

$$\mathcal{P} = \lambda_{\mathrm{xx}}\frac{\partial v_{\mathrm{x}}}{\partial x} + \lambda_{\mathrm{xy}}(\frac{\partial v_{\mathrm{x}}}{\partial y} + \frac{\partial v_{\mathrm{y}}}{\partial x}) + \lambda_{\mathrm{yy}}\frac{\partial v_{\mathrm{y}}}{\partial y} + \lambda_{\mathrm{xz}}(\frac{\partial v_{\mathrm{z}}}{\partial z} + \frac{\partial v_{\mathrm{z}}}{\partial x}) + \lambda_{\mathrm{yz}}(\frac{\partial v_{\mathrm{y}}}{\partial z} + \frac{\partial v_{\mathrm{z}}}{\partial y}) + \lambda_{\mathrm{zz}}\frac{\partial v_{\mathrm{z}}}{\partial z}\,, \tag{A.12}$$

and where x, y, and z are the coordinates of the three dimensional Cartesian frame; v_{x}, v_{y}, and v_{z} are the x, y, and z velocity components; q_{x}, q_{y}, and q_{z} are the x, y, and z components of the heat flux; τ_{xx}, τ_{xy}, τ_{yy}, τ_{xz}, τ_{yz}, and τ_{zz} are the components of the viscous fluid stresses; and λ_{xx}, λ_{xy}, λ_{xz}, λ_{xz}, λ_{yz}, and λ_{zz} are the Reynolds stresses.

Appendix B

Eigensystem of the Inviscid Jacobian

The application of the numerical solution scheme considered in this work requires knowledge of the flux Jacobian matrices and their associated eigenvalues and eigenvectors. The flux Jacobian matrices are $A = \frac{\partial \mathbf{F(U)}}{\partial \mathbf{U}}$, $B = \frac{\partial \mathbf{G(U)}}{\partial \mathbf{U}}$ and $C = \frac{\partial \mathbf{H(U)}}{\partial \mathbf{U}}$. In order to obtain the flux Jacobian matrices, the pressure derivatives are needed and described below.

The computation of pressure is from total energy ρe and equation of state. The derivatives of pressure with respect to the conserved variables will be used to compute the flux Jacobian matrices.

The total energy, ρe, is defined as

$$\rho e = \rho \left[\frac{1}{2}(u^2 + v^2 + w^2) + \sum_{n=1}^{N} c_n \epsilon_n + k \right] \tag{B.1}$$

where

$$\epsilon_n = \int_{T_o}^{T} C_{vn}(\tau) d\tau + h_{fn}^0$$

The equation of state for multispecies is written as

$$p = \rho \sum_{n=1}^{N} c_n R_n T \tag{B.2}$$

Taking the total derivative of equation (B.1) with respect to the conservative variables, and rearranging the terms, they can be written as follows:

$$d(\rho e) = \frac{v_x^2 + v_y^2 + v_z^2}{-2} d(\rho) +$$

$$v_x d(\rho v_x) + v_y d(\rho v_y) + v_z d(\rho v_z) + \sum_{n=1}^{N} \epsilon_n d(\rho c_n) + \rho C_v d(T) + d(\rho k) \tag{B.3}$$

where $C_v = \sum_{n=1}^{N} c_n C_{vn}$. Taking the total derivative of equation (B.2) with respect to the conservative variables, and rearranging the terms, they can be written as follows:

$$dp = \rho R d(T) + \sum_{n=1}^{N} R_n T d(\rho c_n) \tag{B.4}$$

where $R = \sum_{n=1}^{N} c_n R_n$. This equation maybe recast equation (B.4) as:

$$dT = \frac{1}{\rho R}\left[dp - \sum_{n=1}^{N} R_n T d(\rho c_n)\right] \tag{B.5}$$

and then substitute equation (B.5) into equation (B.3)to arrive at:

$$dp = \frac{R}{C_v}\Big[\big(\frac{v_x^2 + v_y^2 + v_z^2}{2}\big)d(\rho) - v_x d(\rho v_x) - v_y d(\rho v_y) - v_z d(\rho v_z) + d(\rho e) - d(\rho k) - \sum_{n=1}^{N}\big(\epsilon_n - \frac{C_v R_n T}{R}\big)d(\rho c_n)\Big] \tag{B.6}$$

It is more straightforward to derive the Jacobian matrices and their associated eigenvalues and eigenvectors for governing equation system written in non-conservative formulation. We rewrite governing equation system (Equation 3.9)in terms of primitive parameters as follows:

$$\frac{\partial \mathbf{W}}{\partial t} + \mathcal{A}\frac{\partial \mathbf{W}}{\partial x} + \mathcal{B}\frac{\partial \mathbf{W}}{\partial y} + \mathcal{C}\frac{\partial \mathbf{W}}{\partial z} = \frac{\partial \mathbf{W}}{\partial \mathbf{U}}\mathbf{S}, \tag{B.7}$$

where $\mathcal{A} = \left(\frac{\partial \mathbf{W}}{\partial \mathbf{U}}\frac{\partial \mathbf{F}}{\partial \mathbf{U}}\frac{\partial \mathbf{U}}{\partial \mathbf{W}}\right)$, $\mathcal{B} = \left(\frac{\partial \mathbf{W}}{\partial \mathbf{U}}\frac{\partial \mathbf{G}}{\partial \mathbf{U}}\frac{\partial \mathbf{U}}{\partial \mathbf{W}}\right)$ and $\mathcal{C} = \left(\frac{\partial \mathbf{W}}{\partial \mathbf{U}}\frac{\partial \mathbf{H}}{\partial \mathbf{U}}\frac{\partial \mathbf{U}}{\partial \mathbf{W}}\right)$, and

$$\begin{aligned}\mathbf{W} &= \left[\ \rho,\ v_x,\ v_y,\ w_z,\ p,\ k,\ \omega, c_1,\ c_2,\ \cdots,\ c_N\ \right]^T,\\ \mathbf{U} &= \left[\ \rho,\ \rho v_x,\ \rho v_y,\ \rho v_z,\ \rho e,\ \rho k,\ \rho\omega, \rho c_1,\ \rho c_2,\ \cdots,\ \rho c_N\ \right]^T\\ &= \left[\ \mathbf{U}_1,\ \mathbf{U}_2,\ \mathbf{U}_3,\ \mathbf{U}_4,\ \mathbf{U}_5,\ \mathbf{U}_6,\ \mathbf{U}_7,\ \mathbf{U}_8,\ \mathbf{U}_9,\ \cdots,\ \mathbf{U}_{N+7}\ \right]^T\end{aligned}$$

Flux vectors ($\mathbf{F}$, $\mathbf{G}$,$\mathbf{H}$) can be written in a general form in terms of the conserved

variables as

$$\vec{\mathcal{F}} = \begin{bmatrix} \mathbf{U}_2\vec{n_i} + \mathbf{U}_3\vec{n_j} + \mathbf{U}_4\vec{n_k} \\ \mathbf{U}_2\left(\frac{\mathbf{U}_2}{\mathbf{U}_1}\vec{n_i} + \frac{\mathbf{U}_3}{\mathbf{U}_1}\vec{n_j} + \frac{\mathbf{U}_4}{\mathbf{U}1}\vec{n_k}\right) + p\vec{n_i} \\ \mathbf{U}_3\left(\frac{\mathbf{U}_2}{\mathbf{U}_1}\vec{n_i} + \frac{\mathbf{U}_3}{\mathbf{U}_1}\vec{n_j} + \frac{\mathbf{U}_4}{\mathbf{U}1}\vec{n_k}\right) + p\vec{n_j} \\ \mathbf{U}_4\left(\frac{\mathbf{U}_2}{\mathbf{U}_1}\vec{n_i} + \frac{\mathbf{U}_3}{\mathbf{U}_1}\vec{n_j} + \frac{\mathbf{U}_4}{\mathbf{U}1}\vec{n_k}\right) + p\vec{n_k} \\ \left(\mathbf{U}_5 + p\right)\left(\frac{\mathbf{U}_2}{\mathbf{U}_1}\vec{n_i} + \frac{\mathbf{U}_3}{\mathbf{U}_1}\vec{n_j} + \frac{\mathbf{U}_4}{\mathbf{U}1}\vec{n_k}\right) \\ \mathbf{U}_6\left(\frac{\mathbf{U}_2}{\mathbf{U}_1}\vec{n_i} + \frac{\mathbf{U}_3}{\mathbf{U}_1}\vec{n_j} + \frac{\mathbf{U}_4}{\mathbf{U}1}\vec{n_k}\right) \\ \mathbf{U}_7\left(\frac{\mathbf{U}_2}{\mathbf{U}_1}\vec{n_i} + \frac{\mathbf{U}_3}{\mathbf{U}_1}\vec{n_j} + \frac{\mathbf{U}_4}{\mathbf{U}1}\vec{n_k}\right) \\ \mathbf{U}_{1+7}\left(\frac{\mathbf{U}_2}{\mathbf{U}_1}\vec{n_i} + \frac{\mathbf{U}_3}{\mathbf{U}_1}\vec{n_j} + \frac{\mathbf{U}_4}{\mathbf{U}1}\vec{n_k}\right) \\ \cdots \\ \cdots \\ \cdots \\ \mathbf{U}_{N+7}\left(\frac{\mathbf{U}_2}{\mathbf{U}_1}\vec{n_i} + \frac{\mathbf{U}_3}{\mathbf{U}_1}\vec{n_j} + \frac{\mathbf{U}_4}{\mathbf{U}1}\vec{n_k}\right) \end{bmatrix} \tag{B.8}$$

where $\vec{n_i}$, $\vec{n_j}$ and $\vec{n_k}$ are unit norm vector. Flux Jacobian matrices, A, B and C, have a general formula as:

$$\frac{\partial\vec{\mathbf{F}}}{\partial\mathbf{U}} = \begin{bmatrix} . & \vec{n_i} & \vec{n_j} & \vec{n_k} & . & . & . & . & . & \cdots & . \\ A_{21} & A_{22} & A_{23} & A_{24} & \frac{\partial p}{\partial\mathbf{U}_5}\vec{n_i} & \frac{\partial p}{\partial\mathbf{U}_6}\vec{n_i} & . & \frac{\partial p}{\partial\mathbf{U}_8}\vec{n_i} & \frac{\partial p}{\partial\mathbf{U}_9}\vec{n_i} & \cdots & \frac{\partial p}{\partial\mathbf{U}_{N+7}}\vec{n_i} \\ A_{31} & A_{32} & A_{33} & A_{34} & \frac{\partial p}{\partial\mathbf{U}_5}\vec{n_j} & \frac{\partial p}{\partial\mathbf{U}_6}\vec{n_j} & . & \frac{\partial p}{\partial\mathbf{U}_8}\vec{n_j} & \frac{\partial p}{\partial\mathbf{U}_9}\vec{n_j} & \cdots & \frac{\partial p}{\partial\mathbf{U}_{N+7}}\vec{n_j} \\ A_{41} & A_{42} & A_{43} & A_{44} & \frac{\partial p}{\partial\mathbf{U}_5}\vec{n_k} & \frac{\partial p}{\partial\mathbf{U}_6}\vec{n_k} & . & \frac{\partial p}{\partial\mathbf{U}_8}\vec{n_k} & \frac{\partial p}{\partial\mathbf{U}_9}\vec{n_k} & \cdots & \frac{\partial p}{\partial\mathbf{U}_{N+7}}\vec{n_k} \\ A_{51} & A_{52} & A_{53} & A_{54} & \vec{V}(1+\frac{\partial p}{\partial\mathbf{U}_5}) & \vec{V}\frac{\partial p}{\partial\mathbf{U}_6} & . & \vec{V}\frac{\partial p}{\partial\mathbf{U}_8} & \vec{V}\frac{\partial p}{\partial\mathbf{U}_9} & . & \vec{V}\frac{\partial p}{\partial\mathbf{U}_{N+7}} \\ -k\vec{V} & k\vec{n_i} & k\vec{n_j} & k\vec{n_k} & . & \vec{V} & . & . & . & \cdots & . \\ -\omega\vec{V} & \omega\vec{n_i} & \omega\vec{n_j} & \omega\vec{n_k} & . & . & \vec{V} & . & . & \cdots & . \\ \frac{-c_1}{\rho}\vec{V} & \frac{1}{\rho}\vec{n_i} & \frac{1}{\rho}\vec{n_j} & \frac{1}{\rho}\vec{n_k} & . & . & . & \vec{V} & . & \cdots & . \\ \frac{-c_2}{\rho}\vec{V} & \frac{1}{\rho}\vec{n_i} & \frac{1}{\rho}\vec{n_j} & \frac{1}{\rho}\vec{n_k} & . & . & . & . & \vec{V} & \cdots & . \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{-c_N}{\rho}\vec{V} & \frac{1}{\rho}\vec{n_i} & \frac{1}{\rho}\vec{n_j} & \frac{1}{\rho}\vec{n_k} & . & . & . & . & . & . & \vec{V} \end{bmatrix}$$

where

$$\begin{bmatrix} A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \\ A_{51} & A_{52} & A_{53} & A_{54} \end{bmatrix}$$

is shown in details by

$$\begin{bmatrix} -v_x\vec{V} + \frac{\partial p}{\partial \mathbf{U}_1}\vec{n_i} & v_x\vec{n_i} + \vec{V} + \frac{\partial p}{\partial \mathbf{U}_2}\vec{n_i} & v_x\vec{n_j} + \frac{\partial p}{\partial \mathbf{U}_3}\vec{n_i} & v_x\vec{n_k} + \frac{\partial p}{\partial \mathbf{U}_4}\vec{n_i} \\ -v_y\vec{V} + \frac{\partial p}{\partial \mathbf{U}_1}\vec{n_j} & v_y\vec{n_i} + \frac{\partial p}{\partial \mathbf{U}_2}\vec{n_j} & \vec{V} + v_y\vec{n_j} + \frac{\partial p}{\partial \mathbf{U}_3}\vec{n_j} & v_y\vec{n_k} + \frac{\partial p}{\partial \mathbf{U}_4}\vec{n_j} \\ -v_z\vec{V} + \frac{\partial p}{\partial \mathbf{U}_1}\vec{n_k} & v_z\vec{n_i} + \frac{\partial p}{\partial \mathbf{U}_2}\vec{n_k} & v_z\vec{n_j} + \frac{\partial p}{\partial \mathbf{U}_3}\vec{n_k} & \vec{V} + v_z\vec{n_k} + \frac{\partial p}{\partial \mathbf{U}_4}\vec{n_k} \\ \vec{V}[\frac{\partial p}{\partial \mathbf{U}_1} - (e + \frac{p}{\rho})] & \vec{V}\frac{\partial p}{\partial \mathbf{U}_2} + (e + \frac{p}{\rho})\vec{n_i} & \vec{V}\frac{\partial p}{\partial \mathbf{U}_3} + (e + \frac{p}{\rho})\vec{n_j} & \vec{V}\frac{\partial p}{\partial \mathbf{U}_4} + (e + \frac{p}{\rho})\vec{n_k} \end{bmatrix}$$

where

$$\vec{V} = v_x\vec{n_i} + v_y\vec{n_j} + v_z\vec{n_k} \quad .$$

The elements in $\frac{\partial \mathbf{U}}{\partial \mathbf{W}}$ and $\left(\frac{\partial \mathbf{U}}{\partial \mathbf{W}}\right)^{-1}$ are given by

$$\frac{\partial \mathbf{U}}{\partial \mathbf{W}} = \begin{bmatrix} 1 & . & . & . & . & . & . & . & . & . & . \\ v_x & \rho & . & . & . & . & . & . & . & . & . \\ v_y & . & \rho & . & . & . & . & . & . & . & . \\ v_z & . & 0 & \rho & . & . & . & . & . & . & . \\ \frac{(v_x^2+v_y^2+v_z^2)}{2} + k + \epsilon - C_vT & \rho v_x & \rho v_y & \rho v_z & \frac{C_v}{R} & \rho & 0 & \rho\eta_1 & \rho\eta_2 & . & \rho\eta_N \\ k & . & . & . & . & \rho & . & . & . & . & . \\ \omega & . & . & . & . & . & \rho & . & . & . & . \\ . & . & . & . & . & . & . & 1 & . & . & . \\ . & . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & 1 & \\ . & . & . & . & . & . & . & . & . & . & 1 \end{bmatrix},$$

and

$$\frac{\partial \mathbf{W}}{\partial \mathbf{U}} = \begin{bmatrix} 1 & . & . & . & . & . & . & . & . & . & . \\ -\frac{v_x}{\rho} & \frac{1}{\rho} & . & . & . & . & . & . & . & . & . \\ -\frac{v_y}{\rho} & . & \frac{1}{\rho} & . & . & . & . & . & . & . & . \\ -\frac{v_z}{\rho} & . & . & \frac{1}{\rho} & . & . & . & . & . & . & . \\ \left(\frac{v_x^2+v_y^2+v_z^2R}{2C_v}\right) & \frac{-R}{C_v}v_x & \frac{-R}{C_v}v_y & \frac{-R}{C_v}v_z & \frac{-R}{C_v} & \frac{-R}{C_v} & 0 & \frac{-R}{C_v}\eta_1 & \frac{-R}{C_v}\eta_2 & . & \frac{-R}{C_v}\eta_N \\ -\frac{k}{\rho} & . & . & . & . & \frac{1}{\rho} & . & . & . & . & . \\ -\frac{\omega}{\rho} & . & . & . & . & . & \frac{1}{\rho} & . & . & . & . \\ . & . & . & . & . & . & . & 1 & . & . & . \\ . & . & . & . & . & . & . & . & . & . & 1 \end{bmatrix}.$$

Jacobian matrices A, B and C, are similar to matrices $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$, so the eigenvalues are the same for both kinds of matrices. The eigenvalues were obtained from matrices $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$.

The eigenvalues for Jacobian matrices are:

$$\lambda_1 = \lambda_2 = \lambda_3 = \cdots = \lambda_n = v_x\vec{n_i} + v_y\vec{n_j} + v_z\vec{n_k} \quad \lambda_{n+1} = \lambda_1 + a \quad \lambda_{n+2} = \lambda_1 - a\,.$$

The right eigenvector matrix for flux Jacobian A is written as:

$$\begin{bmatrix}
1 & 1 & . & . & 1 & . & . & . & . & \cdots & . \\
v_x & v_x - a & . & . & v_x + a & . & . & . & . & \cdots & . \\
v_y & v_y & \rho & . & v_y & . & . & . & . & \cdots & . \\
v_z & v_z & . & \rho & v_z & . & . & . & . & \cdots & . \\
\mathcal{H} - C_pT & \mathcal{H} - v_xa & \rho v_y & \rho v_z & \mathcal{H} + v_xa & \rho & 0 & \rho\eta_1 & \rho\eta_2 & \cdots & \rho\eta_N \\
k & k & . & . & k & \rho & . & . & . & \cdots & . \\
\omega & \omega & . & . & \omega & . & \rho & . & . & \cdots & . \\
c_1 & c_1 & . & . & c_1 & . & . & \rho & . & \cdots & . \\
c_2 & c_2 & . & . & c_2 & . & . & . & \rho & \cdots & . \\
 & & & & & & & & & & \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
c_N & c_N & . & . & c_N & . & . & . & . & \cdots & \rho
\end{bmatrix}.$$

The right eigenvector matrix for flux Jacobian B is written as:

$$\begin{bmatrix}
1 & . & 1 & . & 1 & . & . & . & . & \cdots & . \\
v_x & \rho & v_x & . & v_x & . & . & . & . & \cdots & . \\
v_y & . & v_y - a & . & v_y + a & . & . & . & . & \cdots & . \\
v_z & . & v_z & \rho & v_z & . & . & . & . & \cdots & . \\
\mathcal{H} - C_pT & \rho v_x & \mathcal{H} - v_ya & \rho v_z & \mathcal{H} + va & \rho & 0 & \rho\eta_1 & \rho\eta_2 & \cdots & \rho\eta_N \\
k & . & k & . & k & \rho & . & . & . & \cdots & . \\
\omega & . & \omega & . & \omega & . & \rho & . & . & \cdots & . \\
c_1 & . & c_1 & . & c_1 & . & . & \rho & . & \cdots & . \\
c_2 & . & c_2 & . & c_2 & . & . & . & \rho & \cdots & . \\
 & & & & & & & & & & \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
c_N & . & c_N & . & c_N & . & . & . & . & \cdots & \rho
\end{bmatrix}.$$

The right eigenvector matrix for flux Jacobian C is written as:

$$\begin{bmatrix}
1 & . & . & 1 & 1 & . & . & . & . & \cdots & . \\
v_x & \rho & . & v_x & v_x & . & . & . & . & \cdots & . \\
v_y & . & \rho & v_y & v_y & . & . & . & . & \cdots & . \\
v_z & . & . & v_z - a & v_z + a & . & . & . & . & \cdots & . \\
\mathcal{H} - C_p T & \rho v_x & \rho v_y & \mathcal{H} + v_z a & \mathcal{H} + v_z a & \rho & 0 & \rho\eta_1 & \rho\eta_2 & \cdots & \rho\eta_N \\
k & . & . & k & k & \rho & . & . & . & \cdots & . \\
\omega & . & . & \omega & \omega & . & \rho & . & . & \cdots & . \\
c_1 & . & . & c_1 & c_1 & . & . & \rho & . & \cdots & . \\
c_2 & . & . & c_2 & c_2 & . & . & . & \rho & \cdots & . \\
 & & & & & & & & & & \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
c_N & . & . & c_N & c_N & . & . & . & . & \cdots & \rho
\end{bmatrix},$$

where $a = \sqrt{RT\left(1 + \frac{R}{C_v}\right)}$, $\eta_i = \epsilon_i - \frac{C_v R_i T}{R}$ and the specific enthalpy is $\mathcal{H} = \frac{v_x^2}{2} + \frac{v_y^2}{2} + \frac{v_z^2}{2} + h + k$, $h = \epsilon + \frac{p}{\rho}$, where ϵ is the specific internal energy.