A Parallel Implicit Adaptive Mesh Refinement Algorithm for Predicting Unsteady Fully-Compressible Reactive Flows

by

Scott A. Northrup

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

Abstract

A Parallel Implicit Adaptive Mesh Refinement Algorithm
for Predicting Unsteady Fully-Compressible Reactive Flows

Scott A. Northrup
Doctor of Philosophy
Graduate Department of Aerospace Engineering
University of Toronto
2014

A new parallel implicit adaptive mesh refinement (AMR) algorithm is developed for the prediction of unsteady behaviour of laminar flames. The scheme is applied to the solution of the system of partial-differential equations governing time-dependent, two- and three-dimensional, compressible laminar flows for reactive thermally perfect gaseous mixtures. A high-resolution finite-volume spatial discretization procedure is used to solve the conservation form of these equations on body-fitted multi-block hexahedral meshes. A local preconditioning technique is used to remove numerical stiffness and maintain solution accuracy for low-Mach-number, nearly incompressible flows. A flexible block-based octree data structure has been developed and is used to facilitate automatic solution-directed mesh adaptation according to physics-based refinement criteria. The data structure also enables an efficient and scalable parallel implementation via domain decomposition. The parallel implicit formulation makes use of a dual-time-stepping like approach with an implicit second-order backward discretization of the physical time, in which a Jacobian-free inexact Newton method with a preconditioned generalized minimal residual (GMRES) algorithm is used to solve the system of nonlinear algebraic equations arising from the temporal and spatial discretization procedures. An additive Schwarz global preconditioner is used in conjunction with block incomplete LU type local preconditioners for each sub-domain. The Schwarz preconditioning and block-based data structure readily allow efficient and scalable parallel implementations of the implicit AMR approach on

distributed-memory multi-processor architectures. The scheme was applied to solutions of steady and unsteady laminar diffusion and premixed methane-air combustion and was found to accurately predict key flame characteristics. For a premixed flame under terrestrial gravity, the scheme accurately predicted the frequency of the natural buoyancy induced oscillations. The performance of the proposed parallel implicit algorithm was assessed by comparisons to more conventional solution procedures and was found to significantly reduce the computational time required to achieve a solution in all cases investigated.

Acknowledgements

I feel privileged to have had the opportunity to further my education at the University of Toronto Institute for Aerospace Studies. The academic environment and world class research being done made the experience enjoyable, educational, and extremely rewarding.

I would like to thank my thesis supervisor, Professor Clinton Groth for his encouragement, guidance, and perseverance that made this thesis possible. I would also like to thank my thesis committee members Professors David Zingg and Ömer Gülder for their invaluable feedback and suggestions.

I am grateful to my UTIAS colleagues for providing an enjoyable work environment and for the many technical and non-technical discussions over the years. I would like to especially thank Dr. Jai Sachdev, Dr. Stephen Guzik, Dr. James McDonald, Dr. Marc Charest, and Dr. Jason Hicken.

I would like to thank all my friends and family for their support and encouragement. A special thanks to my wife, Renee Northrup for her love, devotion, and patience over the years. Also to my daughters; Suzanne, Danielle, and Madeline for just being themselves.

I am very grateful to the University of Toronto, the National Science and Engineering Research Council (NSERC), the Ontario Graduate Scholarship program, and Professor Groth for their financial support.

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

Scott Andrew Northrup
University of Toronto Institute for Aerospace Studies
2013

Contents

List of Tables

List of Figures

"If I have seen further it is by standing on ye sholders of Giants."

Sir Isaac Newton – 1675

Chapter 1

Introduction

1.1 Motivation

Advanced combustion science is and will remain a key component for continued economic growth and social stability of todays industrial societies and will play a vital role in the design of future energy utilization systems including electric power generation, transportation, and heating systems. Virtually all practical combustion devices operate in the turbulent regime and currently involve the burning of fossil fuels with high carbon content such as coal, petroleum, and natural gas. Unfortunately, it is the combustion of these fossil fuels that is responsible for nearly all of the anthropogenic emissions of nitrogen oxides (NOx), carbon monoxide (CO), soot, aerosols, and other chemical species that are harmful, or are suspected to be harmful, to human health and the environment [1]. The production of carbon dioxide (CO_2) by fossil fuel combustion is also a major contributor to climate change. In an effort to deal with emerging environmental concerns and policies set forth by governmental agencies as well as for reasons of energy security, the need for high-fuel-efficient, low-emissions combustion technology as well as alternative and sustainable sources of fuels have been pushed to the forefront of both national and international attention.

To address these pressing issues, a significantly enhanced fundamental understanding of combustion processes, specifically low-temperature, lean, turbulent combustion while avoiding thermoacoustic instabilities is needed. Combustion however, is an inherently multi-scale process involving the complex interplay of a wide range of non-linear phys-

ical processes including turbulent fluid transport, complex chemical kinetics of multi-component fuels, multi-phase transport, and radiation transport under conditions of elevated pressure and/or temperature and, as with all multi-scale processes, small-scale physics directly impacts observed large-scale behaviour. It is also important to note that, for devices such as gas turbine engines, future combustion designs will require operation at even higher pressures and fuel-to-air ratios than those seen today. Our fundamental understanding of conventional and alternative fuel combustion under such high-pressure conditions must therefore be significantly improved and the current lack of understanding in this area is one of the major impediments to the adoption of alternative fuels in gas turbines. Moreover the control of thermoacoustic combustion instabilities is one of the greatest challenges faced by today's designers of advanced low-emissions gas turbine engines. It is therefore evident that the complexity of the physics and wide range of physical scales make the understanding of combustion an extremely daunting and challenging task.

Numerical methods offer one avenue for improving our understanding of combustion; however, the development of predictive mathematical models and numerical simulation tools for realistic combustion devices is highly challenging [2]. As noted above, a broad array of physical and chemical models must be integrated into a single mathematical description of the combustion device. Moreover, practical systems involve three-dimensional, time-dependent, flows through complex geometries and often with moving machinery, as in gas turbines and internal combustion engines. Due to the limits on available computational resources and the inability to resolve all solution scales for practical configurations, numerical prediction of combustion processes relies heavily on reduced mathematical modelling and sophisticated numerical methods of varying degrees of approximation to represent the underlying physics and make the problems of interest tractable [3–5].

Unfortunately, current mathematical modelling techniques and numerical solution algorithms are not sufficiently accurate, reliable, robust, and/or scalable enough to address the numerous and complex issues associated with high-efficiency and low-emissions combustor design, such as high pressure and thermoacoustic effects. A common limitation in many current combustion modelling approaches is that they have been predominantly developed with a very specific flow regime or application in mind. As such mathematical modelling assumptions are made that limit their applicability outside of a specific regime. For example, a commonly used practice in numerical combustion modelling, including

the widely used KIVA [6] and OpenFOAM [7] codes, as well as the work of Smooke [8,9], Bennett [10], and Dworkin [11], is to solve the incompressible form of the Navier-Stokes equations and rely on compressibility corrections to account for density variations. While these approaches are adequate for relatively simple low-Mach-number atmospheric combustion regimes such as laminar diffusion flames, they however greatly limit the range of applicability of the method, making them not well suited for investigating high pressure and certainly unsteady thermal acoustic phenomena. To accurately resolve and investigate these more complex physical phenomena a fully compressible formulation is generally required [12]. While some fully compressible approaches for reactive flows have been developed [12–15], this still represents very much an active area of research as necessitated by the physical complexities being investigated. Ultimately if quantitative predictions of unsteady transient behaviour, such as thermoacoustic coupling and combustion instabilities, are to be undertaken, an efficient fully-compressible robust highly scalable algorithm will be required.

1.2 Numerical Solution Methods for Combustion Modelling

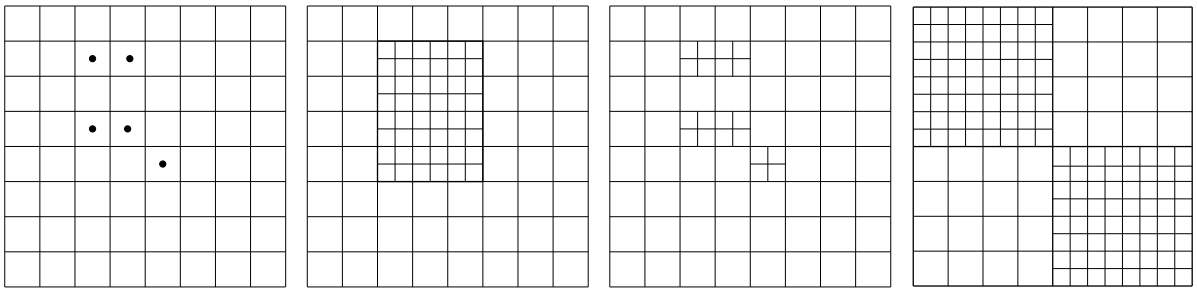
As mentioned in Section 1.1, most practical combustion configurations are fully turbulent, three dimensional, and involve complex chemical reaction mechanisms. The spatial and temporal time scales involved in the fluid flow, turbulence, and chemical reactions can be quite disparate, ranging over multiple order of magnitudes. The chemical reaction mechanisms that describe chemical processes are highly non-linear in nature and can involve many chemical species. In many cases, each species has to be tracked separately and a large system of non-linear partial differential equations must be solved. The disparate scales, non-linearity, and large systems of equations cause many numerical algorithms to be either very inaccurate or to take an inordinate amount of computational resources both in terms of time and storage, to achieve an accurate solution. The computational costs required can thus be extremely prohibitive and many flow regimes and/or conditions are not obtainable with current approaches.

1.2.1 Adaptive Mesh Refinement

One highly successful approach to reducing the computational cost of Computational Fluid Dynamic (CFD) solutions is to make use of solution-directed mesh adaptation where the underlying computational mesh automatically adapts according to the solution, adding mesh resolution only where required. In unsteady calculations the benefit is even more pronounced as mesh resolution is both increased and decreased dynamically allowing the mesh to track solution flow features. A recent assessment of the needs for large-scale and high-performance scientific computing [16, 17] has identified the need for greater automation of mesh generation via adaptive mesh refinement (AMR) to reduce the time to generate high-quality meshes and for the treatment of problems having complex geometries. Adaptive mesh refinement has proven to be very effective for treating problems with disparate length scales, providing the required spatial resolution while minimizing memory and storage requirements. When combined with Godunov-type finite-volume schemes [18], AMR methods have proved to be particularly effective for the solution of hyperbolic systems of conservation laws on structured Cartesian and body-fitted meshes and have been developed for a wide variety of engineering problems [14, 19–47].

To date, several distinct AMR strategies have emerged which can be generally classified into four broad categories depending on the partitioning algorithm and/or data structure used to track mesh connectivity: patch-based AMR methods, cell-based AMR methods, block-based AMR methods, and hybrid block-based AMR techniques. Figure 1.1(a) depicts a base Cartesian mesh with cells flagged 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 schemes.

Berger and Oliger, along with Colella, originally proposed a dynamic gridding technique for computing time-dependent solutions to hyperbolic partial differential equations (PDEs) in multiple space dimensions on regular Cartesian meshes [19–21]. This approach is now more generally referred to as patch-based AMR. The algorithm begins with a coarse base-level Cartesian grid and, as the calculation progresses, individual grid cells are flagged for refinement as illustrated in Figure 1.1(b). The patch-based AMR strategy relies on a fairly sophisticated algorithm to organize collections of individual computational cells into rectangular patches. The mesh within these newly formed patches can



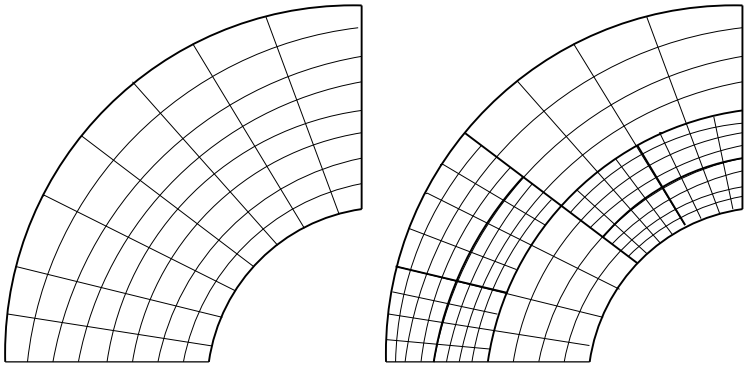
(a) Base Cartesian grid (b) Patch-based AMR (c) Cell-based AMR (d) Block-based AMR

Figure 1.1: Illustration of (b) patch-based, (c) cell-based, and (d) block-based AMR techniques applied to a base Cartesian mesh (a) with cells flagged for refinement indicated by black dots.

then be further refined, creating additional nested patches.

In cell-based AMR, as proposed and developed for example by Powell and co-workers [24–26,31,33,34], Berger and Leveque [22], and Aftosmis and co-workers [35,43,44], each cell may be refined individually as shown in Figure 1.1(c) and is stored using a tree data structure (quadtree in two dimensions, and octree for three dimensions). 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 by the refinement process (4 new cells in two dimensions and 8 in three dimensions). Most cell-based approaches have been applied to Cartesian meshes and, in many cases, cut cells are used to treat complex geometry. Very efficient AMR schemes have been devised using the latter; fully three-dimensional meshes around extremely complex objects can be generated automatically and routinely in a matter of hours or less using this technique [35,43,44]. Nevertheless, discretization of elliptic operators on Cartesian cut cells can be challenging [31] and applications are generally restricted to numerical solution of hyperbolic systems.

In a block-based AMR strategy, mesh adaptation is accomplished by the dividing and coarsening of entire pre-defined grid blocks or groupings of cells. Although not required, each of the groupings or blocks generally has an equal number of cells as shown in Figure 1.1(d). Tree data structures are again used for tracking block connectivity and mesh refinement; however, the block-based AMR strategy results in a much lighter tree structure as compared to that of cell-based methods. While typically larger numbers of mesh



(a) Body-fitted coarse grid (b) Refined body-fitted grid

Figure 1.2: Illustration of block-based adaptive mesh refinement on body-fitted grid topology showing original coarse grid (a) and refined grid (b).

cells are created during the refinement process (i.e., typically more than the corresponding number of cells introduced in cell-based AMR approaches), block-based methods may more readily lend themselves to efficient and scalable parallel implementations via domain decomposition [14, 28, 41, 42, 45–47].

Applications of efficient and scalable parallel block-based approaches on Cartesian meshes are described by Quirk [27], Berger [28], and by Groth *et al.* [41, 42, 48]. More recently, Groth and co-researchers [14, 45–47, 49] have developed a rather flexible block-based AMR scheme allowing automatic solution-directed mesh adaptation on multi-block body-fitted (curvilinear) meshes consisting of quadrilateral (two-dimensional, 2D, case) and hexahedral computational cells (three-dimensional, 3D, case). This block-based approach has been shown to enable efficient and scalable parallel implementations for a variety of flow problems, as well as allow local refinement of body-fitted meshes with anisotropic stretching. The latter aids in the treatment of complex flow geometry and flows with thin boundary, shear, and mixing layers and/or discontinuities and shocks. Extensions of the block-based body-fitted AMR approach for embedded boundaries not aligned with the mesh [50] and with an anisotropic refinement strategy [51] are also possible and have been developed. Figure 1.2 illustrates the application of the block-based AMR technique to a body-fitted mesh.

Finally, hybrid block-based AMR approaches have also been considered. For example, Holst and Keppens [52] applied a hybrid approach to general curvilinear coordinate sys-

tems, modifying the full tree data structure to allow for incomplete block refinement and incorporate ideas from patch-based strategies. The proposed 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 [52] 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.

Solution directed adaptive mesh refinement methods have been applied to the solution of combusting flows. Some examples of this would include the parallel AMR algorithms for 2D low-Mach-number laminar flows developed by Day and Bell [53–55] which was subsequently extended for use in 3D turbulent combustion regimes [53, 56–59]. Bennett and Smooke [9, 60, 61] have employed an adaptive Local Rectangular Refinement (LRR) method to solve 2D co-flow non-premixed and premixed flames. In addition, in the work of Northrup and Groth [62,63] and Gao *et al.* [14,46], a parallel block-based AMR scheme for steady non-premixed 2D laminar and turbulent combusting flows was proposed. More recently, this block-based approach for body-fitted multi-block mesh was extended to the three-dimensional case by Gao and Groth [47, 64] for steady non-premixed flames.

1.2.2 Mesh Refinement Adaption Criteria

One difficulty in achieving definite improvements using adaptive mesh refinement is the lack of reliable error indicators with which to drive the adaptive algorithm. A common strategy has been to adapt the mesh to key physical features of the flow, such as shock waves and boundary layers, by employing indicators based on heuristic measures such the flow gradient, divergence, and/or curl of solution quantities. Such physics-based approaches have proven to be quite effective [14, 24, 27, 45–47, 49, 65]; however, they typically require an a priori knowledge of the flow solution for real success. In some cases, these adaption criteria can also be misleading as resolving the dominant features of a flow as identified by heuristic measures does not necessarily guarantee a corresponding reduction in the overall discretization error.

An alternative approach would be to instead determine a more general estimate of the solution error and then adapt the mesh accordingly so as to reduce this error. A promising procedure originally proposed by Pierce and Giles [66–68] and further refined by Venditti and Darmofal [69–71] and Rannacher *et al.* [72,73] uses an adjoint-based error correction technique. Nemec *et al.* [74–76] have also shown adjoint-based error-estimates for adaptive mesh refinement of embedded-boundary Cartesian meshes to be quite effective for aerodynamic flows.

In spite of the successes to date with error-based mesh adaption, the flows being studied herein are relatively well understood and the flow features requiring adaption, namely the flame fronts and mixing layers, can be easily tracked by using heuristic measures based on the gradients of temperature and species mass fractions. As such, simple physics-based refinement strategies are employed. Investigations into alternative adaption indicators based on estimates of solution error are therefore not considered as part of this research and have been left for future studies.

1.2.3 Parallel Implicit Methods Based on a Newton-Krylov Approach

As previously mentioned, the solution of complex combustion problems result in large systems of nonlinear partial differential equations (PDEs) that can be very computationally expensive to solve. While AMR can help reduce the size of the problem and thus the number of equations that need to be solved, the disparate temporal scales make the systems of equations "stiff", which causes many numerical time-marching schemes to typically have stability and/or convergence issues [77]. For many schemes this necessitates that a very small time step is required which results in the requirement for a very large number of iterations or steps and thus a high computational cost. Further exacerbating the cost for unsteady flows such as those associated with combustion instabilities and thermoacoustic oscillations, is the need to maintain time accuracy. This means many acceleration approaches that have been devised and work efficiently for time-invariant, steady-state, problems can not be directly used.

To deal with numerical stiffness two common algorithms have been proposed and are generally considered: multigrid and implicit methods based on Newton's method. Implicit

schemes have the potential of being unconditionally stable if they are designed in a proper way. The stability allows a larger time step dictated only by accuracy considerations, as the method is not limited by the Courant-Friedrichs-Lewy (CFL) stability criterion that determines the time-step size of traditional explicit schemes. Implicit methods however do require the solution of a linear system of equations, which increases the memory requirements and computation cost per iteration. Efficient parallel implementations of implicit schemes are also more difficult to achieve than for explicit schemes. Nevertheless for a wide range of stiff problems, the higher cost per iteration and programming complexity of implicit methods is often compensated by the algorithm stability and ability to take a much larger time step, leading to an overall reduction in computational time [77, 78].

The application of Newton's method with an appropriate temporal discretization scheme (the application to unsteady problems is discussed below), results in a large sparse set of linear systems of equations that needs to be solved at each iteration. In previous work by Venkatakrishnan [79], a direct method was used to solve the linear system, however the algorithm required high computational cost and memory usage for matrix inversions and as such application to larger problems is prohibitive. Another approach is to use an iterative solver which are less robust however, require less storage than direct methods.

Iterative methods are typically classified into one of two categories, stationary and non-stationary methods. Jacobi, Gauss-Seidel, and successive over-relaxation are examples of stationary iterative methods and all require strong diagonal-dominance properties to converge. The requirement for diagonal dominance can often reduce a stationary method's applicability as a small time step is often required for convergence. Krylov subspace methods are non-stationary iterative methods where the computation involves finding an approximate solution of the linear system from a Krylov subspace. Orthogonality conditions are then imposed to extract the approximate solution. These methods have been shown to perform better than other stationery iterative methods in a number of studies by Nielsen *et al.* [80] and Anderson *et al.* [81]. Various Krylov methods have been developed and are well overviewed in the text by Saad [82] and a review paper by Saad and van der Vorst [83]. Two of the most popular for CFD applications are Bi-Conjugate Gradient Stabilized (Bi-CGSTAB) as first proposed by van der Vorst [84] and the generalized minimal residual (GMRES) algorithm proposed by Saad and co-workers [85–88]. Newer approaches such as a flexible variant of generalized conjugate residual with inner

orthogonalization and outer truncation (GCROT) proposed by Hicken [89] shows promise for certain classes of problems which require high levels of convergence in applications such as gradient based optimization [90].

GMRES is a popular Krylov method for non-symmetric indefinite systems such as those obtained from the Euler and the Navier-Stokes equations. As such GMRES has been very widely applied to the prediction of aerodynamic flows [80, 91–97], and other multi-physics problems such reacting flows [98, 99] and magnetohydrodynamics [100]. In a comprehensive overview paper of Newton-Krylov (NK) methods and application areas, Knoll and Keyes [101] conclude that GMRES and its variants provide the best balance of robustness and performance as long as they are coupled with a suitable preconditioner. The choice of the preconditioner is often more important than the choice of the iterative method [102]. Preconditioning transforms the linear system to a better conditioned one so that it can be solved in fewer iterations. Simple, low cost preconditioners such as a block-diagonal preconditioner have been used successfully by Venkatakrishnan and Mavriplis [92] and Anderson *et al.* [81]. A more effective preconditioner, which Pueyo and Zingg [103] found leads to faster convergence, is incomplete lower-upper factorization (ILU). The factorization can be based on a complete or an approximate Jacobian. ILU preconditioning has higher memory usage and computational cost than the block-diagonal approach but this can be controlled by using a variable fill level ILU. At the limit of zero fill, ILU(0), the preconditioner has the same non-zero pattern as the original Jacobian or approximate Jacobian matrix, making it relatively low storage [102].

Newton-Krylov methods can also be used for the solution of unsteady problems by applying an implicit discretization of the physical time derivative in a dual time-stepping like procedure [104]. This approach has shown great promise for unsteady aerodynamic solutions [78, 105]. This allows the use of non-time accurate convergence acceleration approaches to be used while still maintaining time-accuracy and not being restricted by the CFL stability limitations on time-step size of explicit methods.

To utilize Newton-Krylov methods in large-scale simulations on modern high performance computing (HPC) systems, the algorithm must be adaptable to parallel computing paradigms. The key to any scalable parallel algorithm is finding sufficient independent computations that when computed in parallel greatly surpass the overhead of the required communications. The typical strength of NK methods is the tightly coupled nature of the

algorithm; however, this provides significant challenges when implementing the scheme in parallel. Fortunately, various approaches have been developed to allow NK methods to be applied in a parallel fashion [101, 106–111].

One approach to the parallel implementation of an iterative linear solver and its associated non-linear problem is to solve a Schur complement problem arising from the decomposition of the resulting matrices representing the linear problem [106]. Hicken and Zingg [112] found good success using this approach for the solution of three-dimensional aerodynamic optimization problems. One possible issue with this approach, however is that as the global problem increases for a fixed number of partitions or the number of partitions increases for a fixed size problem so does the Schur complement. This can lead to scaling issues as the number of processors becomes extremely large as is anticipated in the next generation of peta- and exa-scale high performance computers [113].

Another approach is offered by domain-based additive Schwarz preconditioning techniques as pioneered for practical applications by Keyes an co-researchers [107, 108, 110, 114]. Impressive results have been shown in applying parallel implementations of Newton-Krylov-Schwarz (NKS) algorithms for the transonic full potential equations, low-Mach-number compressible combusting flows, three-dimensional inviscid flow, and magneto-hydrodynamics [100, 110, 111, 114, 115]. An additional benefit of the Schwarz preconditioning technique is that it can utilize the same domain decomposition procedure used by the block-based AMR scheme described earlier, making implementation rather straightforward. One possible detraction of the Schwarz approach is that as the level of preconditioning (partitioning) is increased, the overall level of implicitness of the method decreases (less direct coupling of solution content) which can result in longer solution times.

In the specific application area of combustion modelling, Newton-Krylov methods have been successfully applied to the solution of 2D laminar flames by Dobbins and Smooke [116] as well as Veldhuizen *et al.* [117]. NK methods were utilized by Shadid *et al.* [118,119] to solve 3D steady reactive flows and by Pereira *et al.* [99] for solutions of reactions in porous media. Parallel approaches based on Newton-Krylov-Schwarz have also been used for 2D low-Mach-number combustion solutions by Knoll *et al.* [114,115] and Charest *et al.* [120]. Charest *et al.* [121] also applied NK methods to the solution of radiation transport. These early successful applications of Newton-Krylov, especially parallel Newton-Krylov meth-

ods to combustion flows, show that they hold great promise for use in a wider range of more complex and detailed combustion problems.

The approach of combining the strengths of block-based adaptive mesh refinement with an implicit Newton-Krylov-Schwarz time-stepping scheme to arrive at a scalable parallel framework has been intially studied for 2D steady inviscid flows by Groth and Northrup [49] and Charest *et al.* [120, 121] and demonstrated great promise. As such, this thesis deals with further extension of these ideas and the development of a parallel Newton-Krylov-Schwarz AMR algorithm for the solution of steady and unsteady three-dimensional laminar reactive flows.

While as indicated previously, virtually all practical combusting flows are turbulent, for the method development considered here, the focus will be restricted to laminar reactive flows with gaseous fuels. Liquid fuels and multi-phase transport are not considered. In addition the influences of radiation transport and soot formation are also neglected, as are the detailed chemical kinetics required for accurate pollutant prediction. These assumptions will allow some simplification of the physical modelling and thereby facilitate the systematic development of the numerical solution method. Note that an understanding and solution of laminar flames is very often a key ingredient and the basis for modelling turbulent flames [12, 122, 123]. The extension of the proposed solution method to more practical turbulent flames will be the subject of future follow-on studies.

1.3 Thesis Objective

The primary objective of this thesis is the development of an algorithm that will significantly reduce the time it takes to achieve accurate numerical solutions of physically complex steady and unsteady combusting flows. The solution of the Navier-Stokes equations governing laminar flow of a thermally perfect compressible reactive mixture will be considered. This new algorithm will extend the body-fitted, block based, AMR scheme proposed by Gao and Groth [47] to deal with disparate spatial scales and complex geometry. A Newton-Krylov-Schwarz parallel implicit time-marching scheme will be used to deal with the range of temporal scales and numerical stiffness resulting from the chemical mechanisms. Low-Mach-number preconditioning will be used to allow the solution of flows over the full range of Mach numbers. As the Newton-Krylov scheme and low-

Mach-number preconditioning violate time accuracy, a dual-time-stepping like approach will be implemented to calculate unsteady flows. These components will be fully integrated in a ground up parallel implementation that will result in a highly scalable parallel implicit AMR scheme that will be applicable to the solution of a wide range of steady and unsteady two and three-dimensional combustion flows.

As the primary goal of this thesis is the development of a new algorithm for reactive unsteady flow problems, and not the investigation or improvement of the modelling of these flows, relatively simple chemical mechanisms and laboratory-scale laminar flames are investigated, and are used for the purposes of validating the approach. These problems include both premixed and non-premixed laminar flames that are well characterized and studied, yet still provide a physically complex flow that can be computationally demanding to solve. Diffusion flames represent generally more inherently stable reactive flows, that when forced provide a periodic unsteady flow well suited to study the effects of dynamic AMR. Conical premixed laminar flames are inherently more unstable and unsteady due to buoyancy interactions and can be quite difficult to model, but given the correct set of initial and boundary data, are quasi-periodic providing further useful validation cases.

1.4 Thesis Organization

The thesis is structured as follows. Following this introduction, provided here in Chapter 1, the system of governing equations for a compressible thermally perfect reactive mixture of gases is presented in Chapter 2. In Chapter 3, the main elements of the proposed finite-volume scheme and parallel block-based adaptive mesh refinement scheme are outlined. Chapter 4 focuses on the Newton-Krylov-Schwarz algorithm for steady and time-accurate calculations. Validation of the parallel implicit AMR finite-volume scheme for 2D axisymmetric and 3D steady and unsteady flows is then given in Chapter 5. The performance of the proposed algorithm is discussed in Chapter 6. Conclusions and the main research contributions of the thesis are highlighted in Chapter 7 along with possible areas for further study.

Chapter 2

Governing Equations

The numerical solution of steady and unsteady laminar non-premixed and premixed flames for gaseous fuels is considered in this thesis in order to demonstrate the capabilities and potential of the proposed parallel implicit AMR scheme. The mathematical equations and models used to describe the laminar flames of interest are summarized in this chapter. The gaseous reactants and products are treated here as a multi-component mixture of thermally-perfect gases whose transport is governed by the compressible form of the Navier-Stokes equations. Note that the use of the compressible form of the Navier-Stokes equations readily allows for the often large density variations associated with combusting flows as well as the direct prediction of thermoacoustic and other unsteady combustion phenomena often neglected in low-Mach-number descriptions of combustion processes. Appropriate temperature-dependent expressions for species and mixture transport coefficients are used and simple one- and two-step chemical kinetic mechanisms are considered when representing the combustion processes and oxidation of the fuel.

For the purposes of this numerical research, the problems of interest have been restricted to the study of gaseous methane-air flames, and well-established one- and two-step reaction mechanisms are used for this well-characterized hydrocarbon fuel. While such models can obviously introduce errors in the prediction of the maximum flame temperature and cannot provide predictions of pollutant formation, the simplified mechanisms significantly reduce the cost of obtaining solutions and the overall flame structure and dynamics can be quite accurately captured by such methods, as will be demonstrated in Chapter 5. The prediction of pollutant formation is deemed to be outside the scope of

the present work. In addition, the influence of radiation transport and soot formation on flame characteristics, as well as the multi-phase treatment for liquid fuels, are also not considered. The remainder of this chapter provides a summary of the mathematical modelling procedure adopted herein for laminar flames.

2.1 Navier-Stokes Equations for Reactive Mixture of Thermally Perfect Gases

The Navier-Stokes equations for a reactive mixture of thermally perfect gases are used to model flames and combusting flows of interest here. Various aspects of this mathematical description are now reviewed.

2.1.1 Conservation Equations

The conservation form of the Navier-Stokes equations for a multi-species reactive mixture can be written using vector notation as

$$\frac{\partial}{\partial t}(\rho) + \vec{\nabla} \cdot (\rho \vec{u}) = 0 \ , \tag{2.1}$$

$$\frac{\partial}{\partial t}(\rho \vec{u}) + \vec{\nabla} \cdot (\rho \vec{u}\vec{u} + p\vec{\vec{I}}) = \vec{\nabla} \cdot \vec{\vec{\tau}} + \rho \vec{g} \ , \tag{2.2}$$

$$\frac{\partial}{\partial t}\left[\rho E\right] + \vec{\nabla} \cdot \left[\rho \vec{u}(E + \frac{p}{\rho})\right] = \vec{\nabla} \cdot (\vec{u} \cdot \vec{\vec{\tau}}) - \vec{\nabla} \cdot \vec{q} + \rho \vec{g} \cdot \vec{u} \ , \tag{2.3}$$

$$\frac{\partial}{\partial t}(\rho c_s) + \vec{\nabla} \cdot (\rho c_s \vec{u}) = \vec{\nabla} \cdot (\rho \mathcal{D}_s \vec{\nabla} c_s) + \rho \dot{\omega}_s \quad s = 1, \ldots, N \ , \tag{2.4}$$

where

$$\vec{q} = -\kappa \vec{\nabla} T - \rho \sum_{s=1}^{N} h_s \mathcal{D}_s \vec{\nabla} c_s \ , \tag{2.5}$$

and where ρ is the mixture density, p is the total mixture pressure, $\vec{u}$ is the mixture velocity vector, $p\vec{\vec{I}}$ is the identity tensor, E is the total mixture energy, c_s is the mass fraction of species s, $\dot{\omega}_s$ is the time rate of change of the concentration of species s, T is the temperature, $\mathcal{D}_s$ is the diffusion coefficient of species s, κ is the thermal conductivity, $\vec{\vec{\tau}}$ is the fluid stress tensor, $\vec{g}$ is the acceleration vector due to gravity, and N is the number of species in the mixture. Equation (2.1) represents the global or overall mass

conservation equation for the mixture, Equation (2.2) is the mixture momentum equation, Equation (2.3) is the energy equation for the mixture, and Equation (2.4) corresponds to the individual continuity for species s. Note that the total heat flux vector, $\vec{q}$, as given by Equation (2.5), is composed of two components; the thermal energy flux transported by thermal conduction and the energy flux due to diffusion of all species through the mixture.

2.2 Thermodynamic Relations and Transport Relations

The thermodynamic and transport relationships used to complete the mathematical description of a reactive mixture based on the Navier-Stokes equations given in the previous subsection are now summarized.

2.2.1 Perfect Gas Equation of State

The reactive mixture is assumed to obey perfect mixture rules and the perfect or ideal gas equation of state. The latter has the form

$$p = \sum_{s=1}^{N} \rho c_s R_s T = \rho R T \ , \tag{2.6}$$

where

$$R = \sum_{s=1}^{N} c_s R_s \ , \tag{2.7}$$

R_s is the individual species gas constant, and R is the mixture gas constant as defined by Equation (2.7) above.

2.2.2 Internal Energy for a Thermally Perfect Gas

For this research, a single temperature model is used to describe the thermal state of the reactive mixture. For each species, the absolute internal energy, e_s, representing the

sensible and chemical energy and the absolute enthalpy, h_s, have the form

$$e_s = \int_0^T C_{\mathrm{v}_s}(T')dT' \ , \tag{2.8}$$

$$h_s = \int_0^T C_{\mathrm{p}_s}(T')dT' \ . \tag{2.9}$$

where C_{p_s} and C_{v_s} are the specific heat capacities at constant pressure and volume which are taken to be functions of temperature only. Assuming perfect mixing rules for the mixture, the total absolute internal energy for the mixture, e, and the mixture total absolute enthalpy, h, can be expressed as

$$e = \sum_{s=1}^{N} c_s e_s \ , \tag{2.10}$$

$$h = \sum_{s=1}^{N} c_s h_s \ . \tag{2.11}$$

The internal energy and enthalpy are related by $e = h - p/\rho$ and the total mixture energy, E, representing the sum of the internal and bulk kinetic energy of mixtures is then given by

$$E = e + \frac{1}{2}(u^2 + v^2) \ . \tag{2.12}$$

2.2.3 Thermodynamic and Transport Data for Species

The empirical curve fits, correlations and data compiled by McBride and Gordon [124,125] are used herein to prescribe the thermodynamic and transport properties of the individual species. The data set contains properties for over 2,000 substances including 50 reference elements. Curve fits in the form of polynomial functions are provided for specific heat, C_{p_s}, enthalpy, h_s, and entropy, s_s. The data also contains various constant compound properties such as molecular mass $\mathcal{M}_s$ and heat of formation, $\Delta h^o_{\mathrm{f}_s}$. The polynomial expressions for enthalpy, h_s, heat capacity, C_{p_s}, and entropy, s_s, for each species have the

forms

$$\begin{aligned} h_s &= R_s T \left[-a_{1,s} T^{-2} + a_{2,s} T^{-1} \ln T + a_{3,s} + \frac{a_{4,s} T}{2} \right. \\ &\quad \left. + \frac{a_{5,s} T^2}{3} + \frac{a_{6,s} T^3}{4} + \frac{a_{7,s} T^4}{5} + b_1 T^{-1} \right] + \Delta h^o_{\mathrm{f}_s}, \end{aligned} \tag{2.13}$$

$$C_{\mathrm{p}_s} = R_s \left[a_{1,s} T^{-2} + a_{2,s} T^{-1} + a_{3,s} + a_{4,s} T + a_{5,s} T^2 + a_{6,s} T^3 + a_{7,s} T^4 \right] \ , \tag{2.14}$$

$$\begin{aligned} s_s &= -\frac{a_{1,s} T^{-2}}{2} - a_{2,s} T^{-1} + a_{3,s} \ln T + a_{4,s} T \\ &\quad + \frac{a_{5,s} T^2}{2} + \frac{a_{6,s} T^3}{3} + \frac{a_{7,s} T^4}{4} + b_2 \ , \end{aligned} \tag{2.15}$$

where a_i and b_i are the coefficients. As has been mentioned, the enthalpy defined here is the absolute value and includes the heat of formation. This is done to eliminate the need for an extra source term in the energy equation of Equation (2.3) which would be required if values for the sensible energy were used.

The species transport properties of thermal conductivity, κ_s, and viscosity, μ_s, are also obtained from the McBride and Gordon [124, 125] data set and are taken to have the following forms

$$\mu_s = \exp\left(A_{\mu,s} \ln T + B_{\mu,s} T^{-1} + C_{\mu,s} T^{-2} + D_{\mu,s} \right) \ , \tag{2.16}$$

and

$$\kappa_s = \exp\left(A_{\kappa,s} \ln T + B_{\kappa,s} T^{-1} + C_{\kappa,s} T^{-2} + D_{\kappa,s} \right) \ , \tag{2.17}$$

where $A_{\mu,s}$, $B_{\mu,s}$, $C_{\mu,s}$, $D_{\mu,s}$,$A_{\kappa,s}$, $B_{\kappa,s}$, $C_{\kappa,s}$, and $D_{\kappa,s}$ are the provided coefficients for the empirical curve fits.

2.2.4 Mixture Rules

Perfect mixture rules are used to determine mixture thermodynamic properties. In the case of the mixture transport coefficients, Wilke's mixture rule [126] is adopted to evaluate the mixture viscosity. This mixing rule has the form

$$\mu = \sum_{s=1}^{N} \frac{\mu_s c_s}{\mathcal{M}_s \phi_s} \ , \tag{2.18}$$

where

$$\phi_s = \sum_{s'=1}^{N} \frac{c_{s'}}{\mathcal{M}_{s'}} \left[1 + \left(\frac{\mu_s}{\mu_{s'}} \right)^{\frac{1}{2}} \left(\frac{\mathcal{M}_{s'}}{\mathcal{M}_s} \right)^{\frac{1}{4}} \right]^2 \left[8 \left(1 + \frac{\mathcal{M}_s}{\mathcal{M}_{s'}} \right) \right]^{-\frac{1}{2}} \ . \tag{2.19}$$

The thermal conductivity for the mixture is found using the expression due to Mason and Saxena [127] given by

$$\kappa = \sum_{s=1}^{N} \frac{\kappa_s c_s}{c_s + \mathcal{M}_s 1.065(\phi_s - 1)} \ . \tag{2.20}$$

Note that the dimensionless parameters Prandtl number, Pr,

$$\mathrm{Pr} \ = \ \frac{\mu C_\mathrm{p}}{\kappa} \ , \tag{2.21}$$

Lewis number, Le_s,

$$\mathrm{Le}_s \ = \ \frac{\kappa}{\rho \mathcal{D}_s C_\mathrm{p}} \ , \tag{2.22}$$

and Schmidt number, Sc_s,

$$\mathrm{Sc}_s \ = \ \frac{\mu}{\rho \mathcal{D}_s} \ , \tag{2.23}$$

are commonly used to relate the species coefficient of diffusion, $\mathcal{D}_s$, viscosity, μ, and thermal conductivity, κ. As a first approximation, it can be assumed that $\mathrm{Pr} \sim \mathrm{Le} \sim \mathrm{Sc} \sim 1$ for many gases [128]. For this research the coefficients for the diffusion of each species through the mixture, $\mathcal{D}_s$, are determined using the Schmidt number, which are specified for each species in the mixture.

2.3 Chemical Kinetics

2.3.1 Finite Rate Chemistry

The time rate of change of the species concentrations, w_s, is found using the general form of the law of mass action [129] which has the form

$$\dot{\omega}_s = \frac{\mathrm{d}\left[\frac{c_s \rho}{\mathcal{M}_s}\right]}{\mathrm{d}t} = \frac{\mathcal{M}_s}{\rho} \sum_{r=1}^{N_R} \left(\nu_{r,i}^{''} - \nu_{r,i}^{'}\right) \left\{ k_r^f \prod_{i=1}^{N} \left[\frac{c_s \rho}{\mathcal{M}_s}\right]^{\nu_i^{'}} - k_r^b \prod_{i=1}^{N} \left[\frac{c_s \rho}{\mathcal{M}_s}\right]^{\nu_i^{''}} \right\} \ , \tag{2.24}$$

where $\nu_r^{''}$ and $\nu_r^{'}$ are the stoichiometric coefficients of the product and reactant species i of reaction r, k_r^f and k_r^b are the forward and backward reaction rates of reaction r, and N_R is the total number of reactions. The forward reaction rates, k_r^f, for the N_R reactions

used are temperature dependent and are given for each reaction mechanism. The reverse reaction rates, k_r^b, if not given, are defined in terms of the equilibrium constant K_r^{eq} where

$$\frac{k_r^f}{k_r^b} = K_r^{eq} \left(\frac{1}{\mathcal{R}T} \right)^{\sum_s \nu_s} , \tag{2.25}$$

and where

$$K_r^{eq} = e^{\frac{-\Delta G_r^{P=1}}{\mathcal{R}T}} . \tag{2.26}$$

Here $\Delta G_r^{P=1}$ is the change in Gibbs free energy at atmospheric conditions (products minus reactants) for reaction r. The Gibbs free energy for each species is found from $G_s = h_s - Ts_s$ where the species enthalpy, h_s, and entropy, s_s, are specified by again using the empirical curve fits and data sets compiled by McBride and Gordon [124, 125].

2.3.2 Reduced Chemical Mechanisms

As mentioned, the present study is restricted to the combustion of gaseous fuels and in particular methane is used as the representative fuel. Although full or detailed chemical reaction mechanisms are available for describing methane-air (CH_4-air) fuel-oxidizer combustion processes (e.g., refer to the GRI-Mech 3.0 [130] chemical kinetic models), for the purposes of the current numerical study, the finite-rate reaction processes are modeled by employing simplified one and two-step reduced chemical reaction mechanisms. These mechanisms are used to reduce computational time and complexity while still retaining reasonably accurate values for key physical features of methane oxidation such as the premixed laminar flame speed. The primary goal of this research is to develop, verify, and validate a new parallel implicit AMR framework for treating complex three dimensional combusting flows and the simplified chemical mechanisms allow for validation of the methodology without the considerable computational overhead required when dealing with more complex mechanisms. Note that these mechanisms have been used successfully in the solution of 2D axisymmetric methane-air laminar diffusion flames in previous studies by Northrup and Groth [62, 63].

The one- and two-step mechanisms considered here can be summarized as follows:

(a) **Methane - Air – 1 equation, 5 species** (Westbrook and Dryer [131]):

$$CH_4 + 2O_2 \rightarrow CO_2 + 2H_2O$$

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

(b) **Methane - Air – 2 equation, 6 species** (Westbrook and Dryer [131]):

$$\begin{aligned} CH_4 + \frac{3}{2}O_2 &\rightarrow CO + 2H_2O \\ CO + \frac{1}{2}O_2 &\rightleftharpoons CO_2 \end{aligned}$$

The six species considered are methane (CH_4), oxygen (O_2), carbon dioxide (CO_2), carbon monoxide (CO), water (H_2O), and nitrogen (N_2). Nitrogen is again assumed to be inert.

Both mechanisms use Arrhenius-like formulations for the reaction rates. The one-step mechanism only has a forward or overall reaction rate given by

$$k_{ov} = A \exp\left(\frac{-Ea}{\mathcal{R}T}\right) [CH_4]^a [O_2]^b \ . \tag{2.27}$$

The two-step uses the same overall reaction rate for the methane and oxygen, but has forward and reverse reaction rates for the carbon monoxide and oxygen reactions given by

$$k_f = A \exp\left(\frac{-Ea}{\mathcal{R}T}\right) [CO]^a [H_2O]^b [O_2]^c \ , \tag{2.28}$$

$$k_b = A \exp\left(\frac{-Ea}{\mathcal{R}T}\right) [CO_2]^a \ . \tag{2.29}$$

The coefficients for each reaction mechanism are summarized in Table 2.1 and are based on the work of Westbrook and Dryer [131]. They have been "re-tuned" here to provide the correct laminar flame speed as described in Northrup [62]. The units for the concentrations are in cm^3/mol, and the overall units for the reaction rate are in mol/cm^3s.

Unless otherwise stated, the mechanism of choice used for this research is the one-step mechanism with the second-set of rate coefficients for the overall reaction rate given in Table 2.1. The primary reason is that this mechanism has greater than unity coefficients a and b for the reaction rate 2.27, making it easier to deal with computationally in the parallel implicit formulation as will be discussed in Section 4.2.5 of Chapter 4.

Mechanism	**Rate**	**A**	**Ea (J/(mol K))**	**a**	**b**	**c**
One-Step	k_{ov}	2.21×10^{13}	2.0264×10^5	0.2	1.3	
	k_{ov}	1.08×10^{17}	2.0264×10^5	1.0	1.0	
Two-Step	k_{ov}	3.75×10^{13}	2.0464×10^5	0.2	1.3	
	k_f	$10^{14.6}$	1.6747×10^5	1.0	0.5	0.25
	k_b	5.0×10^8	1.6747×10^5	1.0		

Table 2.1: Methane-Air Reduced Chemical Mechanism Reaction Rate Coefficients

Chapter 3

Parallel Adaptive Mesh Refinement Finite-Volume Scheme

The governing partial differential equations outlined in the previous chapter provide a complete description of the physical and chemical processes that occur in the laminar reactive flow of the compressible reactive mixture. This chapter presents the details of the proposed parallel adaptive mesh refinement (AMR) finite-volume method developed for the numerical solution of these equations. Section 3.1 summarizes the finite-volume spatial discretization scheme and the details related to the evaluation of the numerical fluxes, both inviscid (hyperbolic) and viscous (elliptic). Section 3.2 describes the implementation of the finite-volume scheme within a parallel block-based AMR framework. The solution of the resulting semi-discrete form of the conservation equations via a parallel implicit Newton-Krylov method is fully described in the next chapter of the thesis, Chapter 4.

3.1 Finite-Volume Method

In a finite-volume approach, the governing partial differential equations are solved on a domain discretized into series of contiguous control volumes, or cells making up the computational grid for the geometry of interest. The equations are solved in integral form enforcing both locally and globally conservation of key flow properties, such as mass, momentum, and energy. A brief summary of the proposed finite-volume scheme is

provided here; however, for more detailed explanations of the principles of finite-volume methods the reader is referred to the textbooks by Toro [132], Hirsch [133, 134], and Lomax, Pulliam, and Zingg [77].

Using matrix vector notation, the Navier-Stokes Equations (2.1)-(2.4) can be written in weak conservation form as

$$\frac{\partial \mathbf{U}}{\partial t} + \vec{\nabla} \cdot \vec{\mathbf{F}} = \mathbf{S}\,, \tag{3.1}$$

where $\mathbf{U}$ is the vector of flow solution variables, $\vec{\mathbf{F}}$ is the total solution flux dyad, and $\mathbf{S}$ is the source vector. Following integration over a three-dimensional control volume in physical space, and subsequent application of the divergence theorem to the conservation form of the governing equations, Equation (3.1), one arrives at the integral form of the Navier-Stokes which can be written as

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{V(t)} \mathbf{U}\mathrm{d}V + \oint_{\Omega(t)} \vec{\mathbf{F}} \cdot \vec{n}\,\mathrm{d}\Omega = \int_{V(t)} \mathbf{S}\mathrm{d}V\,, \tag{3.2}$$

where 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. The flux dyad, $\vec{\mathbf{F}}$, represents the flux of solution quantities through the boundaries control volume. These fluxes can be generally categorized as either arising from wave propagation phenomena (hyperbolic fluxes) or from diffusion processes (elliptic fluxes). Expressions for $\mathbf{U}$, $\vec{\mathbf{F}}$, and $\mathbf{S}$ are given below.

For a fixed three-dimensional control volume that does not vary in time as shown in Figure 3.1, the cell-averaged solution and source vectors can be defined as follows:

$$\overline{\mathbf{U}} \equiv \frac{1}{V}\int_{\mathrm{V}} \mathbf{U}\ \mathrm{d}V\,, \tag{3.3}$$

$$\overline{\mathbf{S}} \equiv \frac{1}{V}\int_{\mathrm{V}} \mathbf{S}\ \mathrm{d}V\ , \tag{3.4}$$

where V is the cell volume. If one considers a hexahedral computational cell as depicted in Figure 3.1, and then substitutes these definitions into the integral form of the conservation equations given by Equation (3.2), one arrives at the semi-discrete form of the conservation equations given by

$$\frac{\mathrm{d}\overline{\mathbf{U}}}{\mathrm{d}t} = -\frac{1}{V}\sum_{\mathrm{face}} \left(\vec{\mathbf{F}}_{\mathrm{face}} \cdot \vec{n}_{\mathrm{face}} \Delta A_{\mathrm{face}}\right) + \overline{\mathbf{S}}(\overline{\mathbf{U}})\ , \tag{3.5}$$

or

$$\frac{\mathrm{d}\overline{\mathbf{U}}}{\mathrm{d}t} = -\mathbf{R}(\overline{\mathbf{U}})\ , \tag{3.6}$$

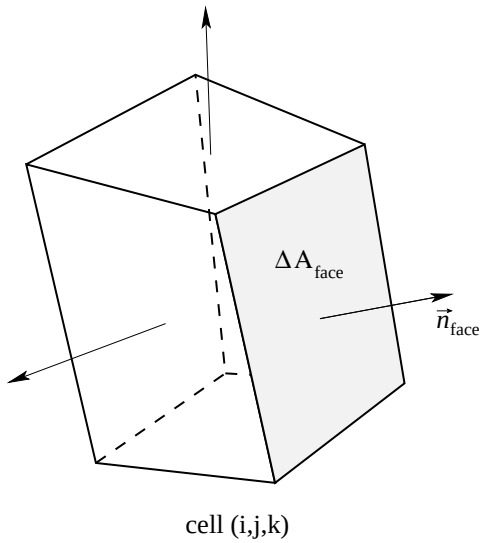


Figure 3.1: Three-dimensional hexahedral cell

where the integration of the solution flux over the cell surface has been replaced by the mid-point quadrature rule (valid for second-order accuracy), where $\vec{n}_{\text{face}}$ and ΔA_{face} are the unit outward normal and area of each cells face respectively, and $\mathbf{R}(\overline{\mathbf{U}})$ is the residual operator for the control volume. Equation (3.5) is a non-linear system of first-order ordinary differential equations describing the time evolution of the average value of the solution vector in each computational cell. In order to solve the resulting system of equations, the various terms appearing in Equation (3.5) must be evaluated. The flux, $\vec{\mathbf{F}}$, at the cell boundary is evaluated as a function of the discontinuous states on either side of the cell interfaces, where discontinuities arise due to the piecewise approximations for $\mathbf{U}$ within each control volume. The numerical evaluation of these fluxes is discussed in Sections 3.1.3 and 3.1.4 to follow.

3.1.1 Conservation Forms of the Governing Equations

As mentioned, the present work considers the numerical solution of laminar reactive flows for both two-dimensional axisymmetric and three-dimensional coordinate frames. As such the vector of flow solution variables, $\mathbf{U}$, the flux dyad, $\vec{\mathbf{F}}$, and the source vector

$\mathbf{S}$ are given by

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

Here $\mathbf{F}$ 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, $\mathbf{G}$ and $\mathbf{G}_{\rm v}$ 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 $\mathbf{H}_{\rm v}$ are the inviscid and viscous flux vectors in the z direction for three-dimensional flows. Finally, $\mathbf{S}_{\rm a}$ and $\mathbf{S}_{\rm av}$ are the source terms associated with the axisymmetric system, and $\mathbf{S}$ is the source term associated with the finite-rate chemical kinetics and gravity. Explicit definitions for $\mathbf{U}$, $\vec{\mathbf{F}}$, and $\mathbf{S}$ are given below for both the two-dimensional (axisymmetric) and three-dimensional Cartesian coordinate frames.

Navier-Stokes for Axisymmetric Coordinate Frame

For a two-dimensional axisymmetric coordinate system, the Navier-Stokes equations for a reactive mixture can be written as

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial(\mathbf{F}-\mathbf{F}_{\rm v})}{\partial r} + \frac{\partial(\mathbf{G}-\mathbf{G}_{\rm v})}{\partial z} = -\frac{(\mathbf{S}_{\rm a}-\mathbf{S}_{\rm av})}{r} + \mathbf{S}\,, \tag{3.7}$$

where the vector of conserved variables, $\mathbf{U}$, is given by

$$\mathbf{U} = \left[\ \rho,\ \rho u,\ \rho v,\ \rho E,\ \rho c_1,\ \ldots,\ \rho c_N\ \right]^T\,, \tag{3.8}$$

and $\mathbf{S_A}$ and $\mathbf{S}$ are the axisymmetric and finite rate chemistry source terms (the latter also includes the gravitational forces) and have the forms

$$\mathbf{S_A} = \begin{bmatrix} -\rho u \\ -\rho u^2 + \tau_{rr} - \tau_{\theta\theta} \\ -\rho u v + \tau_{rz} \\ -\rho u\left(E + \dfrac{p}{\rho}\right) - \mathrm{q}_r + u\tau_{rr} + v\tau_{rz} \\ -\rho c_1 u + \rho \mathcal{D}_1 \frac{\partial c_1}{\partial r} \\ \vdots \\ -\rho c_N u + \rho \mathcal{D}_N \frac{\partial c_N}{\partial r} \end{bmatrix}, \quad \mathbf{S} = \begin{bmatrix} 0 \\ 0 \\ \rho \mathrm{g}_z \\ \rho \mathrm{g}_z v \\ \rho\dot{\omega}_1 \\ \vdots \\ \rho\dot{\omega}_N \end{bmatrix}. \tag{3.9}$$

The inviscid solution flux vectors, $\mathbf{F}$ and $\mathbf{G}$, in this case are given by

$$\mathbf{F} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho v u \\ \rho u \left(E + \frac{p}{\rho} \right) \\ \rho c_1 u \\ \vdots \\ \rho c_N u \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ \rho v \left(E + \frac{p}{\rho} \right) \\ \rho c_1 v \\ \vdots \\ \rho c_N v \end{bmatrix}, \tag{3.10}$$

and the viscous solution flux vectors, $\mathbf{F_v}$ and $\mathbf{G_v}$, can be written as

$$\mathbf{F_v} = \begin{bmatrix} 0 \\ \tau_{rr} \\ \tau_{zr} \\ -\mathrm{q}_r + u\tau_{rr} + v\tau_{rz} \\ \rho \mathcal{D}_1 \frac{\partial c_1}{\partial r} \\ \vdots \\ \rho \mathcal{D}_N \frac{\partial c_N}{\partial r} \end{bmatrix}, \quad \mathbf{G_v} = \begin{bmatrix} 0 \\ \tau_{rz} \\ \tau_{zz} \\ -\mathrm{q}_z + u\tau_{rz} + v\tau_{zz} \\ \rho \mathcal{D}_1 \frac{\partial c_1}{\partial z} \\ \vdots \\ \rho \mathcal{D}_N \frac{\partial c_N}{\partial z} \end{bmatrix}. \tag{3.11}$$

In the preceding axisymmetric equations, r is the radial, z is the axial, and θ is the azimuthal coordinate in the axisymmetric frame, u and v are the radial and axial components of the velocity, g_z is the component of gravity in the axial direction, q_r and q_z are the radial and axial components of the heat flux, and τ_{rr}, τ_{rz}, τ_{zr}, τ_{zz}, and $\tau_{\theta\theta}$ are the elements of the fluid stress tensor.

Navier-Stokes for a 3D Coordinate Frame

For a fully 3D Cartesian coordinate frame, the Navier-Stokes equations for a reactive mixture can be written as

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \left(\mathbf{F} - \mathbf{F}_\mathrm{v} \right)}{\partial x} + \frac{\partial \left(\mathbf{G} - \mathbf{G}_\mathrm{v} \right)}{\partial y} + \frac{\partial \left(\mathbf{H} - \mathbf{H}_\mathrm{v} \right)}{\partial z} = \mathbf{S} \, , \tag{3.12}$$

where the vector of conserved variables, $\mathbf{U}$ is given by

$$\mathbf{U} = \left[\; \rho, \; \rho u, \; \rho v, \; \rho w, \rho E, \; \rho c_1, \; \ldots, \; \rho c_N \; \right]^T \, , \tag{3.13}$$

and the source vector, $\mathbf{S}$, includes the finite rate chemistry and gravitational forces source terms and is given by

$$\mathbf{S} = \left[\; 0,\; 0,\; 0,\; \rho g_z,\; \rho g_z w,\; \rho\dot{\omega}_1,\; \ldots,\; \rho\dot{\omega}_N \; \right]^T . \tag{3.14}$$

The inviscid solution flux vectors, $\mathbf{F}$, $\mathbf{G}$, $\mathbf{H}$, for the 3D case are

$$\mathbf{F} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho v u \\ \rho w u \\ \rho u \left(E + \frac{p}{\rho} \right) \\ \rho c_1 u \\ \vdots \\ \rho c_N u \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ \rho w v \\ \rho v \left(E + \frac{p}{\rho} \right) \\ \rho c_1 v \\ \vdots \\ \rho c_N v \end{bmatrix}, \quad \mathbf{H} = \begin{bmatrix} \rho w \\ \rho u w \\ \rho v w \\ \rho w^2 + p \\ \rho w \left(E + \frac{p}{\rho} \right) \\ \rho c_1 w \\ \vdots \\ \rho c_N w \end{bmatrix}, \tag{3.15}$$

and the viscous solution flux vectors, $\mathbf{F_v}$, $\mathbf{G_v}$, $\mathbf{H_v}$, are given by

$$\mathbf{F_v} = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ -q_x + u\tau_{xx} \\ +v\tau_{xy} + w\tau_{xz} \\ \rho\mathcal{D}_1 \frac{\partial c_1}{\partial x} \\ \vdots \\ \rho\mathcal{D}_N \frac{\partial c_N}{\partial x} \end{bmatrix}, \quad \mathbf{G_v} = \begin{bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ \tau_{zy} \\ -q_y + u\tau_{xy} \\ +v\tau_{yy} + w\tau_{yz} \\ \rho\mathcal{D}_1 \frac{\partial c_1}{\partial y} \\ \vdots \\ \rho\mathcal{D}_N \frac{\partial c_N}{\partial y} \end{bmatrix}, \quad \mathbf{H_v} = \begin{bmatrix} 0 \\ \tau_{xz} \\ \tau_{yz} \\ \tau_{zz} \\ -q_z + u\tau_{xz} \\ +v\tau_{yz} + w\tau zz \\ \rho\mathcal{D}_1 \frac{\partial c_1}{\partial z} \\ \vdots \\ \rho\mathcal{D}_N \frac{\partial c_N}{\partial z} \end{bmatrix} \tag{3.16}$$

In the preceding equations, x, y, and z are the three components of the Cartesian coordinate frame, u, v, and w are the components of the velocity, g_z is the component of gravity in the z direction, q_x, q_y, and q_z are the components of the heat flux, and τ_{xx}, τ_{xy}, τ_{xz}, τ_{yy}, τ_{yz}, τ_{zz} are the elements of the fluid stress tensor.

3.1.2 Semi-Discrete Form of the Governing Equations

The semi-discrete form of the two-dimensional form of the conservation Equation (3.5), is given then by

$$\frac{\mathrm{d}\overline{\mathbf{U}}_{ij}}{\mathrm{d}t} = -\frac{1}{A_{ij}} \sum_{m=1} \left(\vec{\mathbf{F}}_{ijm}, \cdot \mathbf{n}_{ijm} \Delta \ell_{ijm} \right) + \mathbf{S}_{ij} = \mathbf{R}_{ij}(\mathbf{U}) \ , \tag{3.17}$$

where A_{ij} is the cell volume, and $\mathbf{n}_{ijm}$ and $\Delta \ell_k$ are the unit outward normal vector and the length of cell-edge m, respectively. For the three-dimensional case, the semi-discrete form can be written as

$$\frac{\mathrm{d}\overline{\mathbf{U}}_{ijk}}{\mathrm{d}t} = -\frac{1}{V_{ijk}} \sum_{m=1} \left(\vec{\mathbf{F}}_{ijkm}, \cdot \mathbf{n}_{ijkm} \Delta A_{ijkm} \right) + \mathbf{S}_{ijk} = \mathbf{R}_{ijk}(\mathbf{U}) \ , \tag{3.18}$$

where V_{ijk} is the cell area, and $\mathbf{n}_{ijkm}$ and ΔA_{ijkm} are the unit outward normal vector and the area of cell-face m, respectively.

3.1.3 Inviscid (Hyperbolic) Flux Evaluation

The inviscid components of the flux, $\vec{\mathbf{F}}$, appearing in Equations (3.17) and (3.18) are evaluated by applying a Godunov-type upwind finite-volume spatial discretization procedure in conjunction with limited linear solution reconstruction. As proposed by Godunov [18] in 1959, Godunov's method is a monotonicity preserving scheme which is able to capture solution discontinuities, such as shocks, without introducing oscillations in the solutions. Godunov-type finite-volume methods use the solution of one-dimensional Riemann problems to evaluate the numerical flux at the cell boundaries.

Riemann Problem

A Riemann problem is a special form of a one-dimensional initial value problem having discontinuous initial states. This is the situation at the interface between two computational cells described previously, if $\mathbf{U}_i$ is considered the left initial state, $\mathbf{U}_L$, and $\mathbf{U}_{i+1}$ considered the right, $\mathbf{U}_R$, as depicted on Figure 3.2 for a simple one-dimensional computational domain in the direction normal to the cell face. In this case, the inviscid equations describing the solution transport through the face can be reduced to the

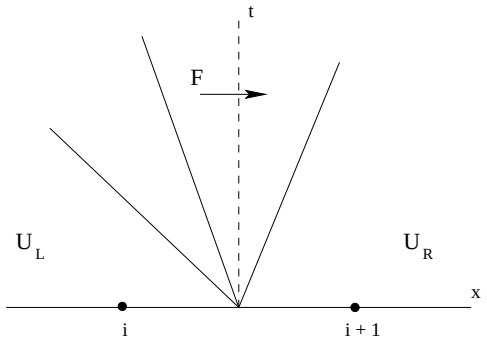


Figure 3.2: The Riemann Problem

one-dimensional form

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} = \frac{\partial \mathbf{U}}{\partial t} + \mathbf{A}(\mathbf{U})\frac{\partial \mathbf{U}}{\partial x} = 0 \ , \tag{3.19}$$

where here x represents the coordinate direction normal to the face and $\mathbf{A}(\mathbf{U}) = \partial \mathbf{F}/\partial \mathbf{U}$ is the Jacobian of the inviscid flux vector with respect to the solution vector.

For a polytropic gas, the Riemann problem can be solved exactly as described by Gottlieb and Groth [135] or approximately using alternative approaches developed by Roe [136], Harten-Lax-van Leer-Einfeldt(HLLE) [137], HLL [138], Linde [139], and AUSM$^+$up [140]. For gaseous mixtures of interest in this research, the Roe approximate linearized Riemann solver has been extended to deal with a reactive mixture of thermally perfect gases and is used almost exclusively in all the calculations. The HLLE and AUSM$^+$up approximate solvers have also been implemented for such reactive mixtures but are not discussed here.

Roe Approximate Riemann Solver

The Roe approximate Riemann solver does not set out to solve the Riemann problem exactly, but instead it seeks a good estimate of the numerical flux at the interface based on a local linearization of the governing equations. It does this by locally (i.e, at each cell interface) replacing the Jacobian matrix $\mathbf{A}(\mathbf{U}) = \partial \mathbf{F}/\partial \mathbf{U}$ in Equation (3.19) above with a constant matrix $\hat{\mathbf{A}} = \hat{\mathbf{A}}(\mathbf{U}_L, \mathbf{U}_R)$ which is a function of the local left and right states. For a reactive mixture, an average state that exactly satisfies Roe's property U can be defined, however in this work an approximate Roe average is used [141, 142]. The approximation used for this work assumes that the Jacobian matrix $\hat{\mathbf{A}}$ is evaluated using the Roe-averaged flow variables, f_*, defined in terms of a mass weighting of the left and

right flow variables, f_L and f_R, as given by

$$f_* = \frac{\rho_R f_R + \rho_L f_L}{\rho_R + \rho_L} \ , \tag{3.20}$$

where f_L and f_R can be any of the variables u, v, h, e, and c_s. The density, ρ, is averaged using $\rho_* = \sqrt{\rho_R \rho_L}$. Once the matrix $\hat{\mathbf{A}}$, its eigenvalues $\hat{\mathbf{\Lambda}}$, and right eigenvectors $\hat{\mathbf{R}}$ are calculated the flux can be found using

$$\mathbf{F}\left(\mathbf{U}_L, \mathbf{U}_R\right) = \frac{1}{2}\left(\mathbf{F}_R + \mathbf{F}_L\right) - \frac{1}{2}|\hat{\mathbf{A}}|\left(\mathbf{U}_R - \mathbf{U}_L\right) \ , \tag{3.21}$$

where

$$|\hat{\mathbf{A}}| = \hat{\mathbf{R}}|\hat{\mathbf{\Lambda}}|\hat{\mathbf{R}}^{-1} \ . \tag{3.22}$$

Piecewise Limited Linear Reconstruction

An important concern is the accuracy of the discretization methods. Simply using the left and the right cell-centered solutions to calculate the cell boundary fluxes yields a first-order accurate solution. For higher-order accuracy (i.e., second-order accuracy in smooth regions), a higher-order representation or spatial reconstruction of the solution is required in each computational cell. The emergence of high-resolution Godunov-type methods motivated the design of effective limiters for use in one-dimensional higher-order reconstructions [143]. 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 [144] and van Leer [145]. These concepts which prevent the occurrence of numerical oscillations, were later generalized via the concept of total variation diminishing (TVD) by Harten [146]. The reader is referred to the paper by van Leer [143] for a systematic review and comparisons of various techniques related to this topic.

In the present work, a higher-order Godunov-type finite-volume upwind formulation based on approximate Riemann solvers with a least-squares piecewise limited linear solution reconstruction procedure is used to evaluate the components of the hyperbolic solution flux and achieve second-order accuracy. Here, the values of the left and right solution states at a cell interface are determined by least-squares piecewise 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

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

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

$$\begin{aligned} \mathbf{W}_{\mathrm{L}} &= \mathbf{W}_{i,j,k} + \Phi_{i,j,k}\vec{\nabla}\mathbf{W}_{i,j,k} \cdot \Delta\vec{x}_{\mathrm{L}}\,, \\ \mathbf{W}_{\mathrm{R}} &= \mathbf{W}_{i+1,j,k} + \Phi_{i+1,j,k}\vec{\nabla}\mathbf{W}_{i+1,j,k} \cdot \Delta\vec{x}_{\mathrm{R}}\,, \end{aligned} \tag{3.23}$$

using the slope limiter, Φ. $\mathbf{W}_{i,j,k}$ and $\mathbf{W}_{i+1,j,k}$ are cell-averaged primitive solution vectors in the neighbouring cells and $\Delta\vec{x}_{\mathrm{L}}$ and $\Delta\vec{x}_{\mathrm{R}}$ are the distances between the centroid of the cell and the cell interface for the left and right cells, respectively.

The slope limiter, Φ, is introduced to limit the solution gradient in order to ensure solution monotonicity. In the present work, the limiter proposed by Venkatakrishnan [147] has been used and is given by

$$\Phi_{i,j,k} = \begin{cases} \phi\left(\frac{\mathbf{W}_{\max} - \mathbf{W}_{i,j,k}}{\mathbf{W}_k - \mathbf{W}_{i,j,k}}\right) & \text{for} \quad \mathbf{W}_k - \mathbf{W}_{i,j,k} > 0 \\ \phi\left(\frac{\mathbf{W}_{\min} - \mathbf{W}_{i,j,k}}{\mathbf{W}_k - \mathbf{W}_{i,j,k}}\right) & \text{for} \quad \mathbf{W}_k - \mathbf{W}_{i,j,k} < 0 \\ 1 & \text{otherwise} \end{cases}, \tag{3.24}$$

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

$$\phi(y) = \frac{y^2 + 2y}{y^2 + y + 2}\,. \tag{3.25}$$

and $\mathbf{W}_{\max} = \max(\mathbf{W}_{i,j,k}, \mathbf{W}_{\text{neighbours}})$, $\mathbf{W}_{\min} = \min(\mathbf{W}_{i,j,k}, \mathbf{W}_{\text{neighbours}})$, and $\mathbf{W}_k$ is the unlimited reconstructed solution value at the k^{th} flux quadrature point.

Least Squares Gradient Evaluation

The gradients of the primitive variables, $\vec{\nabla}\mathbf{W}$, are determined by applying a least-squares approach [148]. This technique 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 are used to generalize the procedure without reducing the reconstruction stencil. 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

$$\sum_{k=1}^{k=N} \epsilon_{ik}^2 = \sum_{k=1}^{k=N} \left(\Delta \mathbf{W}_{ik} - \vec{\nabla}\mathbf{W}_i \cdot \Delta\vec{x}_{ik}\right)^2 , \tag{3.26}$$

where $\Delta \mathbf{W}_{ik} = \mathbf{W}_i - \mathbf{W}_k$, $\Delta\vec{x}_{ik} = \vec{x}_i - \vec{x}_k$, and $N = 8$ for two dimensions, or $N = 26$ for three dimensions. In the 3D case, the 3×3 system of linear algebraic equations resulting from the minimization problem can be expressed as

$$\begin{bmatrix} \overline{(\Delta x)^2} & \overline{\Delta x \Delta y} & \overline{\Delta x \Delta z} \\ \overline{\Delta x \Delta y} & \overline{(\Delta y)^2} & \overline{\Delta y \Delta z} \\ \overline{\Delta x \Delta z} & \overline{\Delta y \Delta z} & \overline{(\Delta z)^2} \end{bmatrix} \begin{bmatrix} \frac{\partial \mathbf{W}}{\partial x} \\ \frac{\partial \mathbf{W}}{\partial y} \\ \frac{\partial \mathbf{W}}{\partial z} \end{bmatrix} = \begin{bmatrix} \overline{\mathbf{W}\Delta x} \\ \overline{\mathbf{W}\Delta y} \\ \overline{\mathbf{W}\Delta z} \end{bmatrix} , \tag{3.27}$$

where

$$\overline{\Delta x^2} = \frac{1}{N} \sum_{k=1}^{N} \Delta x_{ki}^2, \tag{3.28}$$

$$\overline{\Delta x \Delta y} = \frac{1}{N} \sum_{k=1}^{N} \Delta x_{ki} \Delta y_{ki}, \tag{3.29}$$

and

$$\overline{\Delta \mathbf{W} \Delta x} = \frac{1}{N} \sum_{k=1}^{N} \Delta \mathbf{W}_{ki} \Delta x_{ki} . \tag{3.30}$$

The other terms have a similar form. The above terms only depend on grid geometry and so can be pre-computed and stored. Solutions of the 3×3 linear system represented by Equation (3.27) can be readily obtained using Cramer's rule.

Low-Mach-Number Preconditioning

The finite-volume scheme described above is a fully compressible formulation that can readily accommodate large density variations and thermoacoustic phenomena. Nevertheless, laminar combusting flows are in general characterized by very low Mach numbers

($M < 0.2$) and nearly incompressible behaviour. The direct application of unmodified compressible flow solvers to nearly incompressible flows can lead to several numerical difficulties related to disparities between the convective and acoustic propagation speeds (i.e., $|\mathbf{u}| + a \gg |\mathbf{u}|$, where a is the sound speed). The numerical difficulties include slow convergence rates and excessive numerical dissipation. To circumvent these difficulties, a local preconditioning technique proposed by Weiss and Smith [149, 150] is used here to both remove numerical stiffness and maintain solution accuracy for low-Mach-number flows.

For the 3D case, the preconditioned form of the Navier-Stokes equations for the reactive mixture can be written as

$$\mathbf{\Gamma}\frac{\partial \mathbf{U}}{\partial \tau} + \frac{\partial \mathbf{U}}{\partial t} + \frac{\partial}{\partial x}\mathbf{F} + \frac{\partial}{\partial y}\mathbf{G} + \frac{\partial}{\partial z}\mathbf{H} = \frac{\partial}{\partial x}\mathbf{F_v} + \frac{\partial}{\partial y}\mathbf{G_v} + \frac{\partial}{\partial z}\mathbf{H_v} + \mathbf{S} \ , \tag{3.31}$$

where $\mathbf{\Gamma}$ is the Weiss-Smith preconditioning matrix for the conserved variables and τ represents an artificial or pseudo time that has been introduced along with the preconditioner. A properly chosen preconditioning matrix can be used to reduce the spread of the eigenvalues and improves the numerical solution in the low-Mach-number limit. By applying the preconditioner using a pseudo time, τ, the time accuracy can still be preserved for unsteady flows if a dual-time stepping procedure is utilized [104]. Additionally, steady state solutions, for which $\partial \mathbf{U}/\partial t = 0$, are unaffected by the preconditioning procedure.

In the case of partial differential equations, the condition number is the ratio of largest to smallest eigenvalues for the system of equations. The larger the condition number the greater the disparity in the time scales associated with the acoustic and convective solution modes. As the Mach number approaches zero, the condition number of the non-preconditioned system approaches infinity as given in Figure 3.3. The preconditioned system resulting from the application of the Weiss-Smith preconditioning, however remains well conditioned in the low-Mach-number limit, and for ideal one-dimensional inviscid flow, asymptotes to a value of 2.62 [151] as shown in Figure 3.3.

In terms of the symmetrizing variables, the Weiss-Smith preconditioning matrix is a diagonal matrix with a rather simple structure. However, for the thermally perfect reactive system, the Weiss-Smith preconditioner, Γ, based on the vector of primitive

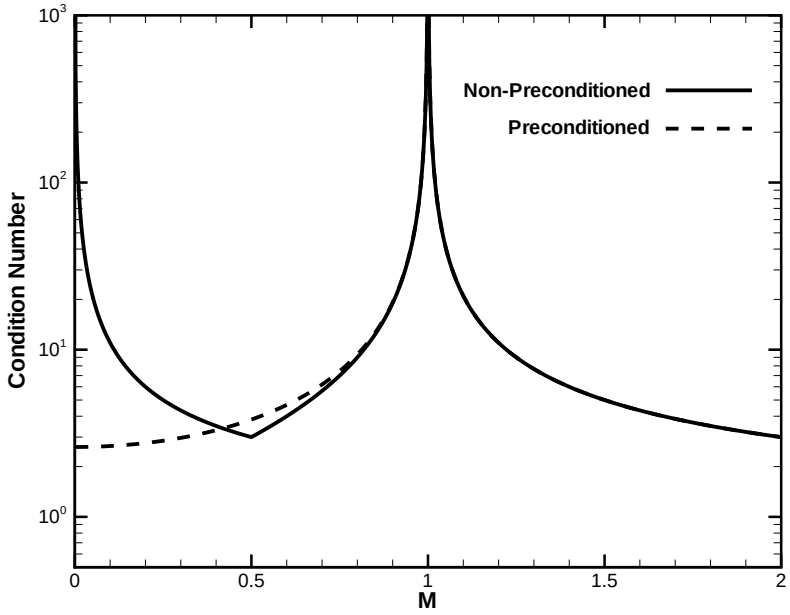


Figure 3.3: Condition number for non-preconditioned and preconditioned inviscid form of the conservation equations for thermally perfect reactive mixtures with fixed composition.

variables, $\mathbf{Q}$,

$$\mathbf{Q} = \left[\; p, \; u, \; v, \; w, \; T, \; c_1, \; \ldots, \; c_N \; \right]^T , \tag{3.32}$$

is given by

$$\mathbf{\Gamma}_{ws} = \begin{bmatrix} \Theta & 0 & 0 & 0 & -\frac{\rho}{T} & 0 & \cdots & 0 \\ 0 & \rho & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \rho & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \rho & 0 & 0 & \cdots & 0 \\ -1 & 0 & 0 & 0 & \rho C_p & \eta_1 & \cdots & \eta_N \\ 0 & 0 & 0 & 0 & 0 & \rho & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & \rho \end{bmatrix} . \tag{3.33}$$

The variables Θ and η_N in the equation above have the forms

$$\begin{aligned} \Theta &= \left(\frac{1}{M_r^2 a^2} + \frac{1}{C_p T} \right) , \\ \eta_s &= h_s - \frac{C_p T R_s}{R} . \end{aligned}$$

A preconditioner for the conservative form of the equations can be achieved by the transformation

$$\mathbf{\Gamma} = \boldsymbol{K}^{-1}\mathbf{\Gamma}_{ws}\frac{\partial \mathbf{Q}}{\partial \mathbf{U}} \ , \tag{3.34}$$

where $\boldsymbol{K}$ is a transformation matrix and $\partial\mathbf{Q}/\partial\mathbf{U}$ is the Jacobian of the primitive solution vector with respect to the conserved solution vector.

The eigenvalues for the preconditioned Jacobian matrix, $\mathbf{\Gamma}^{-1}\partial\mathbf{F}/\partial\mathbf{U}$, are

$$\lambda_1 = u' - a' \ \lambda_2 = u, \ \lambda_3 = u, \ \lambda_4 = u, \ \lambda_5 = u' + a', \ \lambda_6 \cdots \lambda_N = u \ . \tag{3.35}$$

The preconditioned sound speed, a', and preconditioned flow velocity, u', are given by

$$\begin{aligned} u' &= u\,(1-\alpha) \ , \\ a' &= \sqrt{\alpha^2 u^2 + M_r^2 a^2} \ , \end{aligned}$$

with the variables α and β defined by

$$\begin{aligned} \alpha &= \frac{1}{2}\left(1 - \beta M_r^2 a^2\right) \ , \\ \beta &= \left(\frac{1}{RT} - \frac{1}{C_p T}\right) \ , \end{aligned}$$

and where M_r is the reference Mach number for the preconditioner.

The reference Mach number, M_r, controls the level of preconditioning. To avoid singularities when the local Mach number approaches to zero, such as near a stagnation point [149, 151], M_r is determined from the maximum of the local Mach number and a minimum allowable Mach number $M_{r_{min}}$ (the latter is normally set to the mean free stream Mach number) with a maximum value of unity corresponding to no preconditioning (no preconditioning is applied in the supersonic regime). For inviscid flows, M_r is taken to have the form

$$M_r = \min\left(\max\left(\frac{u}{a}, M_{r_{min}}\right), 1\right) \ . \tag{3.36}$$

For viscous flows, an additional constraint is placed on M_r, such that it does not become smaller than the local diffusion velocity and is thus given by

$$M_r = \max\left(M_r, \frac{\mu}{a\rho\Delta x}\right) \ . \tag{3.37}$$

The Weiss-Smith preconditioner as described above can be applied, as indicated in Equation (3.31), to improve the convergence characteristics of the solution method for low-Mach-number flows. However, its application is equally if not more important for the control of numerical dissipation. In order to reduce the excessive dissipation produced by the upwinding approach in the low-Mach-number limit, the preconditioner is also applied to the flux evaluation procedure in the Roe approximate Riemann solver. The dissipation term $|\hat{\mathbf{A}}|\,(\mathbf{U}_R - \mathbf{U}_L)$ from Equation (3.21) is modified as follows

$$\begin{aligned} |\hat{\mathbf{A}}|\Delta\mathbf{U} &\approx \hat{\mathbf{A}}\,(\mathbf{U}_R - \mathbf{U}_L) \\ &= \boldsymbol{\Gamma}\left(\boldsymbol{\Gamma}^{-1}\frac{\partial \mathbf{F}}{\partial \mathbf{U}}\right)(\mathbf{U}_R - \mathbf{U}_L) \\ &= \boldsymbol{\Gamma}|\hat{\mathbf{A}}_\Gamma|\,(\mathbf{U}_R - \mathbf{U}_L)\ , \end{aligned} \tag{3.38}$$

where $\hat{\mathbf{A}}_\Gamma$ is the preconditioned Jacobian

$$|\hat{\mathbf{A}}_\Gamma| = \hat{\mathbf{R}}_\Gamma\,|\hat{\boldsymbol{\Lambda}}_\Gamma|\,\hat{\mathbf{R}}_\Gamma^{-1} = \hat{\mathbf{R}}_\Gamma\,|\hat{\boldsymbol{\Lambda}}_\Gamma|\,\hat{\mathbf{L}}_\Gamma\ , \tag{3.39}$$

with eigenvalue matrix, $|\hat{\boldsymbol{\Lambda}}_\Gamma|$, defined by

$$|\hat{\boldsymbol{\Lambda}}_\Gamma| = \begin{bmatrix} |u'-a'| & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & |u| & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & |u| & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & |u| & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & |u'+a'| & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & |u| & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & |u| \end{bmatrix}\ . \tag{3.40}$$

The resulting numerical flux function then has the form

$$\mathbf{F}\left(\mathcal{R}\left(\mathbf{U}_L, \mathbf{U}_R\right)\right) = \frac{1}{2}\left(\mathbf{F}_R + \mathbf{F}_L\right) - \frac{1}{2}\boldsymbol{\Gamma}|\hat{\mathbf{A}}_\Gamma|\left(\mathbf{U}_R - \mathbf{U}_L\right)\ . \tag{3.41}$$

For unsteady flows, Venkateswaran and Merkle [152, 153] proposed an additional constraint on M_r to ensure acoustic waves are accurately captured, however as only diffusion dominated flames are investigated in this thesis, this is not of current concern. Furthermore, when coupled with a Newton-Krylov scheme, as discussed in the next chapter, the temporal preconditioning is no longer required, and the preconditioning is primarily used to control the excessive dissipation produced by the upwinding approach in the low-Mach-number limit.

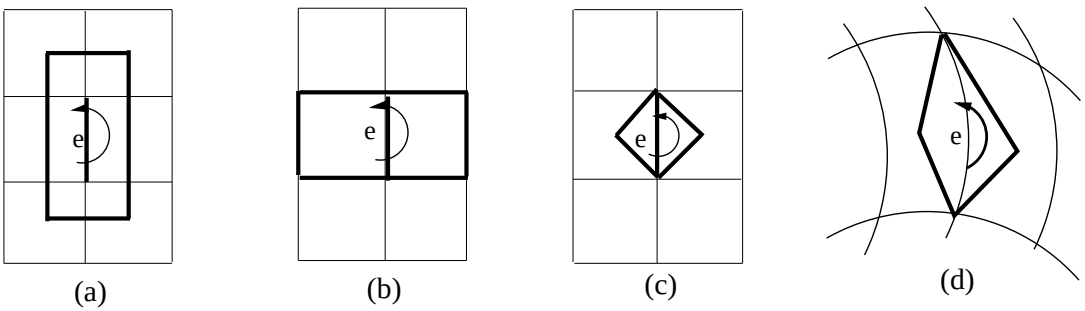


Figure 3.4: Possible reconstruction paths showing: (a) centroidal path; (b) existing faces co-volume; (c) diamond path on a Cartesian grid; and (d) diamond path on a curvilinear grid.

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

$$\vec{\mathbf{F}} \cdot \vec{n} = \vec{\mathbf{F}}(\mathbf{W}_{i+\frac{1}{2},j,k}, \vec{\nabla}\mathbf{W}_{i+\frac{1}{2},j,k}) \ , \tag{3.42}$$

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

$$\mathbf{W}_{i+\frac{1}{2},j,k} = \frac{(\mathbf{W}_{\mathrm{L}} + \mathbf{W}_{\mathrm{R}})}{2} \ . \tag{3.43}$$

The evaluation of the gradients for the primitive variables at the cell interface, $\vec{\nabla}\mathbf{W}_{(i+\frac{1}{2},j,k)}$, 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 3.4 illustrates a choice of three different paths in two dimensions: (a) centroidal path; (b) existing faces co-volume; (c) diamond path on Cartesian grid; and (d) diamond path on a curvilinear grid. Coirier [31] 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 the 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 [34, 154].

In this work, Green-Gauss integration over the diamond path using the linearity-preserving weighting function derived by Holmes and Connell to evaluate the gradients at each cell interface is used in the 2D case and given by

$$\vec{\nabla}\mathbf{W}_{i+\frac{1}{2},j} = \frac{\vec{n}}{\vec{n}\cdot\vec{e_{\mathrm{s}}}}\left(\frac{\mathbf{W}_{i+1,j}-\mathbf{W}_{i,j}}{\Delta \mathrm{s}} + \frac{\mathbf{W}_{i+\frac{1}{2},j+\frac{1}{2}}-\mathbf{W}_{i+\frac{1}{2},j-\frac{1}{2}}}{\Delta l}\vec{e_{\mathrm{t}}}\cdot\vec{e_{\mathrm{s}}}\right). \tag{3.44}$$

In Equation (3.44), $\Delta \mathrm{s}$ is the distance between two centroids, Δl is the face length, and unit vectors, $\vec{e_{\mathrm{t}}}$, $\vec{n}$, and $\vec{e_{\mathrm{s}}}$ are the tangential vector, face norm, and the distance vector from the cell centroid to the neighbour centroid, as shown in Figure 3.5.

In the 3D case, the edges of the diamond path are replaced by surfaces as shown in Figure 3.6. However, extending Equation (3.44) to three dimensions is not straightforward as the face tangential vectors are not uniquely defined for most hexahedral mesh. Therefore, in this research work, the cell-face gradients are evaluated using the formula proposed by Mathur and Murthy [155]

$$\vec{\nabla}\mathbf{W}\Big|_{i+\frac{1}{2},j,k} = \frac{\mathbf{W}_{i+1,j,k}-\mathbf{W}_{i,j,k}}{\Delta \mathrm{s}}\frac{\vec{n}}{\vec{n}\cdot\vec{e_{\mathrm{s}}}} + \left(\overline{\vec{\nabla}\mathbf{W}} - \overline{\vec{\nabla}\mathbf{W}}\cdot\vec{e_{\mathrm{s}}}\,\frac{\vec{n}}{\vec{n}\cdot\vec{e_{\mathrm{s}}}}\right), \tag{3.45}$$

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

$$\overline{\vec{\nabla}\mathbf{W}}\Big|_{i+\frac{1}{2},j,k} = \alpha\vec{\nabla}\mathbf{W}_{i,j,k} + (1-\alpha)\vec{\nabla}\mathbf{W}_{i+1,j,k}\,. \tag{3.46}$$

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

$$\alpha = \frac{V_{i,j,k}}{(V_{i,j,k}+V_{i+1,j,k})}\,. \tag{3.47}$$

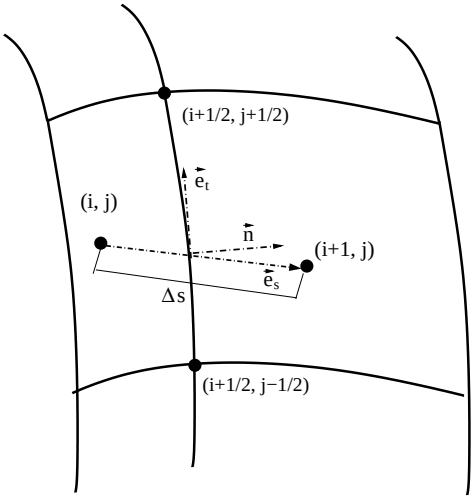


Figure 3.5: 2D Cell face gradient reconstruction.

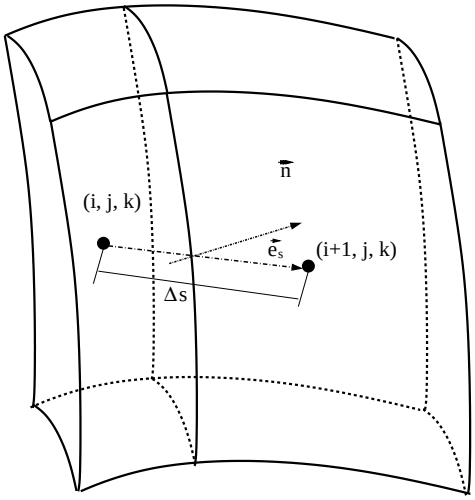


Figure 3.6: 3D Cell face gradient reconstruction.

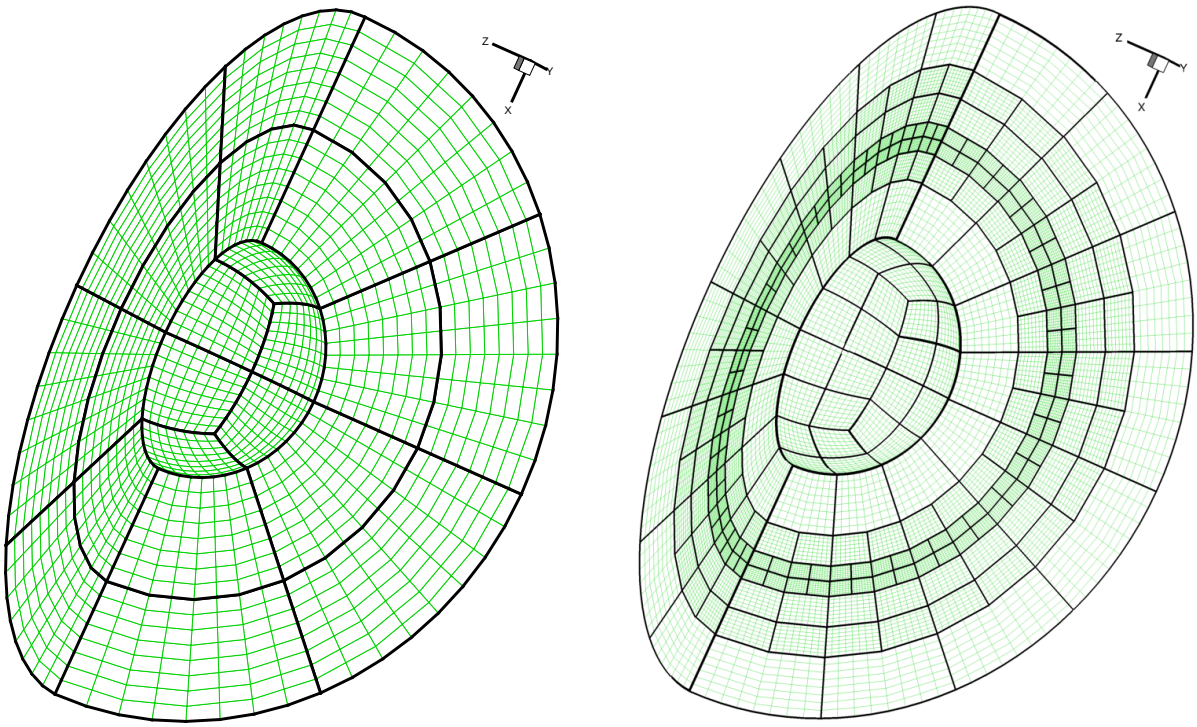
3.2 Block-Based Adaptive Mesh Refinement

Adaptive mesh refinement is an approach which allows local mesh refinement in areas of interest while maintaining a coarse mesh in areas not requiring the finer resolution. This approach minimizes the number of computational cells required thus reducing the size and computational cost of a particular problem. Originally proposed by Berger and Oliger for computing time-dependent solutions to hyperbolic PDEs in multiple space dimensions [19], AMR approaches have since been developed for a wide variety of engineering problems [14, 21, 23–25, 27–30, 32, 35–42, 45–47, 64, 156]. As discussed in Chapter 1, there are many types of AMR approaches such as cell-based, patch-based, block-based, and hybrid block-based that have been developed and each have its benefits and shortcomings. Ultimately the choice comes down to a compromise between refinement efficiency, parallel scalability, and data-structure complexity. Cell-based methods have the highest refinement efficiency but require larger and more complex connectivity data-structures, whereas block-based have the most straight forward data-structures and are easiest to load-balance at the price of slightly more cells and therefore a lower refinement efficiency.

The proposed scheme here adopts a block-based approach to AMR as proposed by Groth and co-workers [49,157] for a flexible block-based AMR scheme allowing automatic solution-directed mesh adaption on multi-block body-fitted (curvilinear) meshes consisting of quadrilateral (two-dimensional, 2D) and hexahedral (three-dimensional, 3D). This

block-based approach has been shown to enable efficient and scalable parallel implementations for a variety of flow problems with anisotropic stretching. The latter aids in the treatment of complex flow geometry and flows with thin boundary, shear, and mixing layers and/or discontinuities and shocks. Applications of the block-based AMR scheme have included laminar flames [14, 63] with soot prediction [120] and radiation transport [121], turbulent non-premixed flames [14, 46, 158] as well as turbulent multiphase rocket core flows [45, 157], magnetohydrodynamics (MHD) simulations [41, 42] and micron-scale flows [159]. Extensions of the block-based body-fitted AMR approach for embedded boundaries not aligned with the mesh [50] and with an anisotropic refinement strategy [51, 160, 161] are also possible and have been developed.

The AMR methodology described above borrows from previous work by Berger and coworkers [19–21, 28, 35], Quirk [23, 27], and De Zeeuw and Powell [25] for Cartesian mesh and has similarities with the block-based approaches described by Quirk and Hanebutte



(a) (b)

Figure 3.7: 3D hexahedral body fitted mesh of a quarter sphere (a) with 12 (8×8×8) initial blocks and (b) 1104 (8×8×8) blocks after four levels of mesh refinement.

[27] and Berger and Saltzman [28]. Other researchers have considered the extension of Cartesian mesh adaptation procedures to more arbitrary quadrilateral and hexagonal mesh. See for example the work by Davis and Dannenhoffer [162] and Sun and Takayama [163]. The flexible block-based hierarchical data structure used here follows the approach developed by Groth *et al.* [41, 42] for computational magnetohydrodynamics.

Block-based AMR corresponds to the situation where the mesh is adapted using a domain decomposed superset of cells, or blocks, that each contain the same number of computational cells. Three-dimensional grid blocks can then be easily adapted by division of the block into 8 sub or child blocks, or 4 children in two-dimensions, that are self-similar to their parents. This has two major benefits over other cell or patch based approaches in that the block locations can readily be tracked using a compact hierarchical data structure, and as each block has the same number of cells, each can be treated as an equivalent work unit making load-balanced parallelization very straightforward. An example of block-based AMR is shown in Figure 3.7 where a 3D body-fitted quarter sphere with an initial 12 block mesh is shown before and after four levels of mesh refinement.

This work builds upon the block-based AMR algorithm described by Gao *et al.* [14,46,47, 64] for three space dimensions. In the previous work by Gao *et al.*, the focus was primarily on refinement only (not coarsening) for steady-state flow problems with somewhat restricted grid block topologies. Assumptions concerning grid block connectivity were made that restricted the generality of the approach: specifically, the approach used for determining block connectivity was not sufficiently general to allow for both refinement and coarsening dynamically in unsteady flow computations. In this work, the approach has been re-implemented with a focus on a more generalized block connectivity and data structure which allows for both refinement and coarsening as required for unsteady flows and a wider range of multi-block mesh topologies. The following sections outline the major components of the AMR algorithm with emphasis on those components and aspects that have been modified for improved functionality and performance, particularly for unsteady flows.

3.2.1 Refinement and Coarsening

As noted above, the governing equations are integrated to obtain volume averaged solution quantities within hexahedral computational cells. These cells are embedded in structured blocks consisting of $N_i \times N_j \times N_k$ cells, where N_i, N_j, and N_k are the number of cells in each logical direction i, j, and k respectively and have values that are even, but not necessarily equal integers. Mesh adaptation is accomplished by the dividing and coarsening of the appropriate solution blocks.

In regions requiring increased cell resolution, a "parent" block is refined by dividing itself into eight "children" or "offspring". Each of the eight octants or sectors 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 process can be reversed in regions that are deemed over-resolved and eight children are coarsened into a single parent block. The mesh refinement 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. An example of the block-based mesh adaption process as described is shown in Figure 3.8 for two and three-dimensional meshes with 3 levels of refinement.

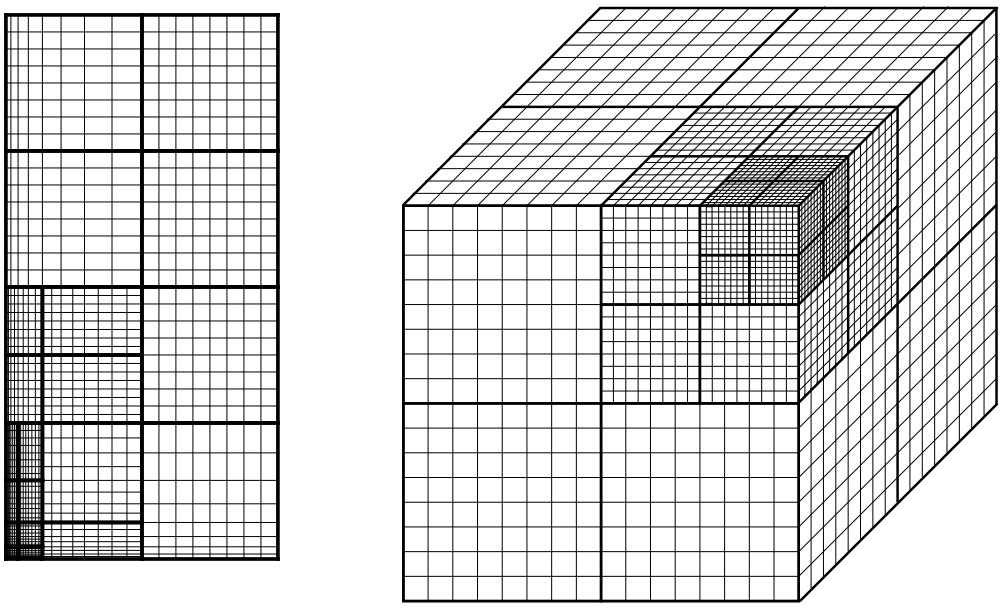
Standard multigrid-type restriction and prolongation operators are used to evaluate the solution on all blocks created by the coarsening and division processes, respectively. Refinement criteria based on a combination of the gradients of the mixture temperature, T, and species mass fractions, c_s, have been shown to provide reliable detection of flame and combustion fronts [14, 63]. The criteria have the form

$$\epsilon_1 \propto |\nabla T| \quad \epsilon_2 \propto |\nabla c_s| \ , \tag{3.48}$$

and where the measures ϵ_1 and ϵ_2 are large blocks are refined, and where small blocks are coarsened based on prescribed relative thresholds.

3.2.2 Solution Block Connectivity

Neighbour information for each computation cell in the mesh is required in order for a solution update to be performed via application of the discretized conservation equations. In standard structured mesh approaches, the cell connectivity is naturally provided by the native i, j, k indexing of the mesh, i.e., by the corresponding computational or logical



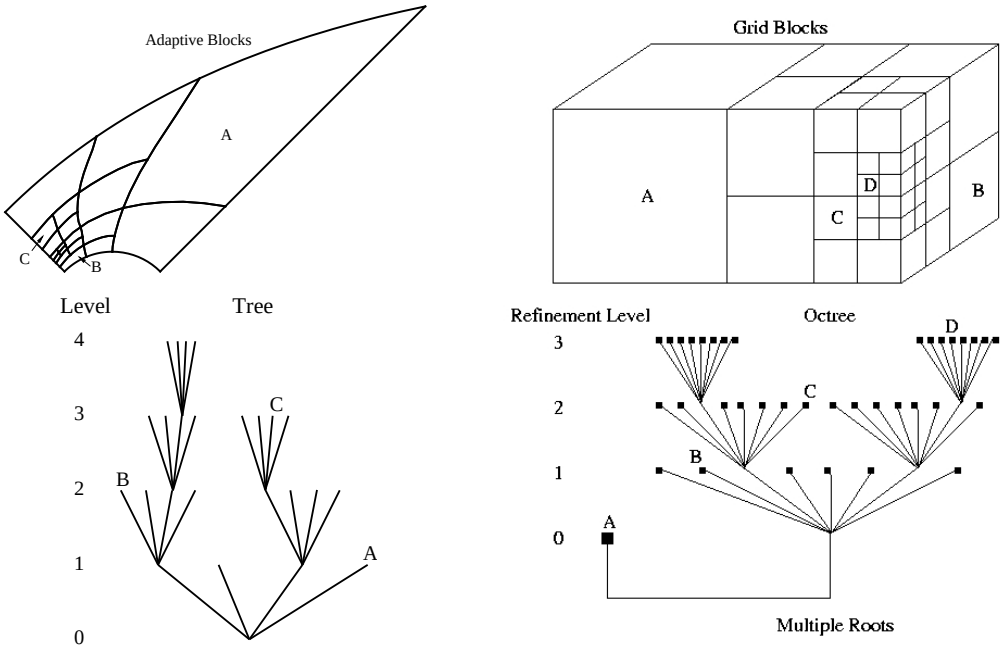
(a) (b)

Figure 3.8: (a) 2D Computational mesh with 20 (8×8) blocks and 1280 cells, showing 3 levels of refinement. (b) 3D Computational mesh with 22 (8×8×8) blocks and 11264 cells, showing 3 levels of refinement.

coordinates. In unstructured meshes, a complete cell by cell connectivity table must instead be determined. In adaptive mesh refinement for the latter, this is further complicated as this connectivity information has to be updated whenever the mesh is updated. In block-based AMR, the native structured indexing is preserved internally to each block (at the cell level), however the block connectivity must be tracked and updated in order to exchange solution and/or geometry information during the solution procedure.

Hierarchical Tree Data Structure

A hierarchical tree-like data structure, a quadtree in two-dimensions and an octree for three-dimensions as shown graphically in Figure 3.9, is used to keep track of mesh refinement and the connectivity between solution blocks. The use of this structured approach in storing block connectivity provides a very efficient data structure that can be quickly traversed and updated. It is also very lightweight, in terms of storage requirements, so can be stored on every processor, reducing communication overhead.



(a) (b)

Figure 3.9: Adaptive mesh refinement data structures and associated solution blocks; (a) quadtree for 2D quadrilateral and (b) octree 3D hexahedral meshes.

The quadtree/octree data structures used here naturally keep 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 the original stretching of the initial mesh and allows for anisotropic mesh spacing based on the stretching and improved treatment of thin boundary and shear layers. Note that strictly anisotropic mesh adaption strategy has been considered by other researchers [31, 51, 164, 165] and a hierarchical binary-tree data structure [31] and/or an indexing scheme for Cartesian mesh can be used to keep track of the grid connectivity [164, 165].

Unstructured Root-Block Connectivity

The connectivity between blocks must to be determined in order to carry out message passing of solution information between blocks. In a structured Cartesian arrangement of blocks, the connectivity calculation is straightforward as the connectivity can be stored logically in two- and three-dimensional arrays and the block orientations are always

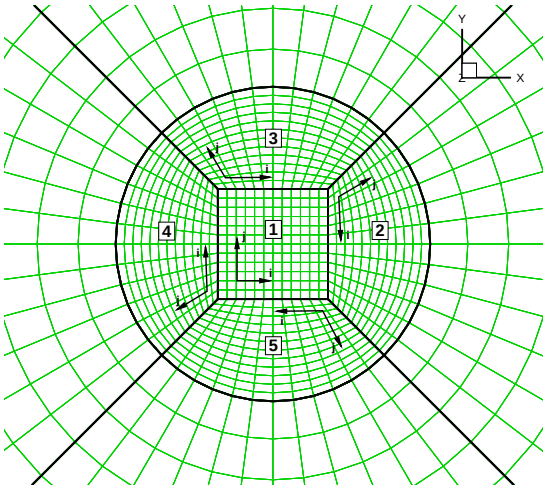


Figure 3.10: Unstructured block connectivity of a cylinder using hexahedral blocks showing local block coordinate frames.

aligned. To accommodate hexahedral based body-fitted grids for more general geometries such as cylinders and cube-spheres, an unstructured root block connectivity is required to track the relative orientation of neighbouring blocks. The logic employed follows the methodology proposed in the Computational Fluid Dynamics General Notation System (CGNS) [166]. A typical refined block stores the relative orientation of the neighbouring blocks in the directions of all 26 boundary elements (6 faces, 12 edges and 8 vertices), and for each neighbour the orientation of the i, j, and k axes relative to the orientation of the i, j and k axes in the current block is stored in compact form as a three-component transformation array.

To illustrate the unstructured connectivity between blocks, consider the center block 1 shown in Figure 3.10 in reference to its south neighbour, block 5. The i and j indices of block 5 run in directions opposite to the i and j indices in block 1. This is indicated by negative signs in the block 1-to-block 5 transformation array, which is given by [-1, -2, +3]. The components 1, 2, and 3 in the array signify the i, j and k indices in block 5. The value 1 in the first component of the array means that the i index from block 5 is associated with the first index of block 1 (its i index), and the negative sign indicates that they run in opposite directions. The value 2 in the second position means that the j indices of the two blocks are also associated and running in opposite directions, and the value +3 in the third direction indicates that the k indices have

the same orientation. These transformation arrays represent a short-hand notation for the transformation matrices [64, 158, 166] describing the relation between indices of two adjacent blocks, which can be used to exchange solution information between blocks having common interfaces in a general and transparent way.

The connectivity information is propagated from the root blocks to refined blocks via the aforementioned quadtree/octree data structure, in such a way that each refined block stores a transformation array describing index axis alignment with all of its (typically 26 in three space dimensions) neighbour blocks. These block-to-block transformation arrays are used in the solution procedure to properly compute numerical fluxes through the block boundaries. Note that the transformation array mechanism is implemented uniformly for all blocks, but it only results in nontrivial action at block-block interfaces along sector boundaries, in a transparent manner.

Unstructured root-block presents some additional challenges as the number of neighbour blocks can vary i.e., not all 8 neighbours in 2D or 26 in 3D are always present. Figure 3.10 shows a 2D case where the center block only has 4 neighbours. This presents an issue at the block corners where grid cells adjacent to one of the eight sector corners have only seven neighbouring cells in the 2D case, while all other cells have 8 neighbours. As these cells are used for gradient reconstruction and flux evaluation, special considerations must be taken at the corners. The approach used is to automatically detect blocks with such corner cells, and assigning collapsed corner ghost cells to those blocks sharing the relevant corner. In practice, this is implemented by using fictitious values in those corner ghost cells. These collapsed ghost cells are not used in the stencils for reconstruction computation, so grid cells adjacent to sector corners employ smaller stencil sizes. The flexible least-squares reconstruction scheme of the proposed finite-volume solution procedure can automatically handle this transparently without a reduction in the order of solution accuracy of the scheme.

An illustration of this matter is shown in Figure 3.11 where a two-dimensional cylinder is again used with the center block having only 4 neighbours. The cells marked with a "**o**" are under gradient reconstruction and the cells marked "**x**" denote the ghost cells that provide information for reconstruction. In the regular interior and corner reconstructions all 8 neighbour cells are available for use, however at the corners of the center block where only 3 blocks abut, only 7 neighbour cells are available, so the reconstruction stencil is

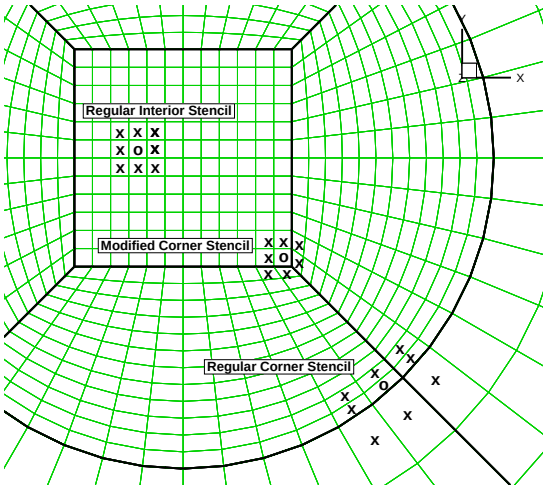


Figure 3.11: Depiction of cells participating in reconstruction stencils in different regions or a cylinder mesh. The cell of which the solution is reconstructed is marked with a "**o**" and the neighbouring cells that are part of the stencil are marked with a "**x**".

reduced accordingly.

Updating the Tree Data Structure

Once the initial, or root, unstructured block layout connectivity is determined from the physical locations of the blocks in the starting mesh and stored in the hierarchical data tree structure, the tree itself contains all the required information to update neighbour information as blocks are refined and coarsened. This is very useful as the reliance on geometry is removed, making the process computationally quite inexpensive and more robust.

In previous work by Gao *et al.* [47, 64], the connectivity was determined by relying on the parents connectivity information for the new child block. Such a procedure is sufficient when only refinement for steady-state problems is performed as a blocks parent information will always be valid; however when coarsening and refinement is performed simultaneously, which is common in dynamic mesh refinement for unsteady problems, the parent neighbour information may no longer be valid and thus cannot be used. In this work, each refined/coarsened block's parent and child connectivity is re-calculated to ensure it is consistent with the new mesh topology. This is done using a recursive

search of the data tree, climbing back towards the root block to update the resulting coarsened/refined block's neighbour information. If the neighbour block is not on the same branch (derived from the same root block) then traversing between root blocks (based on the initial layout) is performed and then the search continues climbing back up the new branch until the neighbour is found. Since the data tree is structured and self-similar, determining neighbour information is relatively straightforward starting from the root block's connectivity and descending from parent to child. Although the revised approach to determining block connectivity is slightly more expensive than the approach of Gao *et al.* [47, 64], the resulting algorithm is far more robust and actually simpler to implement as the whole tree can be recreated from the root blocks without relying on the existing connectivity as was previously required.

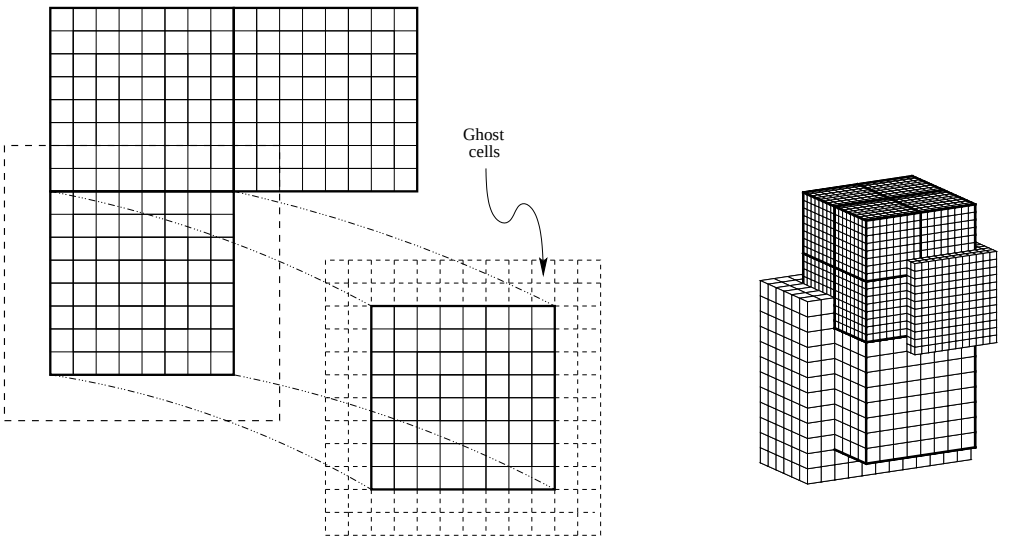
3.2.3 Information Exchange Between Blocks

Solution information is shared between adjacent blocks having common interfaces by employing two additional layers of overlapping "ghost" cells. Figures 3.12(a) and 3.12(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. They also provide a means to reconstruct the solution for second order spatial accuracy as well as applying boundary conditions.

3.2.4 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 [19, 21]. 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:



(a) Two space dimensions (b) Three space dimensions

Figure 3.12: Two layers of overlapping "ghost" cells contain solution information from neighbouring blocks.

- 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 passed to the coarse neighbour solution block;
- evaluate the flux variance, $\Delta\vec{\mathbf{F}} = \vec{\mathbf{F}}_{\text{fine}} - \vec{\mathbf{F}}_{\text{coarse}}$, where $\vec{\mathbf{F}}_{\text{coarse}}$ is the coarse cell flux;
- correct the residual for the coarse cell (i, j, k) using $\vec{\mathbf{R}}_{i,j,k} = \vec{\mathbf{R}}_{i,j,k} - \frac{\text{CFL}\Delta\text{t}_{\text{i,j,k}}\text{A}_{\text{coarse}}\Delta\tilde{\mathbf{F}}}{V_{i,j,k}}$, where $\Delta\text{t}_{i,j,k}$ is the time step and $V_{i,j,k}$ is the cell volume.

A similar variant of this procedure is applied when obtaining solutions on two-dimensional multi-block meshes with AMR.

3.3 Parallel Implementation

Current hardware and software paradigms are still very much geared towards serial computing. To exploit parallelism, whether it be on shared or distributed memory systems, a program must explicitly provide a mechanism to subdivide the computation into independent sub-problems that can be worked on simultaneously, define the distribution

of the sub-problems, and organize the communications between sub-domains. This section describes the proposed algorithm parallel implementation through the use of domain decomposition, Morton ordering task distribution, and using the MPI library for inter-process communication.

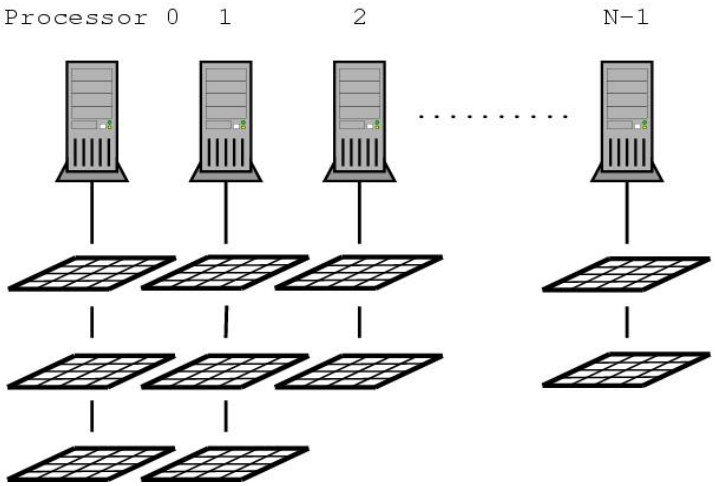


Figure 3.13: Parallel Domain Decomposition

3.3.1 Domain Decomposition

Domain decomposition is a technique of solving 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 available processors. In this thesis, the computational domain of interest is a multi-block mesh, which lends itself naturally to domain decomposition. The domain decomposition is carried out by farming the mesh blocks out to the separate processors, with more than one block permitted on each processor as shown in Figure 3.13.

3.3.2 Morton Ordering

For a parallel algorithm to be successful, i.e., readily scalable, it must avoid load imbalance and reduce communication overhead as much as possible. Many factors can lead

to load imbalance and communication overhead, such as characteristics of the computational architectures and/or the nature of the numerical algorithm. To mitigate these factors various approaches have been devised.

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.

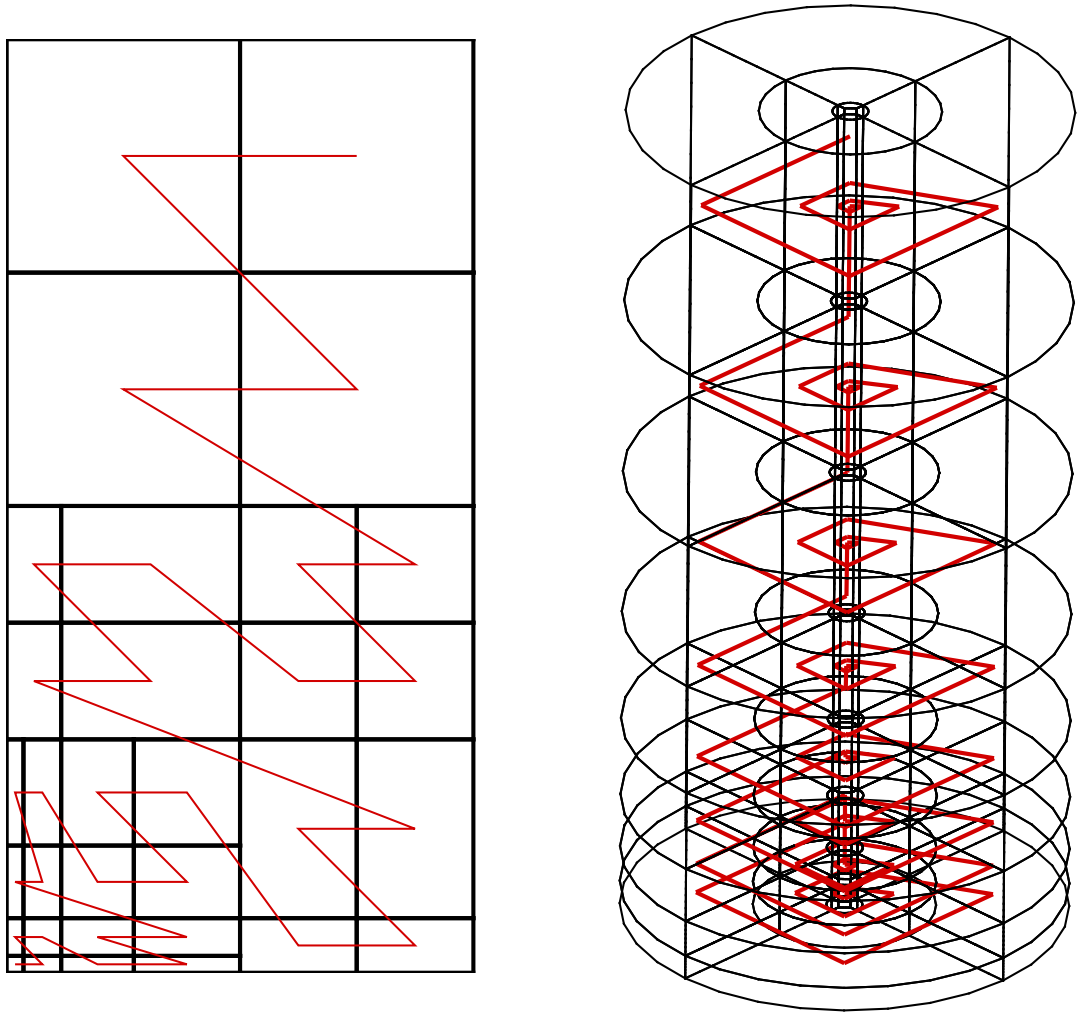
Placing nearest-neighbour blocks on the same processor can also help to reduce the overall communication costs. This is usually realized by utilizing space-filling curves which can provide rather high quality partitions at very low computational costs [44, 168, 169] due to their "proximity preserving" 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 performance of the proposed solution method [44]. Figure 3.14 shows the Morton ordering space filling curve (coloured red line) passing through each of the solution blocks (solid black lines) in both a two-dimensional multi-block quadrilateral and three-dimensional hexahedral mesh.

3.3.3 Message Passing Interface (MPI)

The parallel implementation of the block-based AMR scheme was developed using the C++ programming language [170] and the MPI (message passing interface) library [171]. 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 each residual evaluation. Message passing of the ghost-cell values and flux corrections is performed in an asynchronous fashion with gathered wait states and message consolidation.

3.3.4 Computational Resources

The computational resources for performing all of the calculations reported in this thesis were provided by the SciNet High Performance Computing Consortium [172] at the University of Toronto and Compute/Calcul Canada through funding from the Canada Foundation for Innovation (CFI) and the Province of Ontario, Canada. All of the computations were carried out on two clusters, the General Purpose Cluster (GPC) and the Tightly Coupled System (TCS). The GPC consists of Intel Xeon E5540 (2.53 GHz) nodes whereas the TCS cluster is based on IBM Power6-575 (4.7 GHz) nodes. Both systems are connected with a high-speed, low-latency, non-blocking DDR Infiniband interconnect. Scaling studies were also carried out on an IBM BlueGene/Q (BGQ) supercomputer [173, 174] which consists of 2,048 low-power 16 core CPUs (32,768 cores) connected together with a highly scalable proprietary 5D torus network. The BGQ is also hosted and operated by SciNet with funding provided by the Southern Ontario Smart Computing Innovation Platform (SOSCIP).



(a) 2D Mesh (b) 3D Mesh

Figure 3.14: Morton ordering space filling curve used to provide nearest-neighbour ordering of blocks for efficient load balancing of blocks on multiple processors. The coloured red line represents the space filling curve passing through each of the solution blocks in a multi-block (a) 2D quadrilateral and (b) 3D hexahedral mesh.

Chapter 4

Newton-Krylov-Schwarz Algorithm

The previous chapter outlined the finite-volume approach used to discretize the governing equations in two and three dimensional physical space. The resulting semi-discrete form of the governing equations given in Equation (3.6) form a coupled set of non-linear ordinary differential equations. For completeness they are reproduced here and have the form

$$\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}t} = -\mathbf{R}(\mathbf{U}) \ . \tag{4.1}$$

Various numerical time-marching schemes can be applied to the solution of Equation (4.1). Common methods such as explicit Euler and Runge-Kutta methods tend to have difficulty when the equations systems are numerically stiff, generally requiring very small time-steps as they are conditionally stable and as such limited by the CFL and von Neumann stability conditions [77]. In this case, the time step selection can be dictated entirely by concerns for stability rather than accuracy considerations. Approaches such as multigrid [175] have been shown to be very good at accelerating convergence for aerodynamic solutions, however in previous studies by Gao *et al.* [14, 47] for the solution of turbulent non-premixed combustion, a multigrid approach was found to still require a relatively large amount of iterations and/or time-steps.

Implicit methods offer a possible way forward for the solution of Equation (4.1) in that they are not restricted by the usual stability conditions and are typically less sensitive to numerical stiffness. The user is then free to select the time step based entirely on the desired accuracy of the solution. The compromise is that implicit methods require greater computation per iteration and typically require a higher memory overhead due to the

resulting linear system of equations that often result and must be solved. For the same reason, they are also typically more complex to deploy and are not as easily implemented in a parallel fashion. However, for many numerically stiff problems, a fully implicit treatment can allow a sufficiently large time step to offset the higher computational costs per iteration and result in a more efficient solution scheme overall.

As outlined in the introduction of Section 1.2.3 of Chapter 1, various implicit methods have been developed and successfully used for the solution of stiff systems of nonlinear ODE's. One of the more promising methods for solving systems of non-linear algebraic equations is Newton-Krylov which has been proposed and formulated for reactive flow in this research. The present chapter describes the details of solving Equation 4.1 using a parallel implicit Newton-Krylov-Schwarz algorithm. Section 4.1 summarizes the inexact Newton's method and the temporal discretization procedure. Section 4.2 describes the Krylov subspace iterative solver GMRES, associated preconditioners, and Jacobian approximations employed for the solution of the resulting linear system of equations. Section 4.5 outlines a start-up algorithm that can be used in conjunction with Newton's method for steady-state solutions.

4.1 Inexact Newton's Method

As noted in the introduction of Chapter 1, the primary focus of the thesis is the numerical solution of unsteady reactive flows that effectively deals with the inherent numerical stiffness encountered in such problems. For this purpose, a parallel implicit time-marching formulation is adopted in which the resulting non-linear algebraic equations are solved via an inexact Newton's method. In this approach the time-dependent solutions of Equation (4.1) are obtained by employing a dual-time-stepping-like procedure [104, 176–178]. In this dual-time approach a pseudo temporal derivative with low-Mach-number preconditioner, $\mathbf{\Gamma}$, is introduced in Equation (4.1) resulting in a modified residual, $\mathbf{R}^*(\mathbf{U})$, as defined by

$$\mathbf{\Gamma}\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}\tau} = -\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}t} - \mathbf{R}(\mathbf{U}) = -\mathbf{R}^*(\mathbf{U}) \ , \tag{4.2}$$

where the physical temporal derivative is now included in the modified residual. As the low-Mach-number preconditioner is applied to the pseudo time derivative, physical time accuracy is preserved for unsteady time-accurate calculations provided a converged

solution satisfying $\mathrm{d}\mathbf{U}/\mathrm{d}\tau=0$ is achieved. To this end, Newton's method is then applied to the solution of $\mathbf{R}^*(\mathbf{U}) = 0$ ensuring recovery of the correct physical time accurate solution. For steady reactive flows, for which $\mathrm{d}\mathbf{U}/\mathrm{d}t=0$, the modified residual $\mathbf{R}^*(\mathbf{U})$ reverts back to $\mathbf{R}(\mathbf{U})$ and steady solutions in pseudo time, τ, satisfy $\mathbf{R}(\mathbf{U}) = 0$ and correspond to the physical steady-state solution. Solution of the steady-state problem is discussed in greater detail in Section 4.5 to follow.

For unsteady reactive flows, an implicit second-order backward temporal discretization (BDF2) scheme is applied in this work to the discretization of the physical time derivative yielding

$$\mathbf{R}^*(\mathbf{U}^{(n+1)}) = \frac{3\mathbf{U}^{(n+1)} - 4\mathbf{U}^{(n)} + \mathbf{U}^{(n-1)}}{2\triangle t} + \mathbf{R}(\mathbf{U}^{(n+1)}) = 0 \ , \tag{4.3}$$

where Δt is the physical time step. Equation (4.3) then represents a system of nonlinear algebraic equations defining the solution at time level $n+1$. Application of Newton's method to the solution of Equation (4.3) leads to the following linear system of equations for the solution update or change, $\Delta\mathbf{U}^{(n+1)} = \mathbf{U}^{(n+1)} - \mathbf{U}^{(n)}$,

$$\left[\left(\frac{3}{2\triangle t}\right)\mathbf{I} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]\triangle\mathbf{U}^{(n+1,k)} = \mathbf{J}^*\Delta\mathbf{U}^{(n+1,k)} = -\mathbf{R}^*(\mathbf{U}^{(n+1,k)}) \ , \tag{4.4}$$

which must be solved at each time level, n, and Newton iteration level, k. Starting with an initial estimate, $\mathbf{U}^{(n+1,k=0)}$, successively improved estimates for the solution, $\mathbf{U}^{(n+1,k)}$, are obtained by solving Equation (4.4) at each step, k, of the Newton method, where $\mathbf{J}^*$ is the modified residual Jacobian. The improved approximation for the solution is given by

$$\mathbf{U}^{(n+1,k+1)} = \mathbf{U}^{(n+1,k)} + \Delta\mathbf{U}^{(n+1,k)} \ . \tag{4.5}$$

A good initial estimate, $\mathbf{U}^{(n+1,k=0)}$, can significantly improve the performance of the Newton's method. An obvious candidate is to use the previous time step as the initial estimate, i.e., $\mathbf{U}^{(n+1,k=0)} = \mathbf{U}^{(n)}$ as it is readily available. Other choices such as polynomial extrapolation where previous solution information is used to generate low-order approximations of the next time step can offer improved initial estimates as demonstrated by Boom and Zingg [179]. For this work, the previous time step proved a sufficient initial estimate for the low-Mach-number reactive flows investigated.

The iterative procedure is repeated until an appropriate norm of the solution residual is sufficiently small, i.e.,

$$||\mathbf{R}^*(\mathbf{U}^{(n+1,k+1)})||_2 < \epsilon||\mathbf{R}^*(\mathbf{U}^{(n+1,k)})||_2 \ , \tag{4.6}$$

where ϵ, is the Newton convergence tolerance. For steady-state solutions typically an ϵ in the range, $\epsilon \approx 10^{-8}$–10^{-10}, is desired to ensure overall solution convergence. For unsteady problems, where each time-step is converged, ϵ can typically be less stringent in the range, $\epsilon \approx 10^{-1}$–10^{-2}, to avoid over-solving; however, this also is dependent on the time-step size.

As noted above, each step of Newton's method requires the solution of a system of linear equations given by Equation (4.4) which can be re-expressed as

$$\mathbf{A}\mathbf{x} = \mathbf{b} \ , \tag{4.7}$$

where $\mathbf{A}$ is the Jacobian $\mathbf{J}^*$, $\mathbf{x} = \Delta\mathbf{U}$ is the solution update and $\mathbf{b} = -\mathbf{R}^*(\mathbf{U})$ is the modified residual right-hand-side vectors. As discussed by Dembo *et al.* [180], an exact solution of the linear system at each step is not necessary for rapid convergence of Newton's method. An iterative solution method can be adopted, as opposed to a direct solver, which can be halted after a specified reduction in the norm of the linear residual resulting. Application of the iterative technique to the linear system leads to an overall solution algorithm with iterations within iterations: the "inner loop" iterations involving the solution of the linear system and the "outer loop" iterations associated with the solution of the nonlinear problem via Newton's method and this is carried out for each time step. The Newton's method is referred to as inexact as the inner iterations are not fully converged at each Newton step. The inner iterations are carried out only until

$$||\mathbf{R}^* + \mathbf{J}^*\Delta\mathbf{U}||_2 \leq \zeta||\mathbf{R}^*||_2 \ , \tag{4.8}$$

where ζ represents the convergence tolerance for the iterative solution of the linear problem and is typically in the range 0.01–0.5.

4.2 Solution of the Linear System of Equations

The resulting linear system of Equations (4.7) is typically a non-symmetric banded matrix that is typically both very large and sparse for large computational mesh. While direct solution of this system of equations is an option, the costs in terms of the processor time and memory storage generally make such an approach prohibitive. Iterative solution

methods such as a Krylov subspace methods offer a more economical approach for large sparse systems. A class of Krylov subspace iterative methods, known as Generalized Minimal RESidual (MRES) methods, was developed previously by Saad and co-workers [85–88] and have been used extensively in many applications for the solution of large sparse linear equations systems [80, 91–94, 96, 112, 181]. Such an approach is used here in the parallel implicit formulation and summarized in this section. Further details can be found found in the text by Saad [82].

4.2.1 GMRES Method

The GMRES method of Saad *et al.* [85] solves a linear system of the form $\mathbf{Ax} = \mathbf{b}$ by finding an approximate solution at every iteration m of the form $\mathbf{x}_m \in \mathbf{x}_o + \mathcal{K}_m$, such that the L_2 norm of the residual $\mathbf{r}_m = \mathbf{b} - \mathbf{Ax}_m$ is minimized. The initial guess is denoted by $\mathbf{x}_o$ and $\mathcal{K}_m$ is the Krylov subspace

$$\mathcal{K}_m(\mathbf{A}, \mathbf{v}_1) = \text{span}\{\mathbf{v}_1, \mathbf{Av}_1, \mathbf{A}^2\mathbf{v}_1, \ldots, \mathbf{A}^{m-1}\mathbf{v}_1\} \ , \tag{4.9}$$

based on the first Krylov vector $\mathbf{v}_1$ that is formed from the initial guess

$$\mathbf{v}_1 = \frac{\mathbf{r}_o}{||\mathbf{r}_o||_2} = \frac{\mathbf{b} - \mathbf{Ax}_o}{||\mathbf{b} - \mathbf{Ax}_o||_2} \ . \tag{4.10}$$

Arnoldi's procedure is applied to the Krylov subspace, $\mathcal{K}_m$, using a modified Gram-Schmidt procedure as follows:

$$\begin{aligned} \text{for} \quad & j = 1, 2, ..., m \\ & \mathbf{h}_{ij} = (\mathbf{Av}_j, \mathbf{v}_i), \ \text{for } i = 1, 2, ..., j \\ & \mathbf{w}_j = \mathbf{Av}_j - \sum_{i=1}^{j} \mathbf{h}_{ij}\mathbf{v}_i \\ & \mathbf{h}_{j+1,j} = ||\mathbf{w}_j||_2 \\ & \mathbf{v}_{j+1} = \mathbf{w}_j / \mathbf{h}_{j+1,j} \end{aligned} \tag{4.11}$$

to construct an orthogonal basis of m column vectors $\mathbf{v}_m$. From Equation (4.11) the quantities $\mathbf{h}_{ij}$ can be related to the $\mathbf{v}_j$, by the expression

$$\mathbf{AV}_m = \mathbf{V}_{m+1}\bar{\mathbf{H}}_m \ , \tag{4.12}$$

where $\mathbf{V}_m = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_m]$ is an $N \times m$ matrix, and $\bar{\mathbf{H}}_m$ is an $(m+1) \times m$ Hessenburg matrix with the entries being the $\mathbf{h}_{ij}$ coefficients.

Once the Krylov vectors $\mathbf{V}_m$ are defined, the approximate solution is computed by minimizing the residual vector. Any vector $\mathbf{x}_m$ in the space $\mathbf{x}_o + \mathcal{K}_m$ can be written as

$$\mathbf{x} = \mathbf{x}_o + \mathbf{V}_m \mathbf{y} \ , \tag{4.13}$$

where $\mathbf{y}$ is a vector of length m. By exploiting the optimality property, one can find the values of vector $\mathbf{y}$ such that the residual norm $r(\mathbf{y}) = ||\mathbf{b} - \mathbf{A}\mathbf{x}||_2$ is minimized.

Considering $r(\mathbf{y})$ and Equation (4.12), one can write

$$\begin{aligned} ||r(\mathbf{y})||_2 &= ||\mathbf{b} - \mathbf{A}\mathbf{x}||_2 \ , \\ &= ||\mathbf{b} - \mathbf{A}(\mathbf{x}_o + \mathbf{V}_m\mathbf{y})||_2 \ , \\ &= ||\mathbf{r}_o - \mathbf{A}\mathbf{V}_m\mathbf{y}||_2 \ , \\ &= ||\beta\mathbf{v}_1 - \mathbf{V}_{m+1}\bar{\mathbf{H}}_m\mathbf{y}||_2 \ , \\ &= ||\mathbf{V}_{m+1}(\beta\mathbf{e}_1 - \bar{\mathbf{H}}_m\mathbf{y})||_2 \ , \end{aligned} \tag{4.14}$$

where $\beta = ||\mathbf{r}_o||_2$ and $\mathbf{e}_1$ is the first column of an $m \times m$ identity matrix. The column vectors of $\mathbf{V}_{m+1}$ are orthonormal, so that

$$||r(\mathbf{y})||_2 = ||\beta\mathbf{e}_1 - \bar{\mathbf{H}}_m\mathbf{y}||_2 \ . \tag{4.15}$$

This form can be easily minimized as this is a $(m+1) \times m$ least-squares problem and the structure of the Hessenburg matrix $\bar{\mathbf{H}}$ is such that simple plane rotations transform it into an upper triangular matrix. The approximate solution is thus given by $\mathbf{x}_m = \mathbf{x}_o + \mathbf{V}_m\mathbf{y}_m$, where $\mathbf{y}_m$ minimizes the function $r(\mathbf{y})$ in Equation (4.15).

Restarted GMRES Method

An attractive property of the GMRES method is that it can compute the norm of the residual as each new search direction is introduced without explicitly forming the approximate solution. As well, the iterative GMRES algorithm is guaranteed to converge for well-conditioned systems in at most N steps or Krylov subspace search directions, where N is the system size. However the storage requirements increase linearly with the number of steps and the computational work increases quadratically. To provide control of

memory and computational costs, Saad and Schultz [86] devised a variant of the GMRES algorithm, restarted GMRES(m), where after a certain number of search directions, m iterations, an approximate solution is formed which becomes the initial guess for the next GMRES solve. Care must be taken in selecting a value for m to ensure good convergence properties of the GMRES and Newton iterative methods. A typical value for m used here is in the range of 20–40 and in general the other GMRES parameters are selected as to avoid the need for a restart in the majority of cases.

Jacobian-Free Approach

Another very useful feature of the GMRES algorithm is that it does not explicitly require the evaluation of the global matrix, $\mathbf{A}$. It only requires the evaluation of the matrix-vector product, $\mathbf{Av}$, as shown in the third step of Equation (4.11). This permits the use of a so-called "matrix-free" or in this case "Jacobian-free" approach in which numerical differentiation based on Fréchet derivatives is used to approximate this matrix-vector product [80, 87, 93–95, 114, 115], which when applied to Equation (4.4) yields

$$\mathbf{Av} \approx \frac{\mathbf{R}(\mathbf{U} + \varepsilon\mathbf{v}) - \mathbf{R}(\mathbf{U})}{\varepsilon} + \frac{3\mathbf{v}}{2\triangle t}\,, \tag{4.16}$$

where $\mathbf{R}(\mathbf{U} + \varepsilon\mathbf{v})$ is the physical residual vector evaluated at some perturbed solution state and ε is a small scalar quantity. The term, $3\mathbf{v}/2\triangle t$, in Equation (4.16) results from the BDF2 temporal discretization of the dual-time-stepping approach. Use of the approximation of Equation (4.16) yields a so-called "Jacobian-free" inexact Newton method and is used herein. Although the performance of the Jacobian-free method is sensitive to the choice of ε, Neilsen *et al.* [80] have found that $\varepsilon = \varepsilon_\circ / ||\mathbf{v}||_2^{1/2}$ seems to work well, with $\varepsilon_\circ \approx 10^{-8}$–$10^{-7}$, and this expression is used in the current implementation.

Row Scaling of the Linear Problem

The GMRES iterative solution method is sensitive to scaling of the linear equations. In order to obtain a more robust and efficient solver, row scaling can be applied to the linearized system of Equation (4.7) such that the equations are more appropriately scaled for numerical computation. The strategy is to scale each equation so that their coefficients are all of similar magnitude.

Row scaling can be accomplished by scaling the conserved variable $\mathbf{U}$ and solution residual $\mathbf{R}$ vectors within each computational cell such that $\mathbf{U}' = \boldsymbol{\Upsilon}_u \mathbf{U}$ and $\mathbf{R}' = \boldsymbol{\Upsilon}_r \mathbf{R}$ where $\boldsymbol{\Upsilon}_u$ and $\boldsymbol{\Upsilon}_r$ are diagonal matrices containing the scaling coefficients. Applying this scaling procedure to the linearized system, Equation (4.7) becomes

$$\left(\boldsymbol{\Upsilon}_r \mathbf{J}^* \boldsymbol{\Upsilon}_u^{-1}\right)\left(\boldsymbol{\Upsilon}_u \mathbf{x}\right) = \left(\boldsymbol{\Upsilon}_u \mathbf{R}^*\right), \tag{4.17}$$

and this prescaled linear problem is then solved in place of the original system.

For this work, the diagonal scaling matrices applied to each computational cell take the form

$$\boldsymbol{\Upsilon}_{u,ijk} = \begin{bmatrix} \frac{1}{\rho_o} & & & & & & & & \\ & \frac{1}{\rho_o a_o} & & & & & & & \\ & & \frac{1}{\rho_o a_o} & & & & & & \\ & & & \frac{1}{\rho_o a_o} & & & & & \\ & & & & \frac{1}{\rho_o a_o^2} & & & & \\ & & & & & \frac{1}{\rho_o} & & & \\ & & & & & & \ddots & & \\ & & & & & & & \frac{1}{\rho_o} \end{bmatrix}, \tag{4.18}$$

and

$$\boldsymbol{\Upsilon}_{r,ijk} = \begin{bmatrix} \frac{1}{\rho_o a_o} & & & & & & & & \\ & \frac{1}{\rho_o a_o^2} & & & & & & & \\ & & \frac{1}{\rho_o a_o^2} & & & & & & \\ & & & \frac{1}{\rho_o a_o^2} & & & & & \\ & & & & \frac{1}{\rho_o a_o^3} & & & & \\ & & & & & \frac{1}{\rho_o a_o} & & & \\ & & & & & & \ddots & & \\ & & & & & & & \frac{1}{\rho_o a_o} \end{bmatrix}, \tag{4.19}$$

where ρ_o and a_o are reference pressure and sound speed respectively. When the scaling is applied to the conserved variables in each cell, $\mathbf{U}$, as defined previously by Equation (3.13), the scaled solution vector, $\mathbf{U}'$, becomes

$$\mathbf{U}' = \left[\begin{array}{cccccccc} \frac{\rho}{\rho_o}, & \frac{\rho u}{\rho_o a_o}, & \frac{\rho v}{\rho_o a_o}, & \frac{\rho w}{\rho_o a_o}, & \frac{\rho E}{\rho_o a_o^2}, & \frac{\rho c_1}{\rho_o}, & \ldots, & \frac{\rho c_N}{\rho_o} \end{array} \right]^T, \tag{4.20}$$

and the corresponding scaled solution residual, $\mathbf{R}'$, has the form

$$\mathbf{R}' = \left[\begin{array}{cccccccc} \frac{R_1}{\rho_o a_o}, & \frac{R_2}{\rho_o a_o^2}, & \frac{R_3}{\rho_o a_o^2}, & \frac{R_4}{\rho_o a_o^2}, & \frac{R_5}{\rho_o a_o^3}, & \frac{R_6}{\rho_o a_o}, & \ldots, & \frac{R_N}{\rho_o a_o} \end{array} \right]^T. \tag{4.21}$$

4.2.2 Right Preconditioning of System

For many practical applications, the Jacobian matrix, $\mathbf{J}^*$, is ill-conditioned even after scaling and preconditioning is required for GMRES to be effective. Saad observes that effective preconditioning is as important as the choice of the Krylov method [82]. Although the preconditioner can be applied from either side of $\mathbf{J}^*$, right preconditioning is considered here which can be expressed as

$$(\mathbf{J}^*\mathbf{M}^{-1})(\mathbf{M}\mathbf{x}) = \mathbf{b}\,, \tag{4.22}$$

where $\mathbf{M}$ is the preconditioning matrix. A convenience of right preconditioning is that the solution residual is unaffected by the preconditioning. Saad [88] indicates that the choice of the side for the preconditioner should not significantly impact GMRES convergence, provided $\mathbf{M}$ is itself not poorly conditioned.

A variety of preconditioning methods are possible. The ideal preconditioner, $\mathbf{M}$, will provide a good approximation to $\mathbf{J}^{*-1}$ ($\mathbf{M}^{-1} \approx \mathbf{J}^{*-1}$) while being significantly easier (i.e., computationally inexpensive) to invert than $\mathbf{J}^*$. A good preconditioner will cluster the eigenvalues of the system matrix and thereby reducing the number of steps as GMRES essentially devotes one step to each cluster of eigenvalues [108]. Obviously, there is a trade-off between the cost of constructing and applying the preconditioner and the gain in convergence rate of the GMRES algorithm. Knoll and Keyes [101] discuss various methods of preconditioning and observe that within the Newton-Krylov framework, preconditioning offers the most possibilities for ensuring an efficient implementation.

In the proposed parallel implicit algorithm, a combination of global and local preconditioning techniques is used. In particular, an additive Schwarz global preconditioner with variable overlap is used in conjunction with block incomplete lower-upper (BILU) local preconditioning. This combination of preconditioning fits well with the block-based AMR and domain decomposition described in previous Chapter 3.2, readily enabling parallel implementation of the overall method.

Details of the formation of these two preconditioners and the approximate analytical Jacobian are given in the sections that follow.

4.2.3 Global Additive Schwarz Preconditioner

Schwarz [182] originally developed his domain-decomposition method to solve boundary-value problems of partial differential equations whereby the solution on part of the solution domain was solved and then the values at the interface are taken as updated Dirichlet boundary conditions on another part of the domain. One sweep of Schwarz's procedure can be readily viewed as the action of a preconditioner. Schwarz preconditioning has been utilized extensively by Keyes and co-researchers and successfully applied to the prediction of transonic full potential, low-Mach-number compressible combusting, and three-dimensional inviscid flows [101, 110, 111, 114, 115].

A global additive Schwarz preconditioner for N_{blocks} solution blocks can be defined as follows:

$$\mathbf{M}^{-1} = \sum_{k=1}^{N_{\text{blocks}}} \mathbf{B}_k^{\mathrm{T}} \mathbf{M}_k^{-1} \mathbf{B}_k \,, \tag{4.23}$$

where $\mathbf{B}_k$ is the gather operator or matrix for the k^{th} domain that gathers the solution unknowns for the domain from the global solution vector, show graphically in Figure 4.1. Since the application of each of the sub-preconditioners, $\mathbf{M}_k$, proceeds without regard to other domains this is referred to as additive (and not multiplicative) Schwarz preconditioning. The difference between a multiplicative and additive Schwarz procedure is analogous to the difference between the Gauss-Seidel and Jacobi linear iterative solution methods.

In general, domain overlap is permitted in Schwarz preconditioning. The use of overlapping subdomains can help to offset the loss of overall implicitness of the Newton iterative solver introduced by the block-based Schwarz preconditioning; however, in practice it was found here to not result in an overall faster Newton scheme in terms of computational cost. While reducing iteration counts, the use of overlap was found to require more computational effort and was therefore not used here.

The additive Schwarz preconditioner fits very well with block-based AMR as the same domain decomposition that is used for the AMR can be used in the equation preconditioning without additional work or logic. It also leads to a fully parallel implicit approach with no global serial computations or solution of additional sub-problems, allowing the efficient scalability of the algorithm to a very high number of processes cores for both weak and strong scaling problems without suffering from Amahdal's law and a degradation

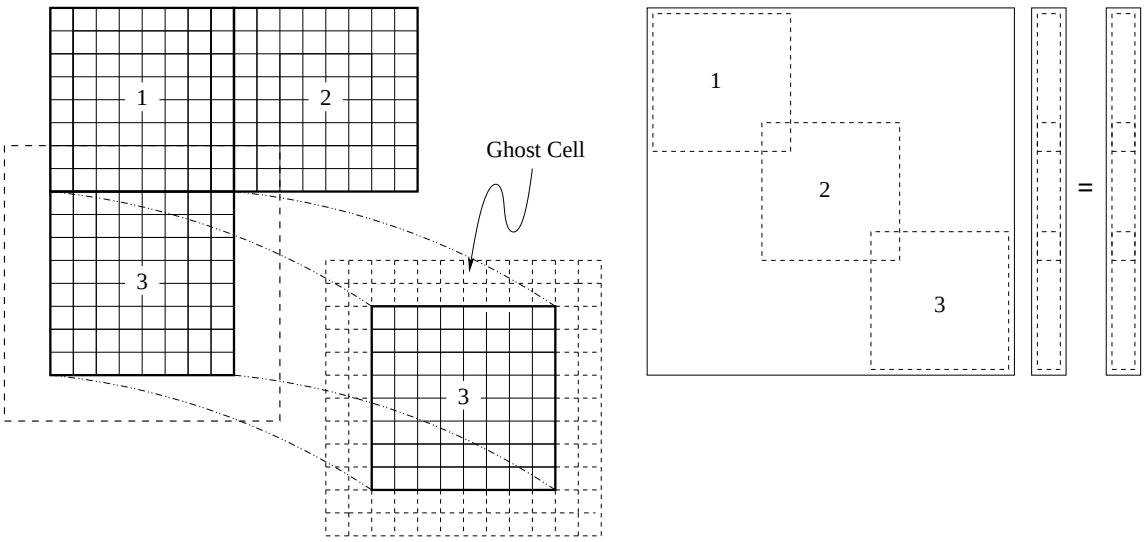


Figure 4.1: Representation of Additive Schwarz preconditioner

in performance as may be encountered with other methods such as Schur complement methods [106].

Use of a Schwarz method as a preconditioner is however not without a cost as typically the Schwarz preconditioning results in an increase in the total number of GMRES iterations required compared to the non-preconditioned computation. This increased cost; however, can be fairly modest when compared with the overall gains from the ability to solve the system in parallel and is investigated in more detail in Chapter 6.

4.2.4 Local BILU Preconditioner

In Equation (4.23), $\mathbf{M}_k^{-1}$ is the local block preconditioner for the subdomain k. In this work the local the local preconditioner is determined via incomplete lower-upper factorization (ILU) [88] of an approximate Jacobian of each sub-domain, $\tilde{\mathbf{J}}_k$. The block ILU(f) or BILU(f) factorization of $\tilde{\mathbf{J}}_k$ where $\mathbf{M}_k$ is given by

$$\mathbf{M}_k = \mathbf{L}_k\mathbf{U}_k \approx \tilde{\mathbf{J}}_k \ , \tag{4.24}$$

and where $\mathbf{L}_k$ and $\mathbf{U}_k$ are sparse lower and upper triangular matrices. The accuracy of the incomplete LU factorization is determined by the level of fill, f, for the approximate inverse. With higher fill levels, more non-zero entries are retained in $\mathbf{L}_k$ and $\mathbf{U}_k$ more

closely resembling the sparsity pattern of the local Jacobian matrix providing a more accurate representation for $\tilde{\mathbf{J}}_k$; however, this is at the cost of greater computational work and storage. Although the existence and stability of ILU(f) factorizations has only been established for a restricted class of matrices [183], the approach has been applied to a wide range of systems and McHugh et al. [115] and Gropp *et al.* [111] have shown that ILU factorization can be an effective local preconditioner for parallel NKS algorithms. In practice, especially for the 3D problems of interest considered herein, fill levels of 0 or 1 are typically used as the extra storage and expense for using a higher fill level does not result in an overall reduction in solution time. The effects of ILU fill-level on solution convergence are investigated in Chapter 6.

4.2.5 Approximate Jacobian of Solution Residual

As mentioned, the BILU(f) preconditioner acts on an approximate Jacobian for each sub-domain, $\tilde{\mathbf{J}}_k$. As $\tilde{\mathbf{J}}_k$ is an approximation to the Jacobian $\mathbf{J}^*$, given in Equation (4.4), and restated here

$$\mathbf{J}^* = \left(\frac{3}{2\triangle t}\right)\mathbf{I} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}} \ , \tag{4.25}$$

it can be determined by an approximation to $\partial\mathbf{R}/\partial\mathbf{U}$ with the addition of $\frac{3}{2\triangle t}$ along the diagonal introduced by the BDF2 temporal discretization. The residual, $\mathbf{R}$, in Equation (4.1) is composed of three major components, as outlined in Chapter 3; those terms associated with the inviscid and viscous solution fluxes, $\mathbf{F}$ and $\mathbf{F_v}$, and source term, $\mathbf{S_p}$, respectively. As such the approximate Jacobian, $\tilde{\mathbf{J}}_k$, is comprised of approximations for each of these components which are outlined in the following subsections.

Approximate Inviscid Jacobian

As the inviscid flux, $\vec{\mathbf{F}}$, is computed using an Godunov-type upwind scheme, Section 3.1.3, the Jacobian is approximated using just the first order terms of the upwind discretization procedure. This approach has been used previously by many researchers [107, 110] and provides a relatively inexpensive yet effective preconditioner.

To determine the contributions of the inviscid flux to the approximate Jacobian, consider first the Jacobian of $\mathbf{R}^*$ in one cell (i, j, k) with respect to the variables in that same cell,

$\mathbf{U}_{i,j,k}$. For the inviscid flux functions, the frame of reference is first rotated to a local frame in the x-direction, the flux at the interface is evaluated and then the frame of reference is rotated back to the original orientation. This process can be written as

$$\vec{\mathbf{F}} \cdot \vec{n} = \mathcal{F}(\mathbf{U}_L, \mathbf{U}_R, \mathbf{n}) = \mathbf{A}^{-1}\mathcal{F}(\mathbf{A}\mathbf{U}_L, \mathbf{A}\mathbf{U}_R) \ , \tag{4.26}$$

where $\mathbf{A}$, is a rotation matrix that rotates the momentum vector leaving the mass and energy fluxes unchanged The matrix $\mathbf{A}^{-1}$ is the inverse of $\mathbf{A}$ which is also equal to the transpose of $\mathbf{A}$ since $\mathbf{A}$ is orthogonal. By definition of the Riemann problem, the left state as input to the flux function $\mathcal{F}$ is the solution vector corresponding to cell (i, j, k). Thus the Jacobian of $\mathbf{R}_{i,j,k}$ with respect to $\mathbf{U}_{i,j,k}$ is

$$\frac{\partial \mathbf{R}_{i,j}}{\partial \mathbf{U}_{i,j}} = -\frac{1}{V_{i,j,k}} \sum_{k \text{ faces}} \left(\mathbf{A}^{-1} \frac{\partial \mathcal{F}(\mathbf{A}\mathbf{U}_L, \mathbf{A}\mathbf{U}_R)}{\partial(\mathbf{A}\mathbf{U}_L)} \mathbf{A} \ \Delta\mathbf{A} \right)_{i,j,k} \ , \tag{4.27}$$

where the chain rule is used to write

$$\frac{\partial \mathcal{F}(\mathbf{A}\mathbf{U}_L, \mathbf{A}\mathbf{U}_R)}{\partial \mathbf{U}_L} = \frac{\partial \mathcal{F}(\mathbf{A}\mathbf{U}_L, \mathbf{A}\mathbf{U}_R)}{\partial(\mathbf{A}\mathbf{U}_L)} \frac{\partial(\mathbf{A}\mathbf{U}_L)}{\partial \mathbf{U}_L} = \frac{\partial \mathcal{F}(\mathbf{A}\mathbf{U}_L, \mathbf{A}\mathbf{U}_R)}{\partial(\mathbf{A}\mathbf{U}_L)} \mathbf{A} \ , \tag{4.28}$$

The term $\partial \mathcal{F}(\mathbf{A}\mathbf{U}_L, \mathbf{A}\mathbf{U}_R)/\partial(\mathbf{A}\mathbf{U}_L)$ is the Jacobian of the flux function with respect to its first argument (that is, the "left" state solution vector) and evaluated in the rotated frame of reference (i.e., local x-direction aligned with face normal). Since the upwind finite-volume method is conservative and each term in the sum of Equation (4.27) (scaled by the ratio of the cell volumes) also describes the neighbouring cell's Jacobian with respect to the variables in cell (i, j, k). In such a way, the Jacobians of $\mathbf{R}_{i,j,k}$ with respect to the cell centered solution values in all neighbouring cells can be determined using Equation (4.27).

Thus the approximate Jacobian of the Roe flux function, including the modification for low-Mach-number preconditioning, as given in Equation 3.41, then becomes

$$\frac{\partial \mathcal{F}_{\text{Roe}}(\mathbf{A}\mathbf{U}_L, \mathbf{A}\mathbf{U}_R)}{\partial \mathbf{U}_L} \approx \frac{1}{2} \frac{\partial \mathbf{F}_L}{\partial \mathbf{U}_L} + \frac{1}{2} \mathbf{\Gamma} |\hat{\mathbf{A}}_\Gamma| \ . \tag{4.29}$$

Approximate Viscous Jacobian

As discussed for the inviscid case, the finite volume method is conservative so we need only determine the Jacobian of the residual $\mathbf{R}$ of Equation (4.1) with respect to the

variables of cell (i, j, k) and then for each face, add this Jacobian to the Jacobian of the neighbouring cell with respect to the variables of cell (i, j, k).

For the viscous operator, the term $\vec{\mathbf{F}_\mathbf{v}} \cdot \vec{n}$ in the equation for the residual is given by Equation (3.42), which is repeated here:

$$\vec{\mathbf{F}_\mathbf{v}} \cdot \vec{n} = \mathbf{F}_\mathbf{v} n_x + \mathbf{G}_\mathbf{v} n_y + \mathbf{H}_\mathbf{v} n_z \ , \tag{4.30}$$

where n_x, n_y, and n_z are the components of $\vec{n}$. Only the x direction Jacobian $\partial \mathbf{F}_\mathbf{v} / \partial \mathbf{U}$ is described in detail; the other Jacobians $\partial \mathbf{G}_\mathbf{v} / \partial \mathbf{U}$ and $\partial \mathbf{H}_\mathbf{v} / \partial \mathbf{U}$ can be approximated readily in a similar manner. The other terms in the equation for the residual are the cell area and length of each face which are constant with respect to the solution variables.

The evaluation of the viscous flux is a three-step process. First, the cell-centered conserved variables, $\mathbf{U}_c$, are converted to cell-centered primitive variables, $\mathbf{W}_c$. Then bilinear interpolation and diamond-path reconstruction are used to approximate the primitive variables and their gradients on a given cell interface, which are stored in an extended solution vector referred to here as $\mathbf{E}_f$. The viscous flux is finally evaluated. Thus $\partial \mathbf{F}_\mathbf{v} / \partial \mathbf{U}$ is performed through a three-step application of the chain rule:

$$\frac{\partial \mathbf{F}_\mathbf{v}}{\partial \mathbf{U}_c} = \left(\frac{\partial \mathbf{F}_\mathbf{v}}{\partial \mathbf{E}_f} \right) \left(\frac{\partial \mathbf{E}_f}{\partial \mathbf{W}_c} \right) \left(\frac{\partial \mathbf{W}_c}{\partial \mathbf{U}_c} \right) \ , \tag{4.31}$$

$\partial \mathbf{W}_c / \partial \mathbf{U}_c$ is the standard Jacobian of the primitive variables with respect to the conserved variables and is readily available. The Jacobian, $\partial \mathbf{E}_f / \partial \mathbf{W}_c$, is due to geometry only, for it only depends on the interpolation used in the viscous flux evaluation described in Section 3.1.4 of Chapter 3.

In the 3D case, the face variables at the cell interface are given by the extended solution vector

$$\begin{aligned} \mathbf{E}_f = \Bigg[& \rho,\ u,\ v,\ w,\ p,\ c_1, ...,\ c_N, \frac{\partial \rho}{\partial x},\ \frac{\partial u}{\partial x},\ \frac{\partial u}{\partial y},\ \frac{\partial u}{\partial z}, \\ & \frac{\partial v}{\partial x},\ \frac{\partial v}{\partial y},\ \frac{\partial v}{\partial z},\ \frac{\partial w}{\partial x},\ \frac{\partial w}{\partial y},\ \frac{\partial w}{\partial z},\ \frac{\partial p}{\partial x},\ \frac{\partial c_1}{\partial x}, ...,\ \frac{\partial c_N}{\partial x} \Bigg]^T , \end{aligned} \tag{4.32}$$

where ρ is the density, u, v, and w are the components of the velocity vector, p is pressure and c_n are the species mass fractions. With the solution vector of cell-centered primitive variables given as

$$\mathbf{W}_c = [\rho,\ u,\ v,\ w,\ p,\ c_1, ...,\ c_N]^T, \tag{4.33}$$

thus $\partial \mathbf{W}_f / \partial \mathbf{W}_c$ can be easily determined. Finally we need to evaluate $\partial \mathbf{F_v} / \partial \mathbf{E}_f$ which follows from using Equations (3.16) and (4.32). To simplify the calculations, the viscosity, μ, thermal conductivity, κ, and finite-rate chemistry are taken to be constants and invariant with respect to the solution variables when evaluating the Jacobian.

Approximate Source Term Jacobian

The Jacobian components related to the source terms, Equation (3.14), is straight forward as the chemical and gravitational terms only rely on the solution values for local cell i, j, k. Thus, the contribution to the residual Jacobian can be calculated simply as

$$\frac{\partial \mathbf{S}}{\partial \mathbf{U}_{ijk}} \ . \tag{4.34}$$

One complication that arose was related to the calculation of the derivative of the time rate of change of the species concentrations, ω_N, $\partial \dot{\omega}_{\mathbf{N}} / \partial \mathbf{U}_{ijk}$. As shown in Equation (2.24), the time rate of change of species concentrations includes two stoichiometric coefficients which are powers of the species concentrations. When the values of these parameters are greater than one, there is no problem with the evaluation of the derivatives numerically; however, when the exponents are less than one, as is the case in some of the simple mechanisms of Table (2.1), concentration terms appear in the denominator of the Jacobian. As the concentration approaches zero, this leads to divide by zero issues and near singular values making the numerical evaluation of the source term Jacobians inaccurate. To alleviate this issue, all calculations for this research were done with the second one-step methane-air mechanism of Table (2.1) which has integer coefficients for exponents in the expressions for the reaction rates. This does not limit the algorithms applicability for use with complex mechanisms, as more complex and physically accurate mechanisms only have integer coefficients greater than one. The problem with non-integer values of the exponent is only an issue caused by the reduced mechanisms being based on experimental curve fits.

4.2.6 GMRES Performance Diagnostics

Determining why a solution fails to converge can be difficult when using a Newton-Krylov method. For steady problems, it may be as simple as being overly aggressive in the startup

phase. However, more subtle problems typically related to GMRES convergence can be difficult to pinpoint. One technique useful in diagnosing GMRES convergence issues, as discussed by Chisholm [184], is to monitor the accuracy of the GMRES residual reduction. This can be done by evaluating the non-dimensional residual

$$\bar{r} = \frac{||\mathbf{Ax} - \mathbf{b}||}{||\mathbf{b}||} \ , \tag{4.35}$$

where $||\mathbf{Ax} - \mathbf{b}||$ is the linear residual. As $\mathbf{Ax}$ and $\mathbf{b}$ have been calculated in the final GMRES iteration the cost of evaluating $\bar{r}$ is just the cost of the calculating the norms of the two vectors. If the matrix-vector products have been perfectly formed, then the norm of the linear residual should exactly equal that calculated by the GMRES method and the $\bar{r}$ would equal the GMRES convergence tolerance. However, if the reduction is greater than the GMRES convergence tolerance, then some level of matrix-free breakdown has occurred. If the breakdown is significant the linear system solution will be inaccurate which typically leads to divergence of the Newton method. The breakdown is generally caused by inaccurate Fréchet derivatives, errors in the approximate Jacobian, or improper scaling of the equations.

To provide further insight, it is often useful to look at the reduction of the residual of each equation in the system. Ideally the reduction of each equation would be equal to that of the overall GMRES convergence tolerance, however if this is not the case then this can be a sign of improper scaling parameter selection, as discussed in Section 4.2.1. It can also point to errors or inaccuracies with the associated approximate Jacobian terms.

4.3 Implementation of Linear Solver

The BILU local preconditioner was implemented using the Block Preconditioning Toolkit (BPKIT) package developed by Chow and Heroux [185]. BPKIT incorporates many of the preconditioners from SPARSKIT2 developed Saad *et al.* [186] as well as its own and is extensible providing a framework to allow a user to implement other preconditioners. For compatibility with this package and future algorithm flexibility, the actual GMRES implemented is a flexible variant called FGMRES [187]. FGMRES allows the preconditioner to vary between each iteration allowing a wider range of preconditioner choices.

The computational cost is equivalent; however, the preconditioned vectors $\mathbf{M}^{-1}\mathbf{v}_j$ now must be stored for each iteration, resulting in FGMRES having a memory footprint that is approximately double that of the standard GMRES algorithm.

4.4 Storage Requirements

As mentioned in the introduction to this chapter, one of the trade-offs or challenges faced when using an implicit method in place of an explicit method is that of a higher memory overhead for the implicit method. In this work, the higher memory overhead was not a major concern as most cases considered were more computationally bound than memory bound so the memory available per processor core on the systems used was significantly greater than necessary to perform the calculations. However, for comparison purposes, an examination of the proposed implicit algorithms memory usage is provided here.

A single block 3D cube grid consisting of 24×24×24 (13,824) cells was used as the basis for the evaluation of the memory usage of the proposed 3D parallel implicit solver with ILU(0) preconditioning and GMRES(40). Reactive flow with 5, 10, 15, and 20 species for a total of 10, 15, 20, and 25, equations per cell was examined. Using the memory heap profiling tool "Massif" which is part of the valgrind [188] toolkit, the memory usage of the implicit NKS algorithm was determined. The results of this profiling are summarized in Table 4.1 along with an equivalent number of variables per cell metric which is calculated by dividing the total memory by the number of cells and by the size of a double precision floating point number, 8 bytes in this instance. This metric can be useful when comparing memory requirements across differing algorithms and platforms.

Equations per cell	Memory (MB)					Equivalent Number Variables per cell
	Other	Approx. Jacobian	ILU(0)	GMRES(40)	Total	
10	22	60	68	95	245	2,215
15	28	146	168	141	483	4,367
20	34	269	308	188	799	7,225
25	39	430	494	233	1196	10,815

Table 4.1: Comparison of memory storage requirements of the 3D NKS algorithm for a single block 13,824 cell grid for a variable number of equations per cell.

As expected, it is evident from Table 4.1 that in this implementation the storage requirements are dominated by: (i) the various storage components of the GMRES algorithm described in Section 4.2.1; (ii) the preconditioner, $\mathbf{M}_k^{-1}$, described in Section 4.2.4; and (iii) the approximate Jacobian, $\tilde{\mathbf{J}}_k$, described in Section 4.2.5. The total memory appears to grow approximately quadratically with the number of equations, primarily due to the approximate Jacobian and ILU preconditioner components. For the reduced chemical mechanisms of methane considered in this work, the resulting memory overhead was not of particular concern; however; for future computations involving detailed chemistry with a large number of species memory overhead may be an important factor to consider.

4.5 Steady-State Startup Algorithm

For the solution of steady-state problems, a good startup or globalization algorithm is invariably required in order to increase the radius of convergence and ensure global convergence of the NKS method. While this is not the primary focus here, development of the NKS algorithm required evaluation and validation for several steady problems. Several different startup strategies have been proposed in the literature [80,92,94,97,181]. One approach that is often adopted and has proved to be an effective startup procedure is an implicit Euler time-marching method with switched evolution/relaxation (SER) as proposed by van Leer and Mulder [189].

As described in Section 4.1 for time invariant solutions where $\mathrm{d}\mathbf{U}/\mathrm{d}t{=}0$, Equation (4.2) simplifies to

$$\mathbf{\Gamma}\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}\tau} + \mathbf{R}(\mathbf{U}) = 0 \ . \tag{4.36}$$

The application of an implicit Euler time-marching method for the pseudo-time derivative yields

$$\left[-\frac{\mathbf{\Gamma}}{\Delta\tau^n} + \left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)^n\right] \Delta\mathbf{U}^n = -\mathbf{R}^n \ . \tag{4.37}$$

where $\Delta\tau^n$ is the time step. As $\Delta\tau^n \rightarrow \infty$, Newton's method of Equation (4.4) is recovered.

In the SER approach, the time step is varied, starting from a finite-value and gradually

increasing and becoming very large as the desired steady solution is approached. As the time step becomes large, Newton convergence is achieved. As a finite time step is initially used the low-Mach-number preconditioner, $\mathbf{\Gamma}$, is most influential during the intial startup and the temporal preconditioning becomes less dominant as the time step becomes large.

For the solution of steady time-invariant problems in this work, the SER approach has been used. A time step multiplier, ν^n, is introduced that increases as the solution residual, $\mathbf{R}$, decreases with a adjustable minimum or initial multiplier, $\nu^{\min}$,

$$\nu^n = \nu^{\min} \max\left(1, \frac{1}{||\mathbf{R}||_2}\right) \ . \tag{4.38}$$

This multiplier is applied in conjunction with a series of stability criteria to determine the time step,

$$\Delta\tau^n = \nu^n \min\left(\frac{\Delta x}{u+a}, \frac{\rho\Delta x^2}{\mu}, \left(\max\left(\frac{\partial \mathbf{S_p}}{\partial \mathbf{U}}\right)\right)^{-1}\right) \ . \tag{4.39}$$

The CFL, $\Delta x/(u+a)$, and von Neumann, $\rho\Delta x^2/\mu$, stability criteria are considered for inviscid and viscous flows, respectively. The inverse of the maximum diagonal of the chemical source term Jacobian, $\partial\mathbf{S_p}/\partial\mathbf{U}$, is incorporated as a measure of reaction time scales for reactive flow cases.

Chapter 5

Numerical Results: Validation

This chapter discusses the set of numerical results used in the verification and validation of the proposed parallel implicit AMR algorithm for reactive flows. A series of well characterized inviscid, viscous, and reactive flow problems are described in Sections 5.1–5.3 and are used to verify that each component of the numerical scheme produces physically correct results. The influence of various aspects of the proposed algorithm, such as AMR and low-Mach-number preconditioning, on solution accuracy are also investigated. Where possible, predicted solutions are compared with analytical solutions and/or experimental data to validate the numerical results.

Sections 5.4 and 5.5 then describe the application of the parallel finite-volume scheme to the prediction of both steady and unsteady 3D laminar diffusion and premixed methane-air flames. The solutions are compared against published numerical and experimental results and, where applicable, against 2D axisymmetric solutions. The application of the algorithm to a rather wide range of laminar combustion problems and regimes is used to show the functionality, robustness, and range of applicability of the parallel implicit AMR algorithm.

As this chapter focuses primarily on verifying and validating the algorithm's accuracy and ability to produce physically correct solutions, details of the algorithms computational and parallel performance have been left for discussion in Chapter 6 to follow.

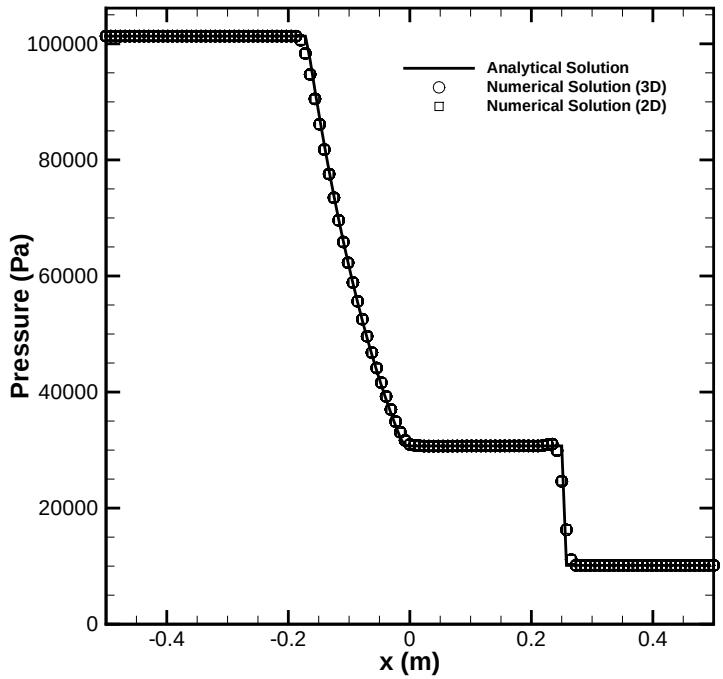
5.1 Inviscid Flow

5.1.1 Unsteady One-Dimensional Shock-Tube Problem

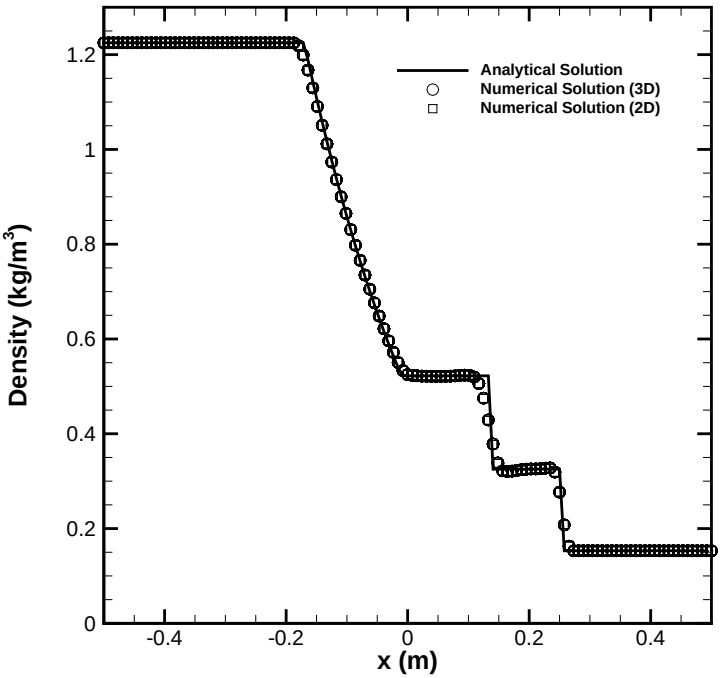
The verification of the inviscid finite-volume spatial discretization operators used herein have been previously and quite extensively studied for both two-dimensional [63,190] and three-dimensional flows [64]. As verification of the implementation and accuracy of the combined inviscid spatial and temporal discretization operators applied in the proposed parallel implicit algorithm, the solution of a one-dimensional shock-tube problem for a non-reacting inviscid gas is first compared with its analytical solution. As this solution is time variant, it is also particularly helpful in verifying the implementation of the BDF2 temporal discretization scheme.

A one-dimensional shock-tube problem is initialized with two separated regions (left and right) of quiescent air, defined as 79% nitrogen, N_2, and 21% oxygen, O_2, by volume, at different pressures (p_L/p_R=10) and densities (ρ_L/ρ_R=8). The shock-tube problem was solved time accurately using the BDF2-NKS method described in Section 4.1, until time t=0.5 ms. For each outer physical time step, the Newton step was converged two orders of magnitude with an inner GMRES tolerance of 0.1 and the ILU preconditioner with a fill of level 2 was used. No Schwarz preconditioning was used as only a single computational block was used and the problem was solved serially. The approximate Jacobian preconditioner was only updated for the first Newton step of each time-step, or if the number of GMRES iterations per Newton-step increased over the previous steps number of GMRES iterations required for inner loop convergence. The physical time-step was determined using the standard CFL criteria for inviscid flow as dictated by Equation (4.39), with a CFL=0.5.

The Roe flux function with the Barth-Jespersen limiter was used to solve the problem on a 1 m $\times$ 1 m computational grid of 128$\times$2 cells using the 2D algorithm, and on a 1 m $\times$ 1 m $\times$ 1 m computational grid of 128$\times$2$\times$2 cells using the 3D algorithm. The predictions of the density, pressure, Mach number and velocity in both cases are compared to the analytic solution in Figures 5.1 and 5.2. It can be seen that the two- and three-dimensional results are virtually identical and agree very well with the analytical solution. Although very slight non-monotone behaviour is observed in several of the derived quantities, for the most part the shocks and discontinuities are well resolved and oscillation free.

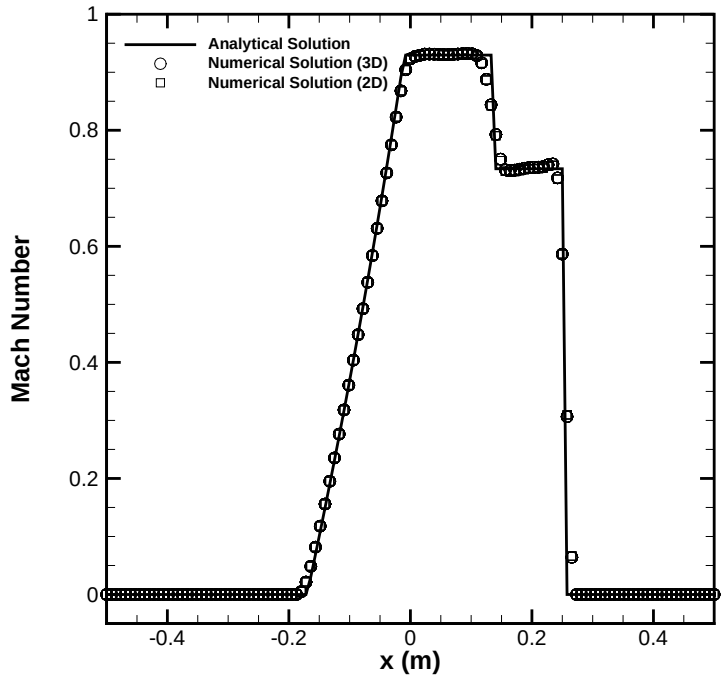


(a) Pressure

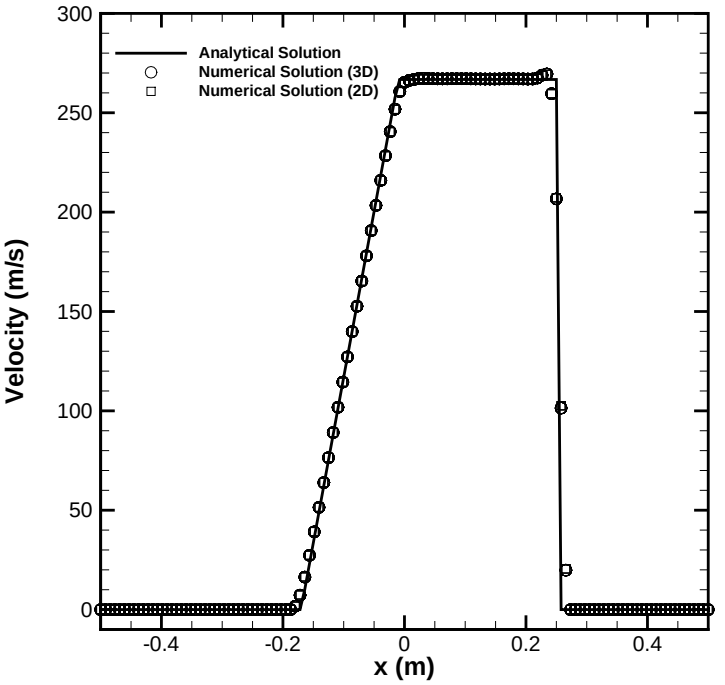


(b) Density

Figure 5.1: Comparison of 2D and 3D BDF2-NKS predicted solutions for a 1D shock tube problem at t=0.5 ms to the exact analytical solution.



(a) Mach Number



(b) Velocity

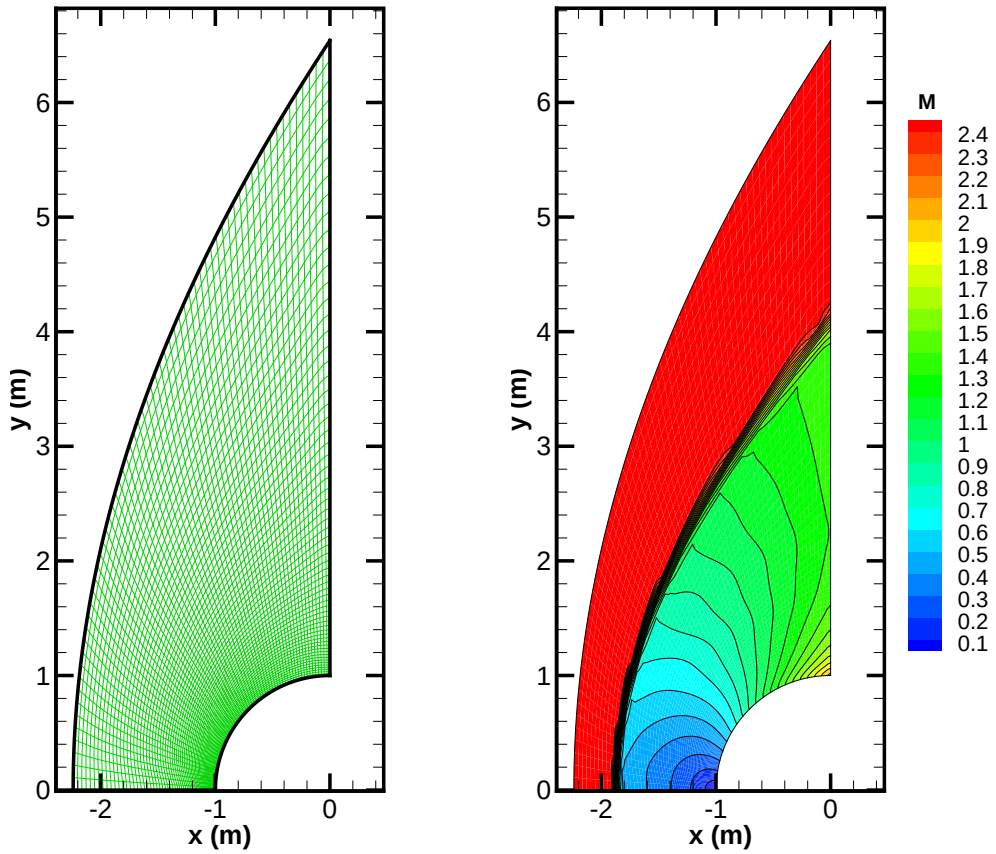
Figure 5.2: Comparison of 2D and 3D BDF2-NKS predicted solutions for a 1D shock tube problem at t=0.5 ms to the exact analytical solution.

5.1.2 Steady Two-Dimensional Supersonic Flow Past a Circular Cylinder

Verification of the inviscid spatial discretization procedure for the multi-dimensional case was evaluated using supersonic flow past a circular cylinder with a $r_c = 1$ m radius. The axis of symmetry for the cylinder was taken to be perpendicular to the free-stream flow direction and assumed to be infinitly long. The free-stream Mach number is $M_\infty = 2.5$ for the case of interest, such that a stationary bow shock forms about the body. The computational domain extends upstream of the cylinder approximately $1.25 \times r_c$ parallel to the flow and $5 \times r_c$ perpendicular to the flow, as shown in Figure 5.3(a), such that the resulting bow shock will be approximately centered in the domain. Dirichlet-type fixed free-stream flow conditions are applied on the upstream boundary of the domain with the face of the cylinder treated as a reflective boundary. Neumann-type boundary conditions are applied to the bottom and outlet boundaries.

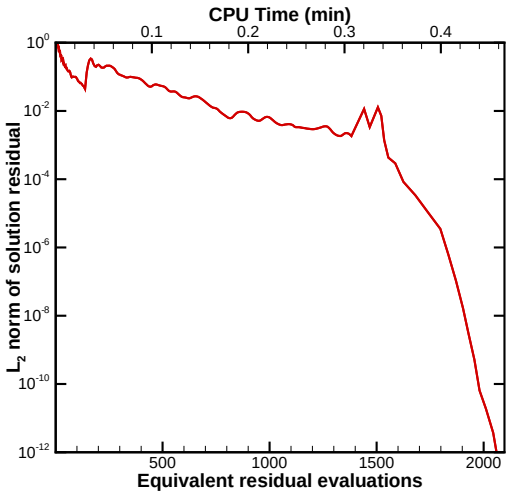
The numerical solution was calculated on a single block 64×64 mesh consisting of 4,096 computational cells as shown in Figure 5.3(a). The predicted solution was obtained with the two-dimensional version of the parallel implicit solver where the Roe approximate Riemann solver is used with the Venkatakrishnan slope limiter, a GMRES tolerance of 0.1, and ILU(2). In addition, the values of the slope limiters were "frozen" and held constant after the norm of the solution residual was reduced by four orders in magnitude. The limiter-freezing enables more rapid convergence of Newton's method in the presence of strong shocks.

The computed Mach number distribution for the $M_\infty = 2.5$ blunt-body flow is shown in Figure 5.3(b) and the convergence of the parallel NKS algorithm is given in Figure 5.3(c), where the 2-norm of density residual is depicted as a function of the number of equivalent residual evaluations and total processor time. The solution residual was converged 12 orders of magnitude in 22 Newton steps with 458 total GMRES iterations. The computed Mach number distributions demonstrate the prediction capabilities of the proposed algorithm for this class of flow problem. The structure and position of the bow shock are well resolved and the subsonic region in the vicinity of the stagnation point on the cylinder is accurately represented. Furthermore, rather rapid convergence of the solution is obtained even for this highly nonlinear problem with strong shocks.



(a) Solution Grid

(b) Mach Contour



(c) Convergence History

Figure 5.3: Numerical prediction of steady two-dimensional supersonic flow past a cylinder with a free-stream Mach number of $M_\infty = 2.5$ obtained using a (a) 64×64 grid showing (b) computed Mach number distribution and corresponding (c) convergence history of Newton scheme.

5.1.3 Steady Two-Dimensional Flow Over Bump in Channel

Subsonic Flow with Low-Mach-Number Preconditioning

The problem of inviscid flow through a two-dimensional channel with a non-smoothed bump, as described by Lynn [191], is used herein to show the benefits and improved accuracy provided by the low-Mach-number preconditioning. The bump flow problem consists of a rectangular shaped flow domain with a size of 5.5 m $\times$ 2.0 m, and with a circular arc as a bump on the bottom boundary. Dirichlet-type fixed upstream flow conditions are applied on the inlet boundary of the domain and the bottom and top edges are taken to be reflective boundaries. A Neumann-type boundary condition is applied on the outlet boundary. The initial computational mesh for this case consisted of 128 cells in the x-direction and 64 cells in the y-direction, and was subdivided into 8 blocks as shown in Figure 5.4. A fine solution mesh with 5 levels of mesh refinement and 94 blocks (94,208 cells) is shown in Figure 5.5 for comparison.

The initial condition for the problem was a uniform flow of air, the composition for which was as given in Section 5.1.1, with upstream Mach numbers of M_∞=0.01 and M_∞=0.2. The predicted solutions were obtained with the two-dimensional version of the parallel implicit implementation both with and without low-Mach-number preconditioning. In

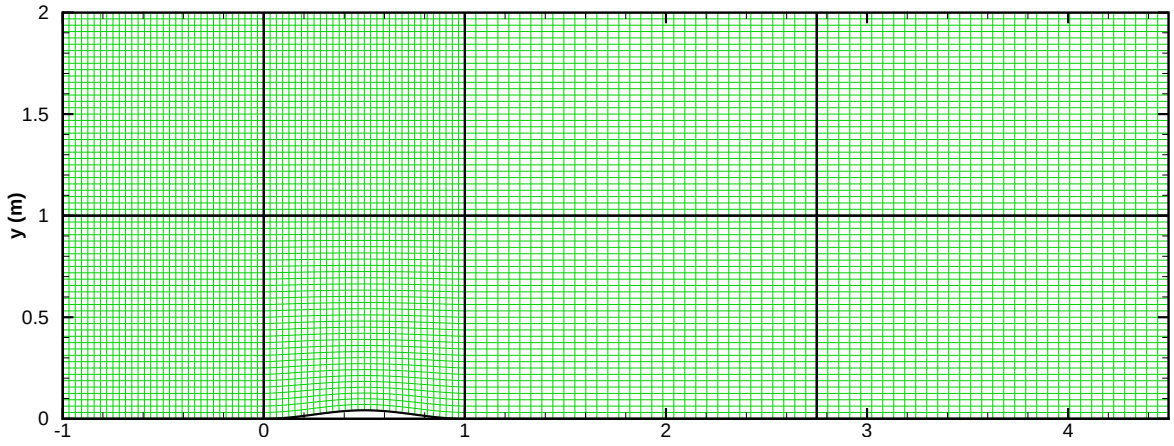


Figure 5.4: Initial 8 block, 128×64 computational grid for inviscid flow past a bump in a 2D channel.

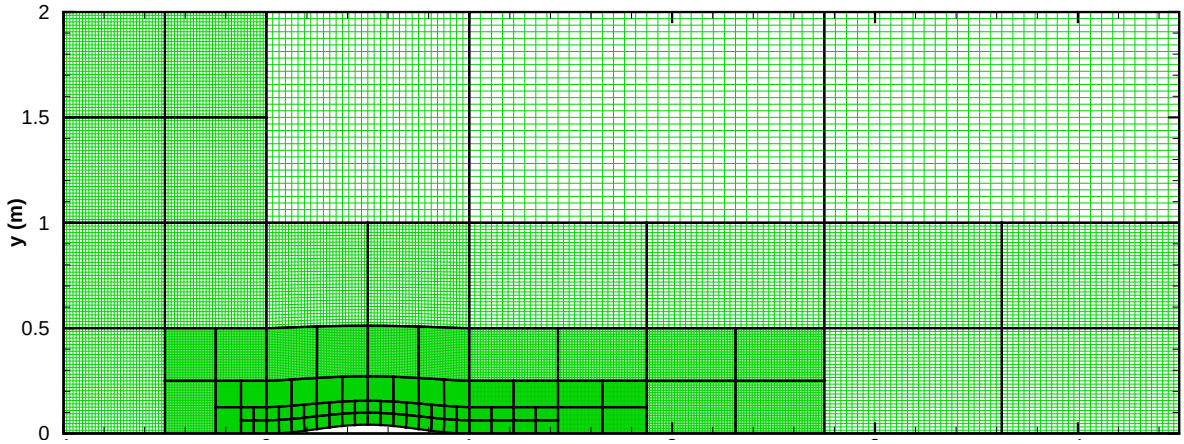
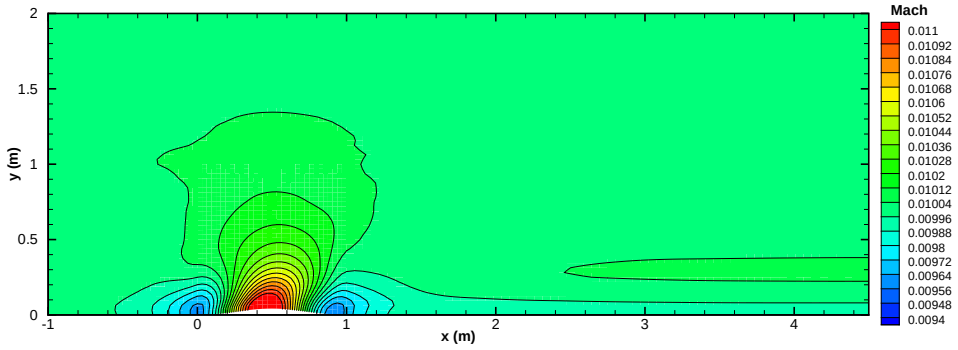


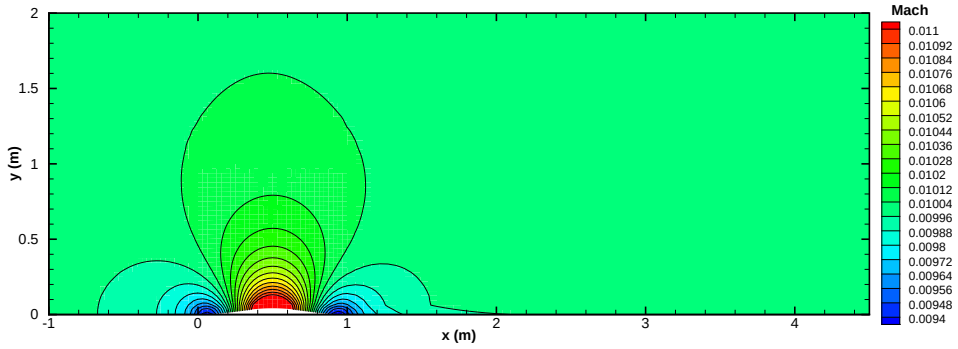
Figure 5.5: Final 94 block, 94,208 cell computational grid for inviscid flow past a bump in a 2D channel after 5 levels of mesh refinement.

the case of the latter, a minimum preconditioner Mach number equivalent to the upstream Mach number, $M_{r_{min}} = M_\infty$, was used. For both results, the Roe approximate Riemann solver was used with the Venkatakrishnan slope limiter, a GMRES tolerance of 0.1, and ILU(2). The slope limiter was frozen after three orders of magnitude reduction of the solution residual. Predicted distribution of the flow Mach number obtained by solving the Euler equations using the NKS algorithm with and without low-Mach-number preconditioning on the initial coarse grid are shown for M_∞=0.01 in Figures 5.6(b) and 5.6(a) and for the M_∞=0.2 case in Figures 5.7(b) and 5.7(a), respectively. A fine solution with 5 levels of mesh refinement with low-Mach-number preconditioning is shown in Figure 5.6(c) for M_∞=0.01 and in Figure 5.7(c) for M_∞=0.2 for comparison.

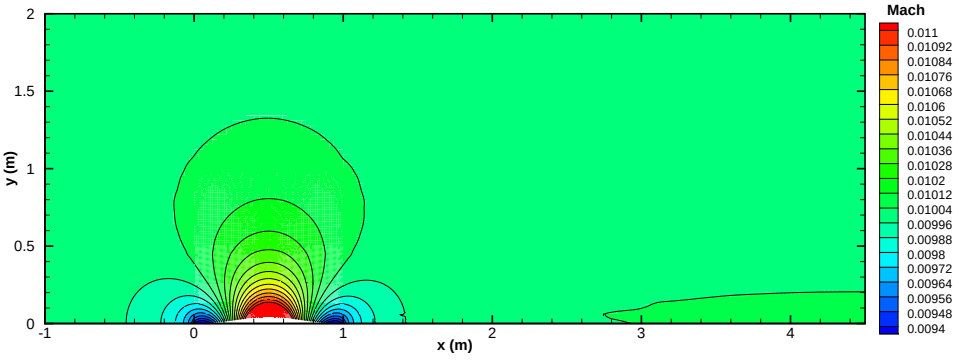
For low-speed subsonic flow over a symmetrical bump, it is expected that the resulting shapes of Mach contours should be self similar and symmetric with respect to the upstream and downstream edges of the bump even though the magnitude of the Mach number varies depending on the specified upstream value. This can be seen when comparing the two solutions in Figure 5.6(c) and Figure 5.7(c). This is a useful feature when investigating the accuracy of flow solutions and in this case the effects of the low-Mach-number preconditioning. The low-Mach-number M_∞=0.01 non-preconditioned coarse solution results in distorted Mach contours, due to excessive dissipation, whereas the



(a) Coarse Grid No Preconditioning

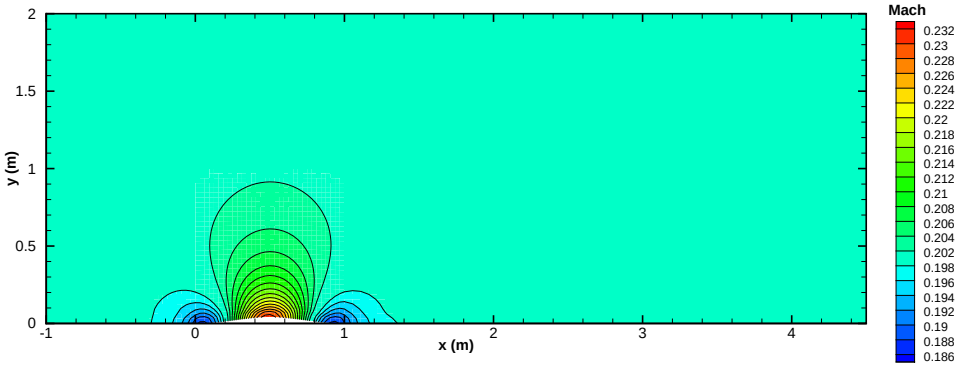


(b) Coarse Grid with Low-Mach-Number Preconditioning

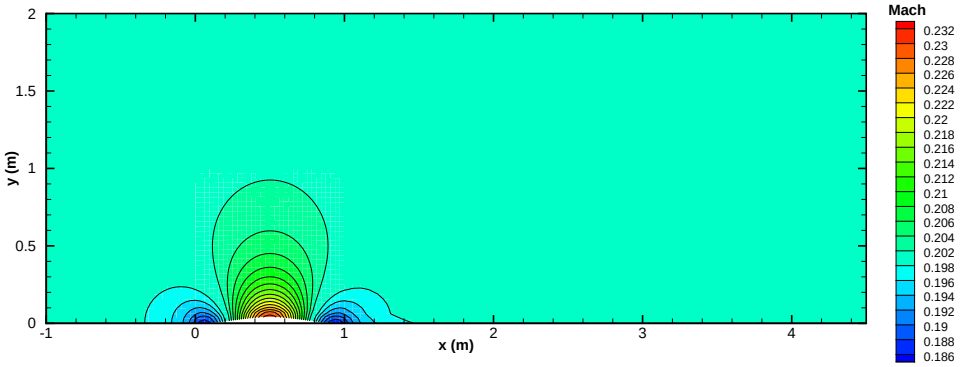


(c) Fine Grid with Low-Mach-Number Preconditioning

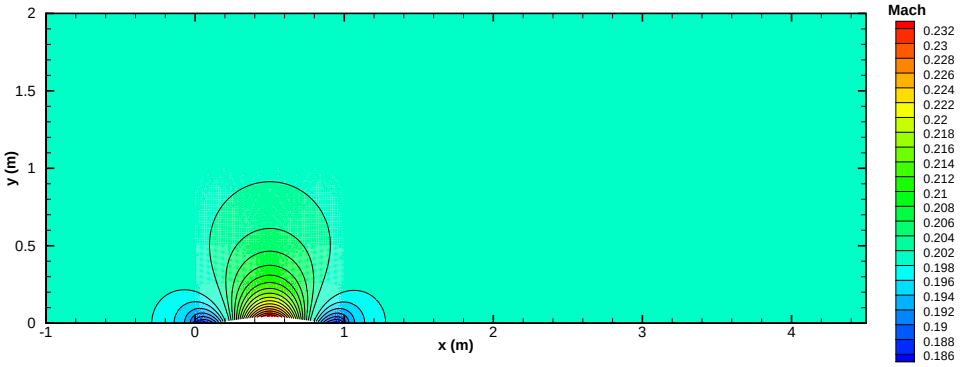
Figure 5.6: Numerical prediction of steady two-dimensional subsonic, M=0.01, flow past a a bump in a channel showing the Mach number distribution corresponding to calculations (a) with and (b) without low-Mach-number preconditioning on a coarse 8 block (8,192 cell) grid and on a (c) fine solution with 94 block (94,208 cell) grid with 5 levels of mesh adaptation.



(a) Coarse Grid No Preconditioning



(b) Coarse Grid Low-Mach-Number Preconditioning

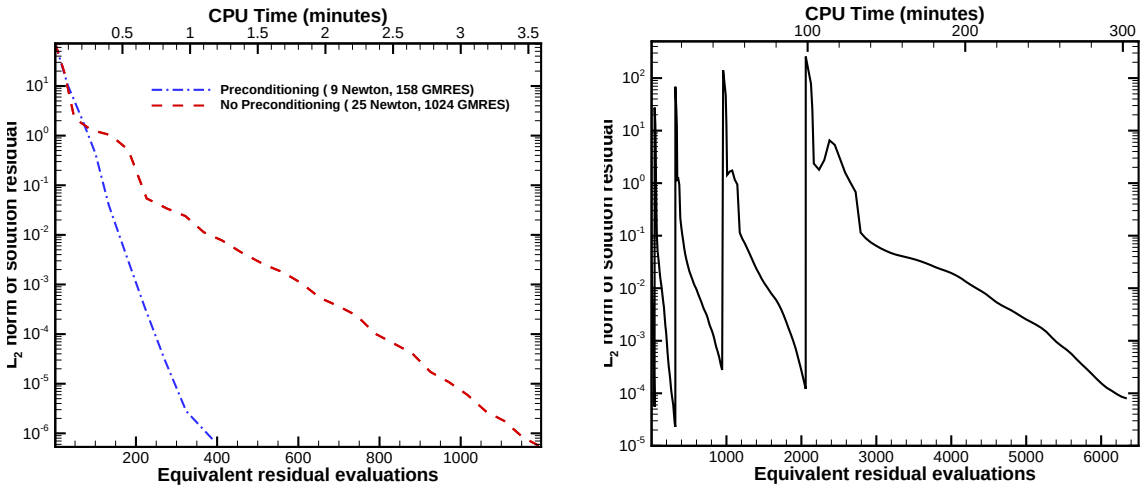


(c) Fine Grid with Low-Mach-Number Preconditioning

Figure 5.7: Numerical prediction of steady two-dimensional subsonic, M=0.2, flow past a bump in a channel showing the Mach number distribution corresponding to calculations (a) with and (b) without low-Mach-number preconditioning on a coarse 8 block (8,192 cell) grid and on a (c) fine solution with 94 block (94,208 cell) grid with 5 levels of mesh adaptation.

Mach contours in the coarse preconditioned case are less distorted and clearly have a much better agreement with that of the fine solution in Figure 5.6(c). For the M_∞=0.2 case, this distortion of the Mach number contours is significantly reduced for this higher speed case and the solutions with and without low-Mach-number preconditioning are almost identical as shown in Figures 5.7(a) and Figures 5.7(b), respectively. The results of Figures 5.6 and Figures 5.7 provide a clear indication of the necessity of low-Mach-number preconditioning for low-speed solutions of nearly incompressible flows when applying a compressible algorithm, at least for inviscid flow cases.

Along with more accurate solutions, the solution residual convergence history shown in Figure 5.8(a) shows that the preconditioned solution converges significantly faster than the non-preconditioned case. This is a result of the low-Mach-number preconditioner also serving as a matrix preconditioner reducing the stiffness of the governing equations, resulting in far fewer GMRES iterations to achieve convergence. The convergence history of the solution residual on the sequence of refined AMR meshes leading to the fine grid, with low-Mach-number preconditioning applied, is also provided in Figure 5.8(b) for comparison.



(a) Coarse Grid Convergence

(b) Fine Grid Convergence

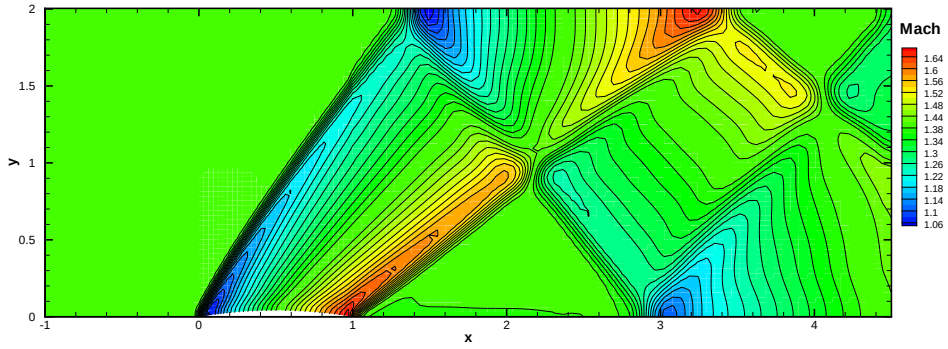
Figure 5.8: Convergence histories of the Newton scheme for the numerical prediction of steady two-dimensional subsonic flow past a a bump in a channel for solutions obtained on a (a) coarse 8 block (8,192 cell) grid with and without low-Mach-number preconditioning and (b) fine 94 block (94,208 cell) grid with low-Mach-number preconditioning.

Supersonic M=1.4 Flow

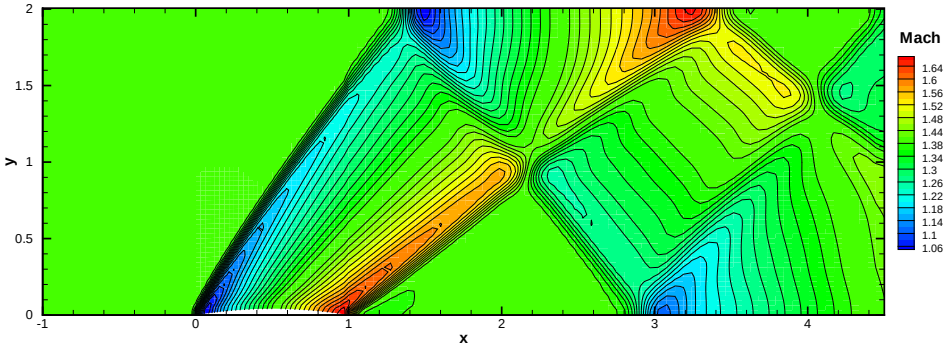
To verify further that the two- and three- dimensional implementations of the proposed solution method are producing equivalent results, and illustrate the AMR capabilities of the proposed solution method, the flow over a bump in channel case is again used, however the upstream flow is now taken to be supersonic at the channel inlet. This results in a complex shock structure propagating downstream from the bump. The same initial and boundary conditions are used as in the previous case; however, the inlet upstream Mach number in this case was specified to be M=1.4 and as such no low-Mach-number preconditioning was required. For 2D simulations, the same 8 block initial grid shown in Figure 5.4 was used. For the 3D computations, each block of the 2D grid was extruded in the z-direction by 2 cells to create the 3D mesh. In both simulations, the Roe approximate Riemann solver was used with the Venkatakrishnan slope limiter, a GMRES tolerance of 0.1, and ILU(2). The slope limiter was frozen after three orders of magnitude reduction of the solution residual.

The predicted Mach contour distributions obtained using the two- and three-dimensional versions of the parallel implicit finite-volume scheme are compared in Figures 5.9(a) and 5.9(b). As expected, both versions produce virtually identical results. While there are minor differences, the computed solutions appear to be very similar. The convergence plots for the 2D and 3D simulations of Figures 5.9(c) and 5.9(d) also show the number of equivalent residual evaluations required for solution convergence are approximately the same; however, the required CPU time, depicted on the top x-axis, is greater for the three-dimensional solver due to the extra overhead of solving the problem in three space dimensions with twice the number of grid cells. In both cases, rather rapid convergence of the solution is obtained after the initial startup phase reduces the residual by about two orders of magnitude.

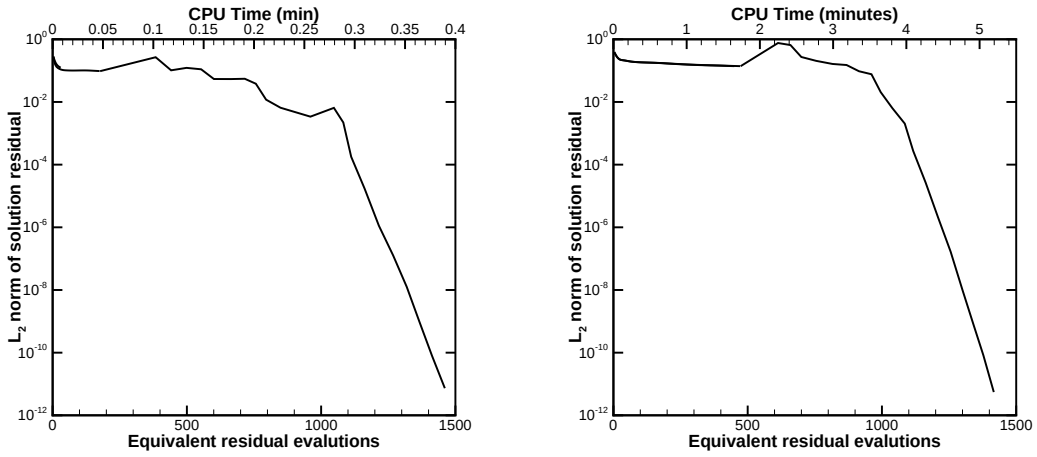
The resulting complex shock structure of the channel flow in this case provides a nice opportunity to demonstrate the predictive and resolution capabilities of the block-based AMR scheme and how it may be used to greatly reduce the number of computational cells required to solve a problem. Figure 5.10 shows the predicted distribution of the Mach number and block-boundaries of the adapted mesh for a sequence of two-dimensional steady solutions obtained using the proposed parallel AMR finite-volume scheme on six consecutively refined AMR meshes carried out based on the gradient of density. The



(a)



(b)



(c) (d)

Figure 5.9: Numerical predictions of steady supersonic flow past a bump in a channel with an inlet Mach number of M = 1.4 showing the Mach number distributions for a (a) 8 blocks of (32×32 cells) solution in two-dimensions and (b) 8 blocks of (32×32×2 cells) in three-dimensions along with their corresponding convergence histories for (c) two- and (d) three-dimension solution schemes.

predicted Mach number at $y = 0.8$ m in Figure 5.11 shows that with each level of mesh refinement the solution is improved to a point where by the 6th level the solution is essentially mesh independent with accurate representation of the steady-state shock structure. The solution residual convergence history of the parallel NKS scheme showing each of the 6 levels of mesh refinement is shown in Figure 5.12. At each level of refinement, the convergence history shows similar trends, whereby an initial startup phase, leads into a rapid convergence rate after the limiter-freezing is enabled.

A measure of the efficiency of the block-based AMR scheme for this problem can be defined by a refinement efficiency parameter, η, given by

$$\eta = 1 - \frac{N_{\text{cells}}}{N_{\text{uniform}}}, \tag{5.1}$$

where N_{cells} is the actual number of cells in the mesh and N_{uniform} is the total number of cells that would have been used on a uniform mesh composed of solution blocks all at the finest level. The efficiency of the AMR scheme is $\eta = 0$ for the initial mesh, where all solution blocks at the same level of refinement, but rapidly improves as the number of refinement levels increases. In this case the final grid has only 113,408 cells, whereas without refinement a uniform grid of 16,777,216 would be required to achieve the same solution resolution. That is a refinement efficiency of 99.32% or put another way less than 1% of the computational cells are required using mesh refinement to achieve the same solution resolution and accuracy. This translates into approximately a 100 fold reduction in solution time and memory requirements based on mesh size alone.

5.2 Viscous Flow

5.2.1 Subsonic Laminar Boundary-Layer Flow Past a Flat Plate

The computation of two-dimensional subsonic laminar flow over a flat plate is now considered to demonstrate the accuracy of the viscous spatial discretization scheme implemented within the parallel implicit algorithm for reactive flows. The initial condition for the flat plate flow problem was a uniform flow of air, the composition for which was as given previously in Section 5.1.1, with a free-stream Mach number of $M_\infty = 0.2$, and a Reynolds number of Re=9,318 based on a 0.002 m plate length.

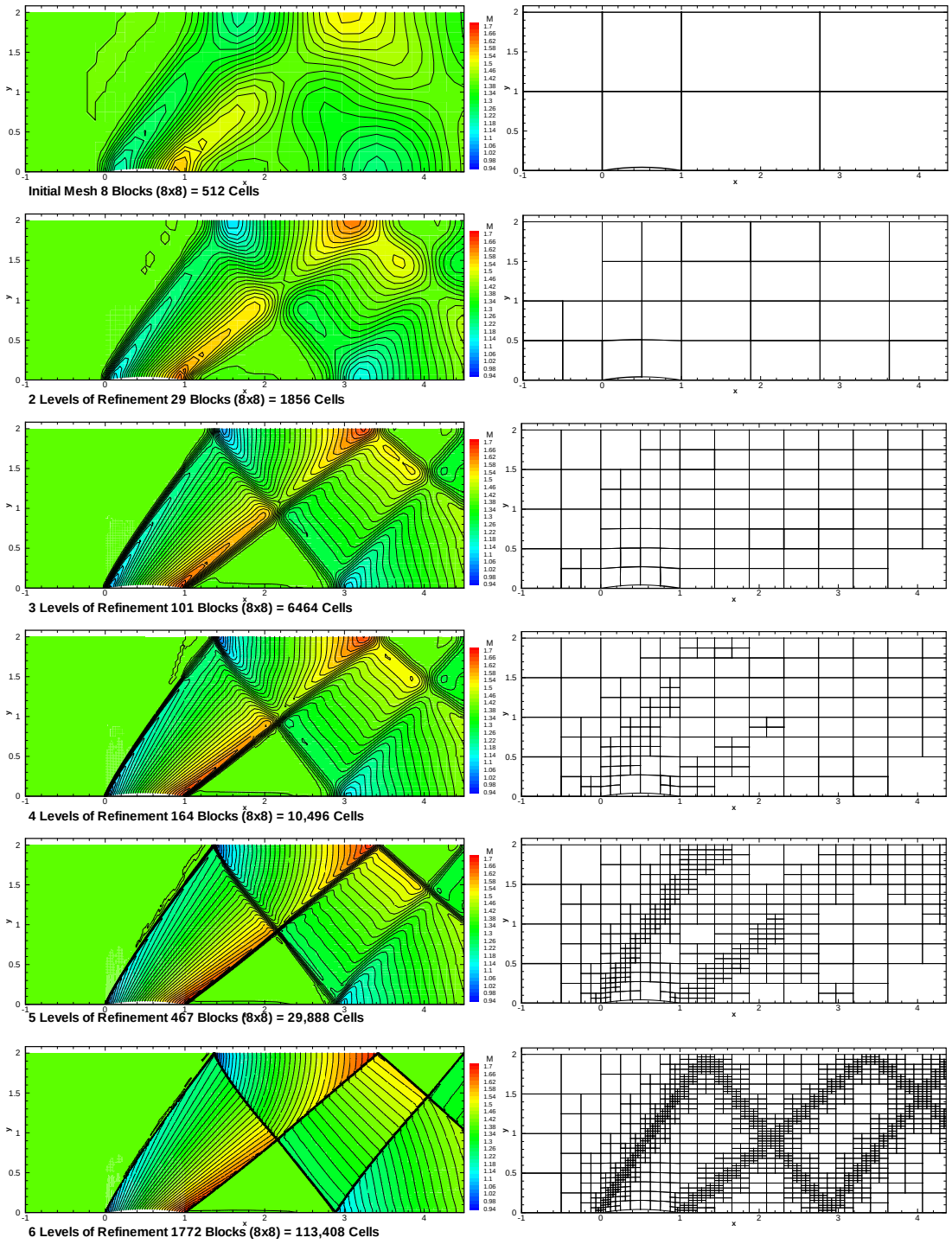


Figure 5.10: Numerical prediction of two-dimensional steady supersonic flow past a bump in a channel with an inlet Mach number of M=1.4 showing the Mach number distributions and block boundaries at each of the 6 levels of adaptive mesh refinement based on the gradient of density.

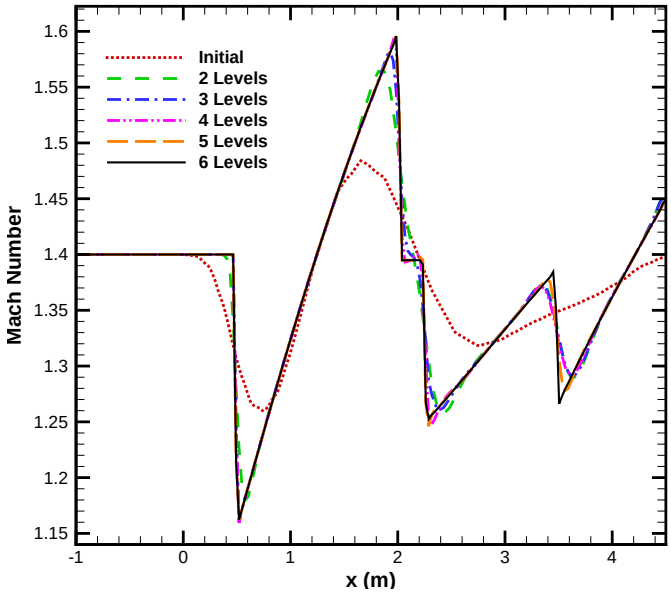


Figure 5.11: Numerical prediction of two-dimensional steady supersonic flow past a bump in a channel with an inlet Mach number of M=1.4 showing the Mach number distribution at $y = 0.8$ m at each of the 6 levels of adaptive mesh refinement.

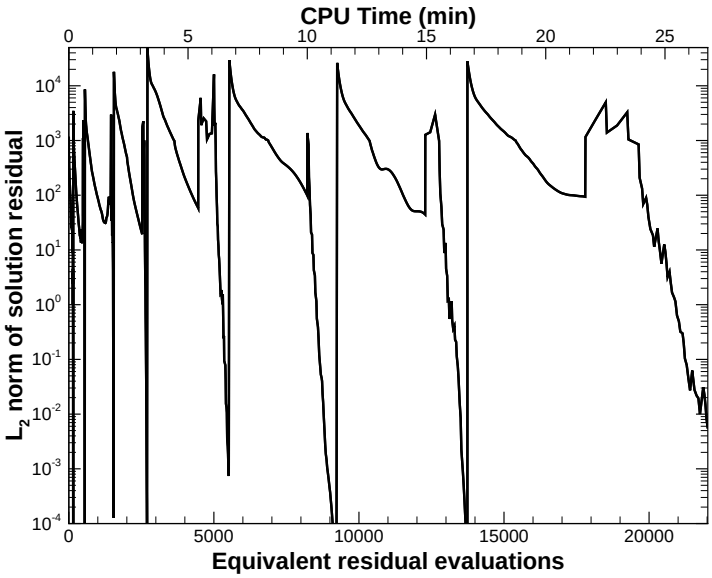


Figure 5.12: Numerical prediction of steady supersonic flow past a bump in a channel with an inlet Mach number of M=1.4 showing the convergence history of the Newton scheme through 6 levels of adaptive mesh refinement.

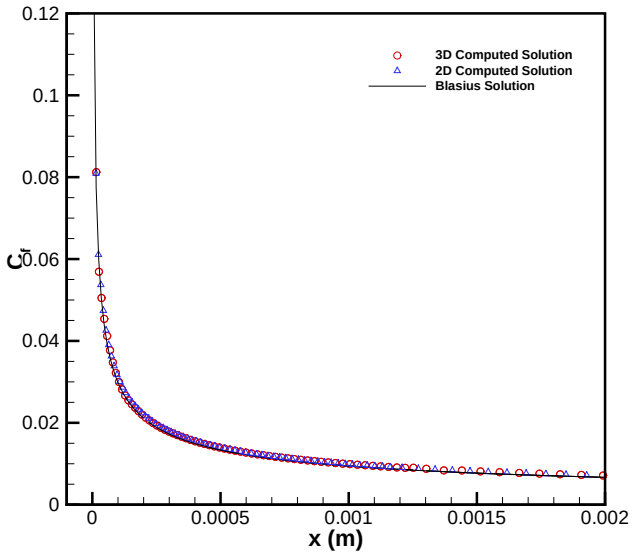
The predicted solution was again obtained using both the two and three-dimensional parallel implicit solution methods and compared with the exact solution of the incompressible boundary layer equations first obtained by Blasius as presented in the textbook by Schlichting [192]. For the 2D case, the initial mesh consisted of three blocks of 32 $\times$ 64 cells and was uni-directionally stretched towards the plate. In three space dimensions, the same 2D mesh was extruded in the z-direction producing a 3D mesh consisting of three blocks of 32 $\times$ 64 $\times$ 2 cells.

The Roe flux function with the Venkatakrishnan limiter was used together with the with a GMRES tolerance of 0.1 and ILU(2) fill level to achieve a steady-state solution. Three levels of adaption based on the gradient of density were applied resulting in final meshes of 14 blocks with 28,672 cells for the 2D simulation and 31 blocks with 126,976 cells for the 3D computations. The predicted skin-friction coefficient along the plate are shown in Figure 5.13(a). It can be seen that the velocity components and the skin friction coefficient are in excellent agreement with the Blasius solution, providing a good indication of the validity of the spatial discretization procedure for laminar flows.

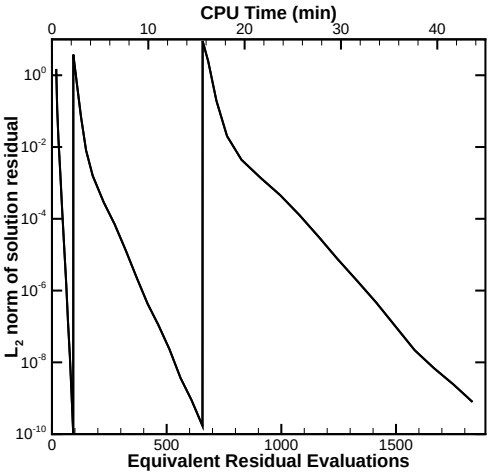
The convergence histories for both 2D and 3D simulations showing solution convergence at each of the three mesh refinements is shown in Figures 5.13(b) and 5.13(c), respectively. This problem converges very rapidly requiring effectively no startup, even on the fine meshes with grid adaption. The number of equivalent residual evaluations required for solution convergence are approximately the same. However, the required CPU time, depicted on the top x-axis, is greater for the three-dimensional solver due to the extra overhead of solving the problem in three space dimensions with over four times the number of grid cells due to differences in the two and three-dimensional mesh adaption.

5.3 One-Dimensional Planar Unstrained Laminar Premixed Flame

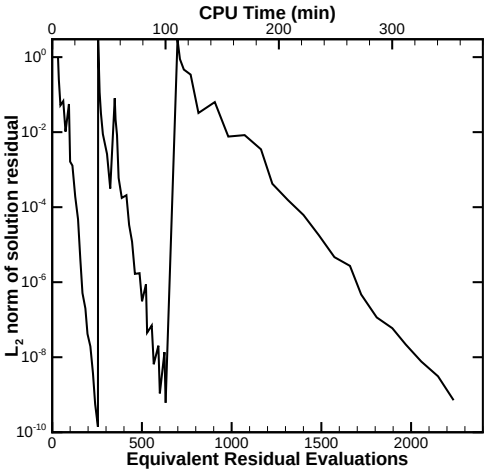
The implementation of the chemical kinetics and thermodynamic and transport models used herein for describing methane-air combustion are verified by performing laminar flame speed and flame structure computations for a one-dimensional, planar steady unstrained laminar flame. Similar one-dimensional premixed flame calculations were



(a) Skin friction



(b) 2D convergence history



(c) 3D convergence history

Figure 5.13: Numerical prediction of subsonic laminar flow, M=0.2 and Re=9,318, over a 0.002 m flat plate (a) computed plate skin friction, C_f, compared with Blasius solution, with the (b) two- and (c) three-dimensional parallel implicit solution methods convergence histories.

performed in the previous work of Northrup [62] using a two-dimensional explicit time-marching algorithm for 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 [193]. 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.

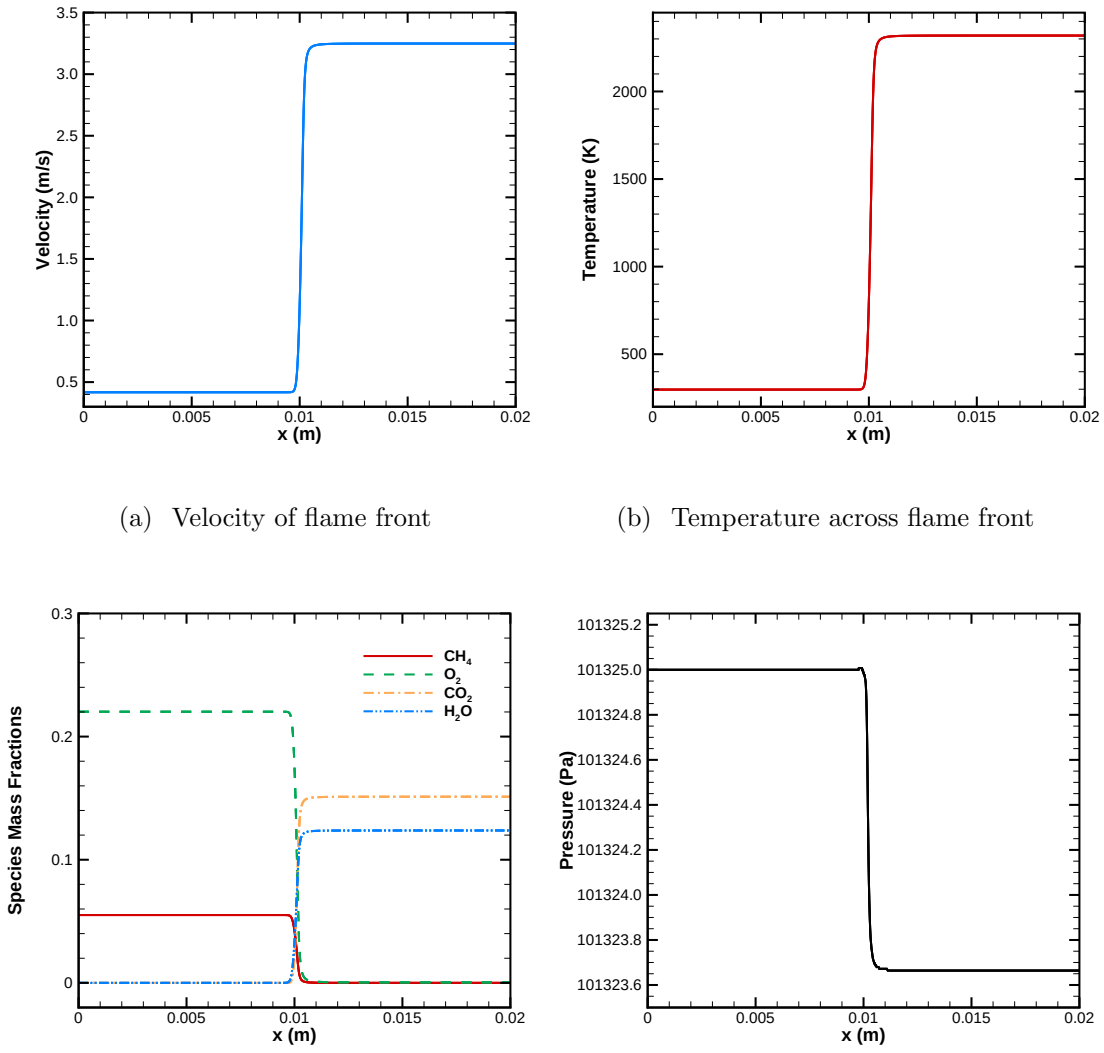
In this work, the same one-dimensional premixed flame was re-computed using both the two- and three-dimensional implementations of the NKS solution algorithm. In the 2D case, the computational domain consisted of 100 $\times$ 2 cells on a solution domain of size 0.02 m $\times$ 0.0002 m with a mesh clustered near the flame front located at the center of the domain. For the 3D computations, the 2D mesh was extruded in the z direction by 0.0002 m producing a computational domain that consisted of 100 $\times$ 2 $\times$ 2 cells. 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 [62] for further details of the boundary conditions for a stationary premixed flame).

In the simulations, the Roe flux function with the Venkatakrishnan limiter was used together with the Newton-Krylov algorithm with a GMRES tolerance of 0.01 and ILU(2) to achieve a steady-state solution for both two- and three-dimensional implementations. No Schwarz preconditioning was used as the domain consists of only one solution block and the problem was solved serially.

As should be expected, the predicted solutions of the two- and three-dimensional NKS implementations, as depicted in Figure 5.14, were identical and matched well with the previous results of Northrup [62]. The numerical predictions of the laminar flame structure is well represented by the profiles of the velocity, temperature, and mass fraction of Figures 5.14(a),5.14(b),5.14(c) which show the variation of these quantities through the flame. For the five-species, one-step model, the predicted laminar flame speed was about 45 cm/s and the flame temperature was 2,300 K. Both values are slightly higher

than the laminar flame speed and flame temperature provided by CHEMKIN, which were 41 cm/s and 2,234 K, respectively. Nevertheless, in general the predictions of the one-step model are thought to be acceptable considering the simplified nature of the one-step chemical kinetics. The only significant difference between the two- and three-dimensional implementations was in the convergence histories shown in Figures 5.16(a) and 5.16(b), respectively. While the equivalent residual evaluations are fairly consistent between the two implementations, the CPU time for the 3D case is as expected significantly greater than that for the two dimensional case, as found previously in the inviscid and viscous non-reactive flow results.

Low-Mach-number preconditioning with $M_{r_{min}} = 0.01$ was employed when obtaining the low-speed laminar premixed flame results of Figure 5.14, both for convergence acceleration and to avoid introducing excessive numerical dissipation. The necessity of the low-Mach-number preconditioning is easily demonstrated if one compares the accuracy of the solution profiles calculated with and without low-Mach-number preconditioning as given in Figure 5.15. The prominent effects of improved solution accuracy afforded by the preconditioning are seen in the species mass concentrations, Figure 5.15(c), and the resolution of the pressure drop across the flame front, Figure 5.15(d). Without preconditioning it is not possible to obtain the correct pressure drop across the flame front without resorting to excessive mesh resolution. Also in comparing the convergence histories of solutions with and without low-Mach-number preconditioning as given in Figure 5.16(c), it is evident that the low-Mach-number preconditioning significantly aids in reducing the solution convergence time similar to what was observed for the inviscid subsonic flow over a bump as discussed previously in Section 5.1.3.



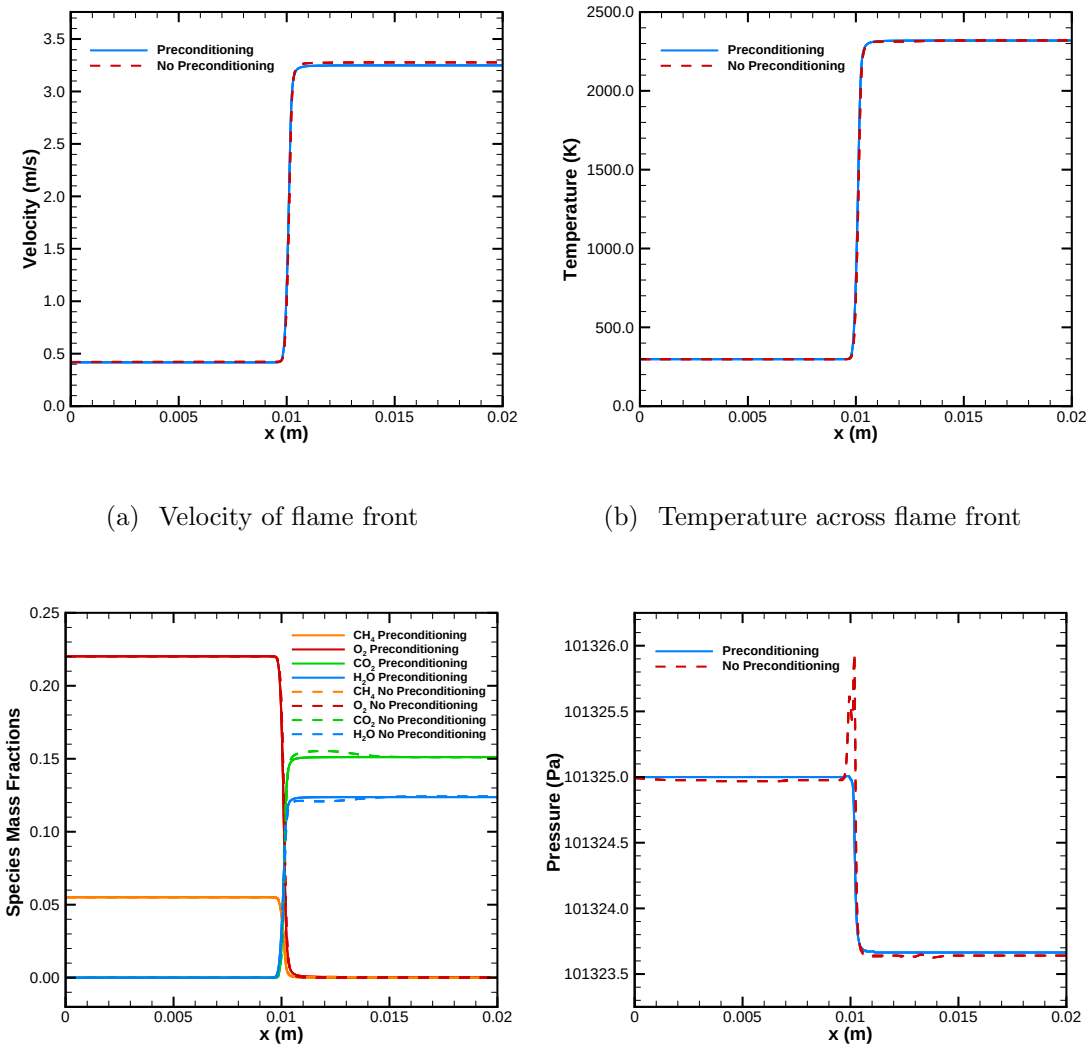
(a) Velocity of flame front

(b) Temperature across flame front

(c) Species mass fractions

(d) Pressure drop across flame front

Figure 5.14: Steady one-dimensional planar laminar premixed methane-air flame with an equivalence ratio, $\phi = 1$, predicted solution profiles of (a) velocity, (b) temperature, (c) species mass fractions, and (d) pressure across the flame front.



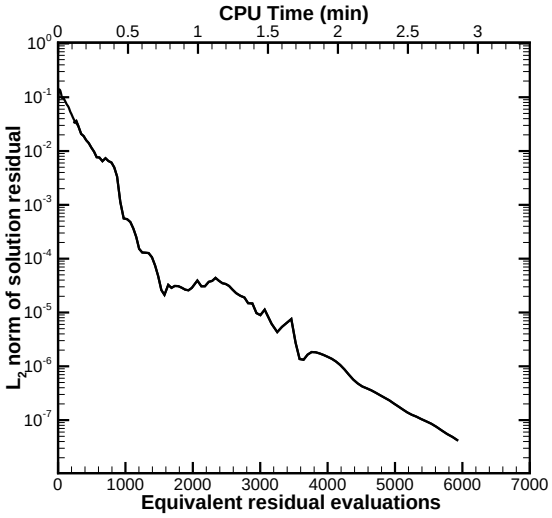
(a) Velocity of flame front

(b) Temperature across flame front

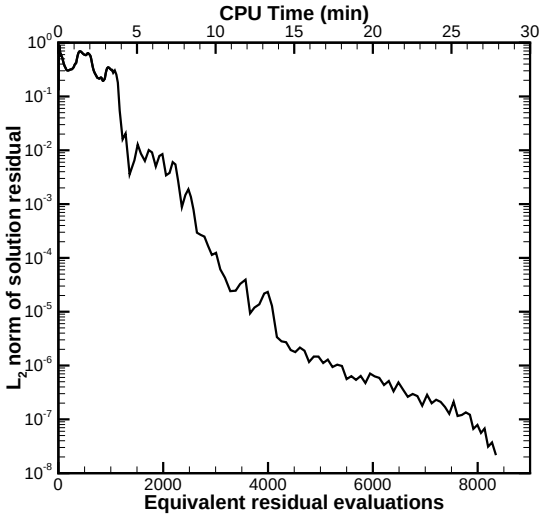
(c) Species mass fractions

(d) Pressure drop across flame front

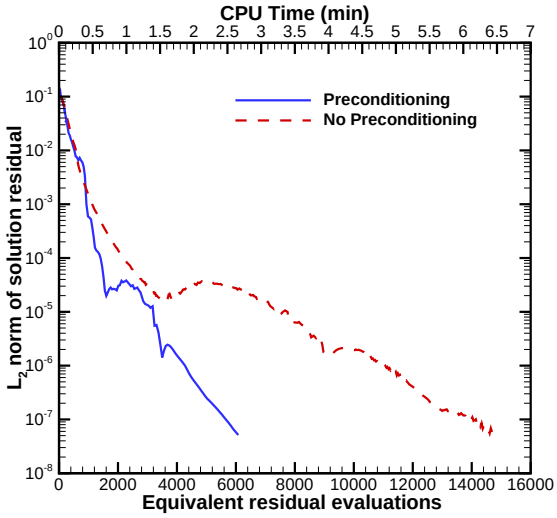
Figure 5.15: Steady one-dimensional planar laminar premixed methane-air flame with an equivalence ratio, $\phi = 1$, comparison of predicted solution profiles with and without low-Mach-number preconditioning of (a) velocity, (b) temperature, (c) species mass fractions, and (d) pressure across the flame front.



(a) 2D convergence history



(b) 3D convergence history



(c) 2D convergence comparison

Figure 5.16: Steady one-dimensional planar laminar premixed methane-air flame with an equivalence ratio, $\phi = 1$, convergence histories for (a) two- and (b) three-dimensional NKS implementations with low-Mach-number preconditioning and (c) comparison without low-Mach-number preconditioning.

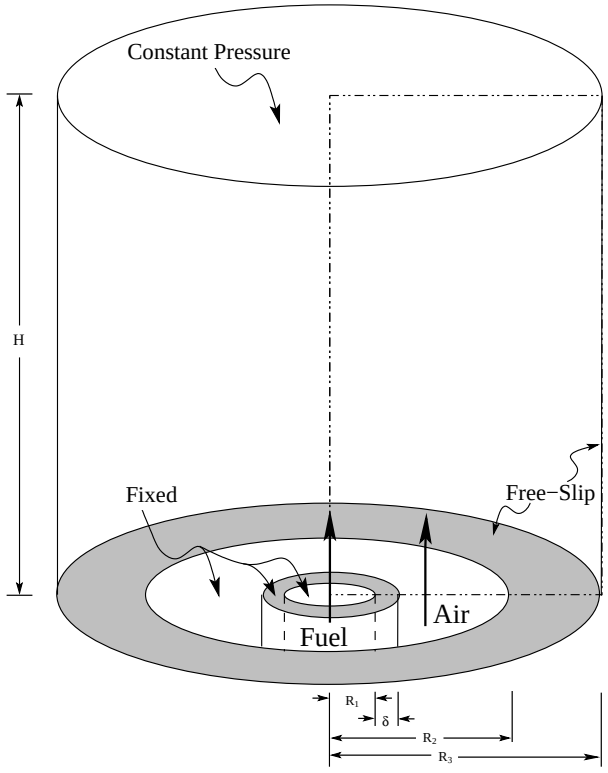


Figure 5.17: Schematic of Laminar Diffusion flame setup

5.4 Co-Flow Laminar Diffusion Flames

The application of the proposed parallel implicit AMR algorithm is now considered for the solution of a methane-air co-flow laminar diffusion flame. Such flames are well characterized and studied both experimentally and computationally, providing a very good basis for the evaluation of solution methods for laminar reactive flow. They also are typically axisymmetric in nature allowing validation of both the 2D axisymmetric and 3D implementations of the proposed parallel implicit AMR finite-volume scheme. In particular, both implementations should produce the same cross-sectional results and this will be demonstrated as part of the verification process.

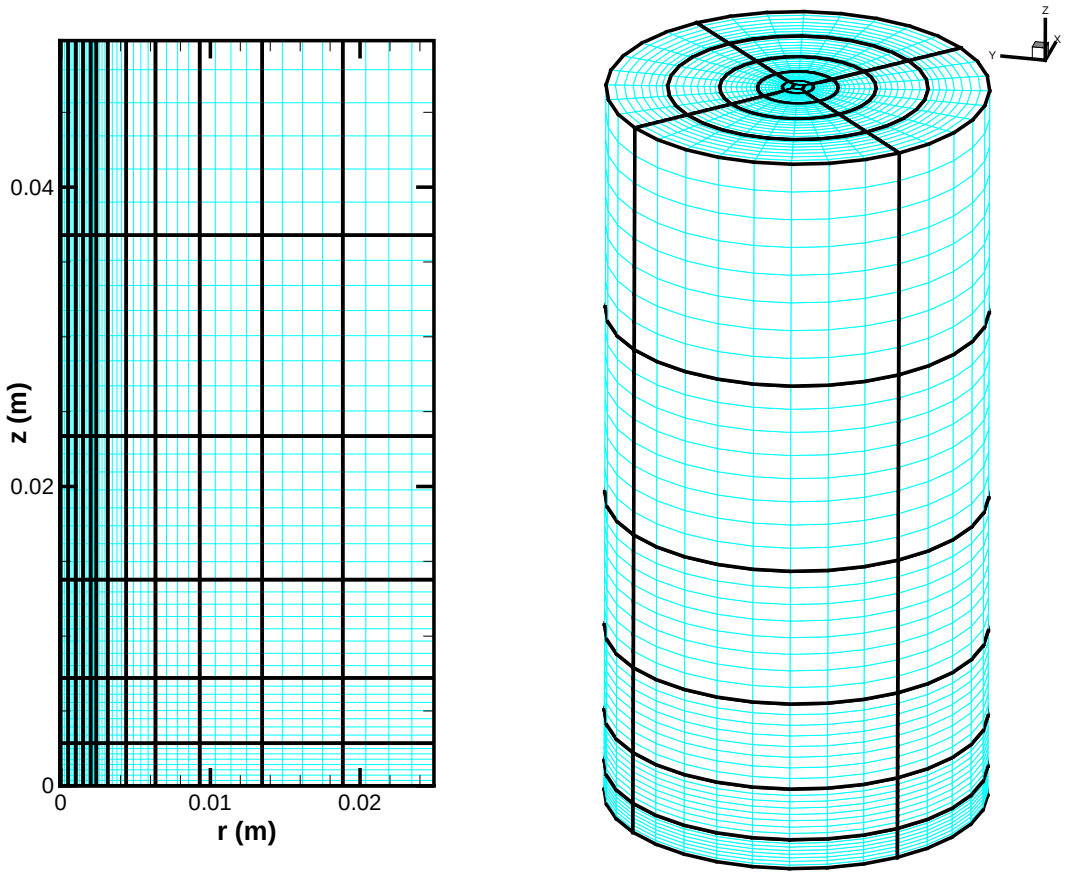
The particular laminar flame of interest here is a methane-air co-flow configuration as described by Mohammed *et al.* [194], Day and Bell [53], Dworkin *et al.* [11] and Dobbins *et al.* [116]. The flame boundary and initial conditions are the same as those used in the previous studies and are depicted in the schematic diagram of Figure 5.17.

5.4.1 2D and 3D Computational Domains

For the fully three-dimensional simulation, the computational domain is cylindrical in shape and 10 cm high with a radius of 5 cm. In the two-dimensional case, an axisymmetric coordinate frame is used with the axis of symmetry aligned with the centerline of the cylinder, in cross-section the dimensions of the axisymmetric domain is 10 cm by 5 cm, depicted by the dash-dot line in Figure 5.17. The outside of the cylinder (right hand edge in the 2D case) is taken to be a free-slip boundary along which inviscid reflection boundary data is specified. In both the 2D and 3D domains, the top or outlet of the flow domain is open to a stagnant reservoir at atmospheric pressure and temperature conditions and Neumann-type boundary conditions are applied to all properties except pressure which is held fixed or constant. In both 2D and 3D, the bottom or inlet is subdivided into four regions. The inner-most region ($r \leq 2$ mm) defines the fuel inlet or jet, which injects a nitrogen diluted methane fuel mixture ($c_{\mathrm{CH_4}} = 0.5149$, $c_{\mathrm{N_2}} = 0.4851$, $c_{\mathrm{O_2}} = 0$, $c_{\mathrm{CO_2}} = 0$, $c_{\mathrm{CO}} = 0$, and $c_{\mathrm{H_2O}} = 0$) at 298 K with a parabolic axial velocity profile of 0.7 m/s. The next region ($r = 2$ to $r = 2.38$ mm) represents the small gap associated with the annular wall separating the fuel and oxidizer. The third region ($r = 2.38$ to $r = 25$ mm) contains the inflowing stream of the co-flowing oxidizer with a uniform axial flow velocity of 0.35 m/s, in this case air at 298 K ($c_{\mathrm{O_2}} = 0.232$ and $c_{\mathrm{N_2}} = 0.768$). The final outer region of the lower boundary inlet ($r = 25$ to $r = 50$ mm) is specified to be a far-field boundary along which free-slip boundary conditions are applied such that there is no co-flow.

5.4.2 Steady Flame with Constant Inlet Fuel Mass Flow Rate

In the absence of any external perturbations, laminar diffusion flames are generally more stable than their premixed counter parts and more readily exhibit steady-state or time-invariant behaviour. As such a steady-state methane-air laminar flame solution was first considered. The solution domain was initialized with a uniform quiescent gas mixture corresponding to the four inlet regions defined previously, i.e. diluted methane-air fuel mixture in the innermost region and air in the outer three regions. The temperature of the gas mixture for a thin region across the fuel and oxidizer inlets is taken to be 1,500 K so as to ignite the flame. Note that the Mach and Reynolds numbers based on the fixed

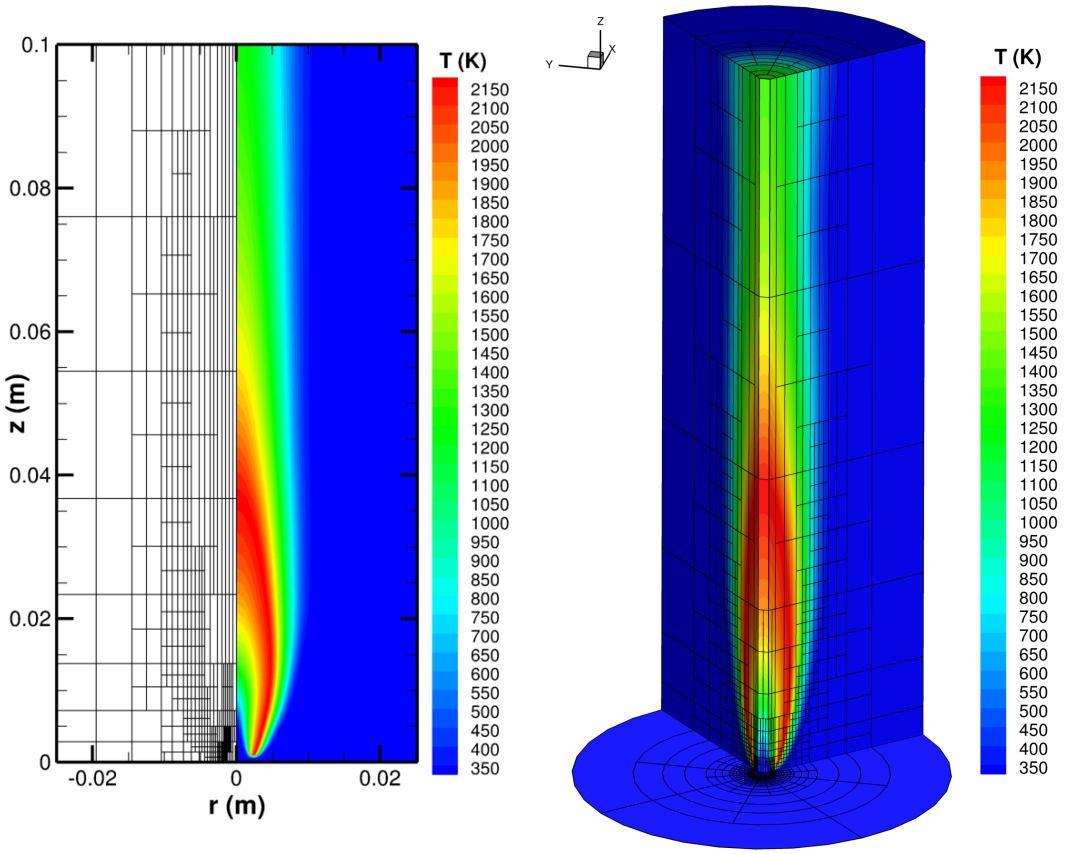


(a) Initial Coarse 2D Axisymmetric Mesh (b) Initial Coarse 3D Mesh

Figure 5.18: Initial coarse meshes: (a) 2D axisymmetric 96 block (4×8 cell) and (b) 3D 126 block (8×8×8) mesh, for methane-air co-flow laminar diffusion flame simulations.

diluted methane-air flow in the fuel inlet are M=0.0016 and Re=169. Additional details concerning the setup for this diffusion flame can be found in the papers by Mohammed *et al.* [194] and Day and Bell [53].

For both the 2D and 3D diffusion flame calculations, the Roe flux function with the Venkatakrishnan limiter was used with low-Mach-number preconditioning with a $M_{r_{min}}$ of 0.1. The Newton-Krylov-Schwarz algorithm with a GMRES tolerance of 0.01 and ILU(2) fill level for the 2D computation and ILU(0) for the 3D simulation was used to achieve steady-state solution results. For the 2D case, the rectangular domain was initialized with a coarse grid consisting of 96 blocks of 4×8 cells as shown in Figure 5.18(a), and for the 3D simulation a cylindrical domain was initialized with a hexahedral non-symmetric

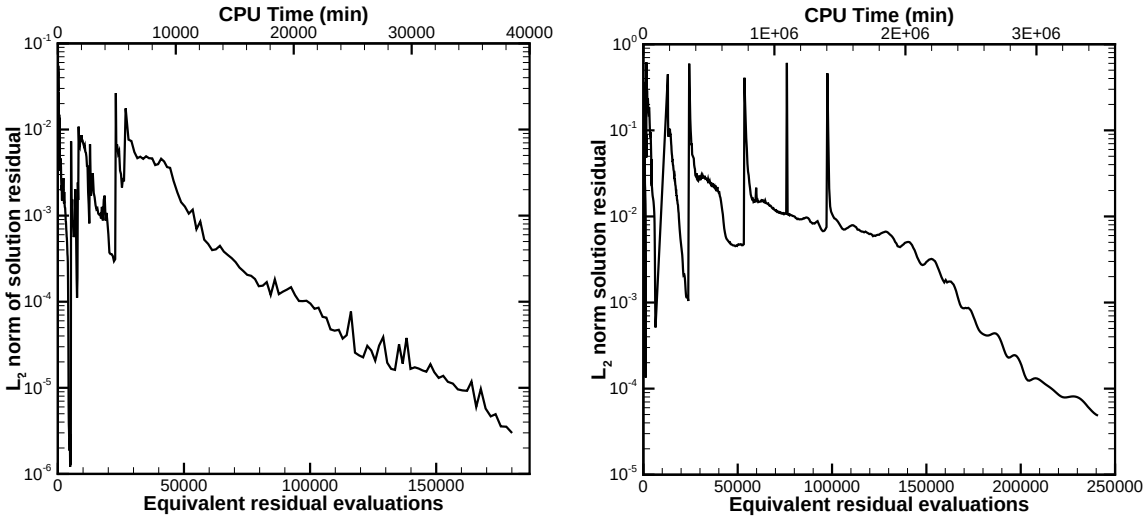


(a) 2D Axisymmetric Solution (b) 3D Solution

Figure 5.19: Solution of methane-air diffusion flame showing the computed isotherms, flame structure, and block boundaries obtained for (a) 2D axisymmetric 1392 (4×8) block mesh with 44,544 cells and five levels of refinement and (b) 3D 9275 (8×8×8) blocks with 4,748,800 cells with four levels of refinement (quarter section shown).

coarse grid consisting of 126 blocks of 8×8×8 cells as depicted in Figure 5.18(b). Coarse solutions were obtained on these initial meshes and then mesh refinement based on the gradients of temperature and mass fraction of CO_2 was applied.

The computed isotherms, flame structure, and block boundaries on the final refined meshes are shown for the 2D case after 5 levels of adaption resulting in a 1,392 block mesh with 44,544 cells with a refinement efficiency of η=97% in Figure 5.19(a) and, in the 3D case, after 4 levels of adaption resulting in a 9,275 block mesh with 4,748,800 cells with a refinement efficiency of η=85% in Figure 5.19(b). The solution convergence rates for the 2D and 3D steady-state calculations are shown in Figures 5.20(a) and 5.20(b), respectively. Both the 2D and 3D convergence histories are similar in terms of number



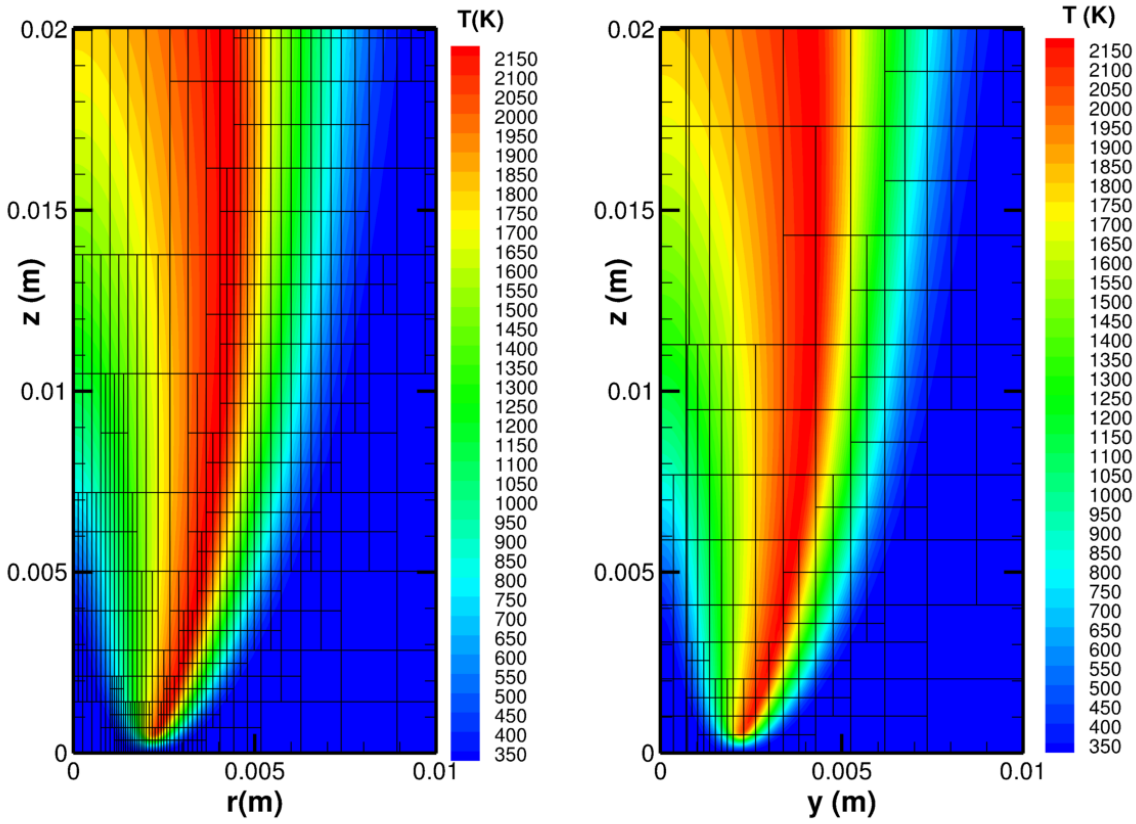
(a) 2D with AMR

(b) 3D with AMR

Figure 5.20: Newton-Krylov-Schwarz solution convergence histories for adapted (a) 2D 1,392 block and (b) 3D 9,275 block meshes for the solution of a co-flow laminar methane-air diffusion flame.

of equivalent residual evaluations, however as expected the computational time for the 3D solution computational time is significantly greater than the 2D solution, due to the larger grid size associated with the fully 3D solution.

A side by side comparison of the temperature isotherms for a cross-section (r–z plane) at x=0 of the predicted 3D solution and the predicted 2D axisymmetric solution are provided in Figures 5.21(a) and 5.21(b), respectively. The block boundaries are shown in the figure to highlight that the computational meshes are somewhat different, and yet both implementations of the proposed finite-volume scheme converge to what appear to be identical results. The temperature and species mass fractions for both solutions agree very well, as is shown in the extracted radial profiles the 2D and 3D solutions at the axial location of z=0.01 m as given in Figures 5.22(a) and 5.22(b). In particular, such close agreement provides strong support for the valid and correct implementation of the proposed 3D finite-volume solution method. The 3D implementation is clearly capable of providing accurate and grid independent solutions of the present axisymmetric co-flow diffusion flame that are in excellent agreement with the results of the 2D axisymmetric



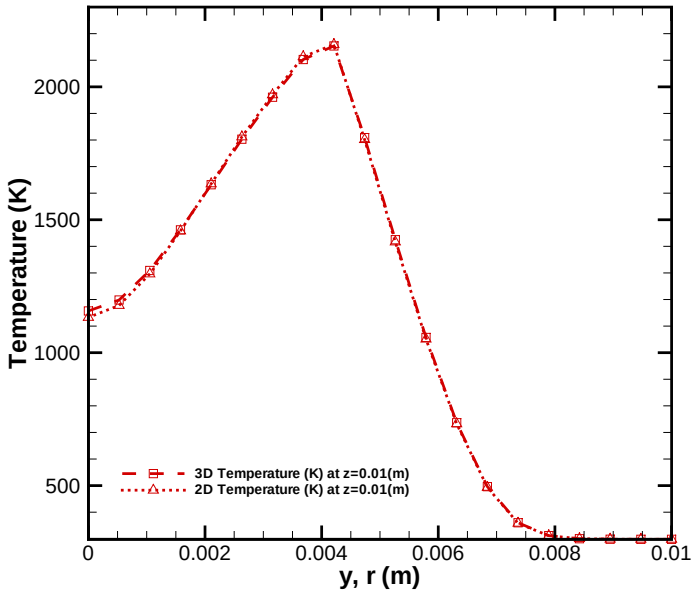
(a) 2D Axisymmetric Solution

(b) 3D Solution cross-section at x=0

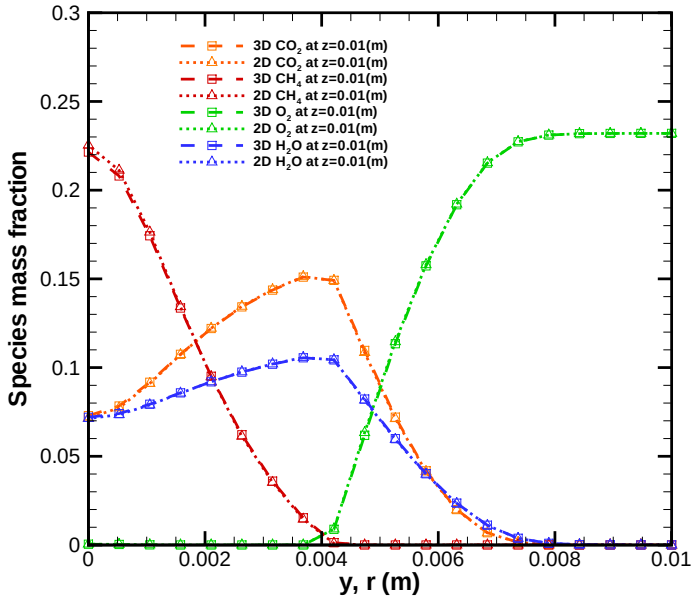
Figure 5.21: Comparison of steady methane-air co-flow laminar diffusion flame isotherms predicted from (a) 2D axisymmetric solution and (b) 3D at x=0 solution.

solution scheme, as should be expected theoretically. Finally, Figure 5.23 depicts the predicted mass fractions of the combustion reactants and products as obtained from the axisymmetric solution. Virtually identical results were obtained for the fully 3D computation.

A comparison of the results of Figures 5.19–5.23 with those given in previous studies [53, 194] (not shown here) reveals, that in spite of the inherent simplifications used in the single-step reaction mechanism, the predicted flame structures of the 2D and 3D results agree very well with the previous work. The "wishbone" structure of the high-temperature region is present and the computed lift-off and flame heights are 0.05 cm and 3.3 cm, respectively, with a maximum centerline temperature of 2,100 K. All of these

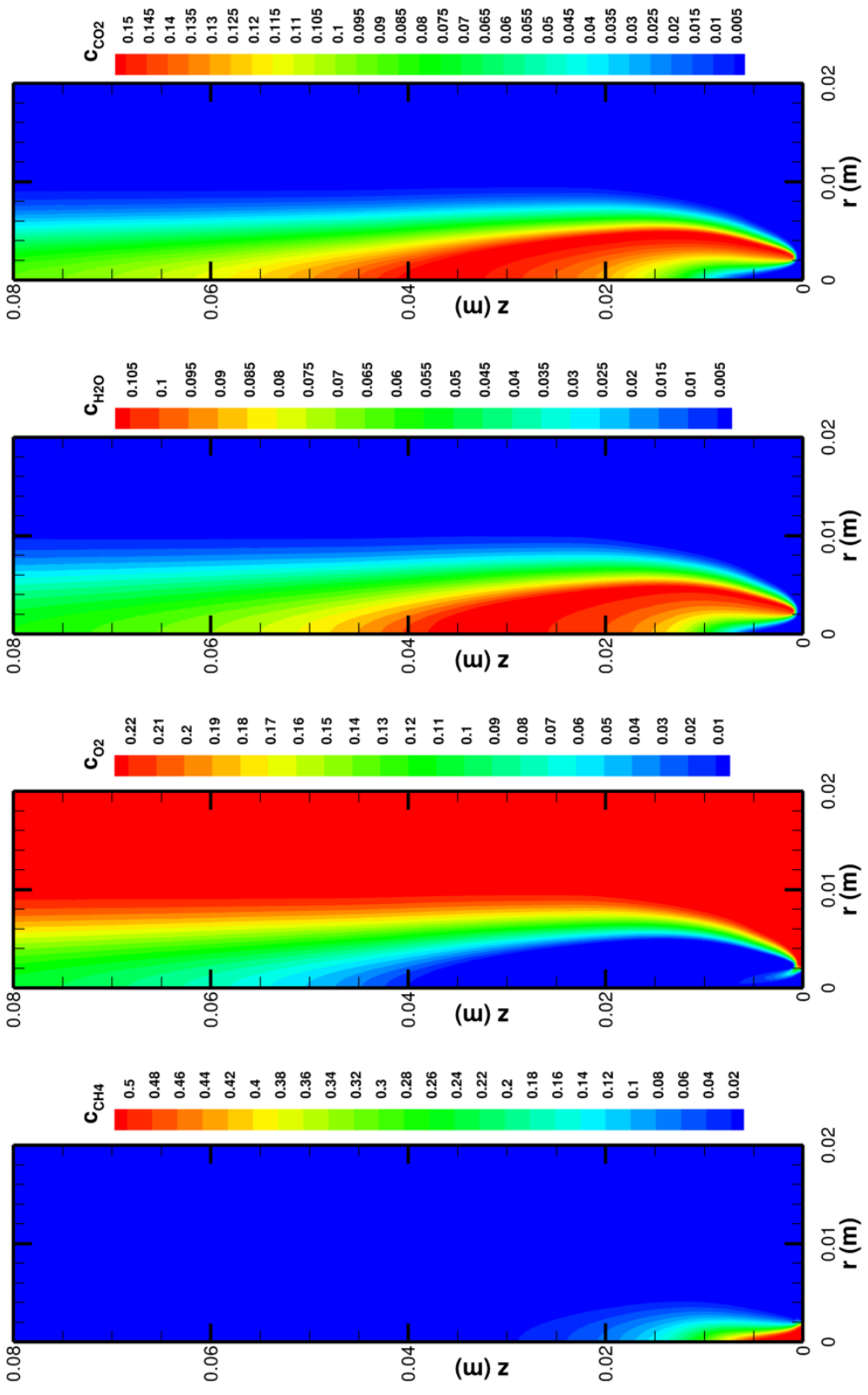


(a) Temperature profile across flame front



(b) Mass fractions across flame front

Figure 5.22: Steady methane-air co-flow laminar diffusion flame temperature isotherms predicted from (a) 2D axisymmetric solution and (b) 3D at x=0 solution and (c) temperature and (d) species mass fraction cross-sections at an axial position of flame height z=0.01 m for both 2D and 3D solutions.



(a) CH_4 (b) O_2 (c) H_2O (d) CO_2

Figure 5.23: Solution of methane-air axisymmetric laminar diffusion flame showing the species mass fractions of reactants CH_4, O_2, and products H_2O, CO_2.

values agree reasonably well with the previously published results and provide a good indication of the validity of the 2D and 3D implementations, considering that a simple one-step chemical mechanism was used here, whereas more detailed chemical kinetic mechanisms were considered in the previous work.

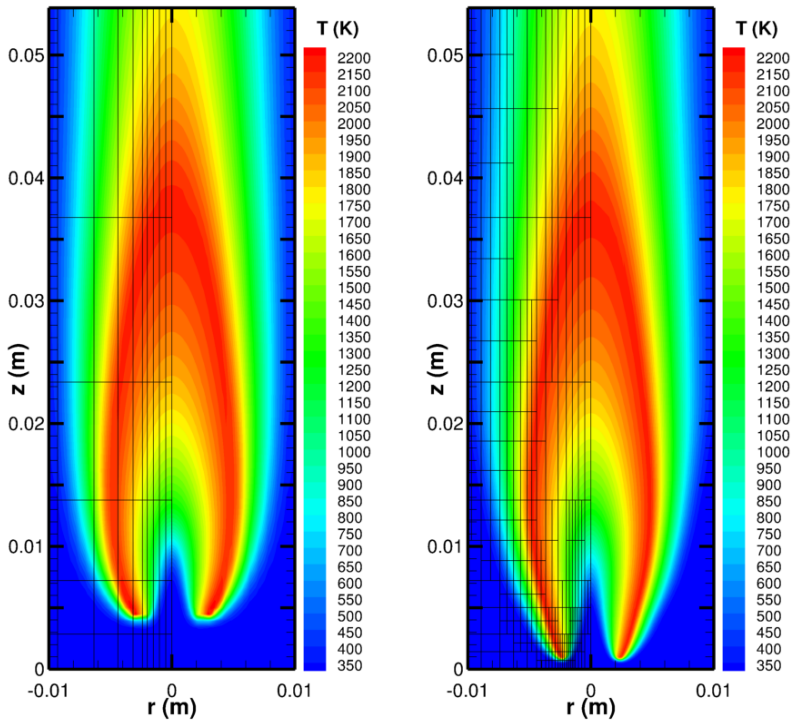
Evaluation of AMR Performance

The solution of the steady-state laminar co-flow diffusion flame also provides an opportunity to demonstrate the capabilities of the adaptive mesh refinement. The sequences of adaptively refined grids used in obtaining the 2D and 3D steady co-flow laminar diffusion flame solutions, showing both the grid blocks and computational cells, are provided in Figures 5.24 and 5.25, respectively. In both the 2D and 3D results, the effect the finer mesh resolution can be clearly seen, as the flame structure as indicated by the predicted temperature distributions becomes sharper and more fully resolved. On closer inspection of the predicted radial cross-sections of the temperature and species mass fractions given in Figure 5.26 for each of the 4 levels of mesh adaption for the 3D meshes, it is clear that mesh refinement is indeed necessary, especially near the centerline to accurately resolve the flame height and species concentrations. However, following four levels of refinement, a near "grid-independent" solution appears to have been achieved. While the results of Figure 5.26 are not conclusive evidence that a grid independent solution has been achieved (it is possible that finer solution detail could still appear as the mesh is further refined), they certainly provide strong evidence for grid independence of the predicted solutions.

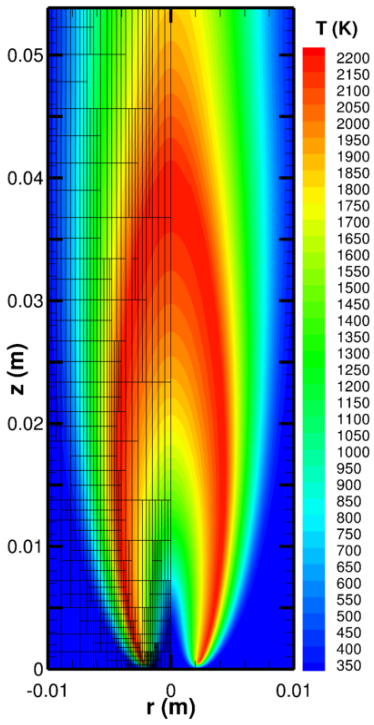
5.4.3 Unsteady Flame with Periodic Inlet Fuel Mass Flow Rate

To showcase the algorithm's performance for unsteady reactive flow prediction, which is the primary focus here, a variation of the steady laminar diffusion flame considered in the previous section is examined with a time-varying inlet fuel mass flow rate. An unsteady diffusion flame is produced by imposing a sinusoidal variation in the axial flow inlet fuel velocity of the form

$$v_z = 0.7(1 - \frac{r^2}{R})(1 + \alpha \sin \omega t) \quad \text{m/s} \tag{5.2}$$

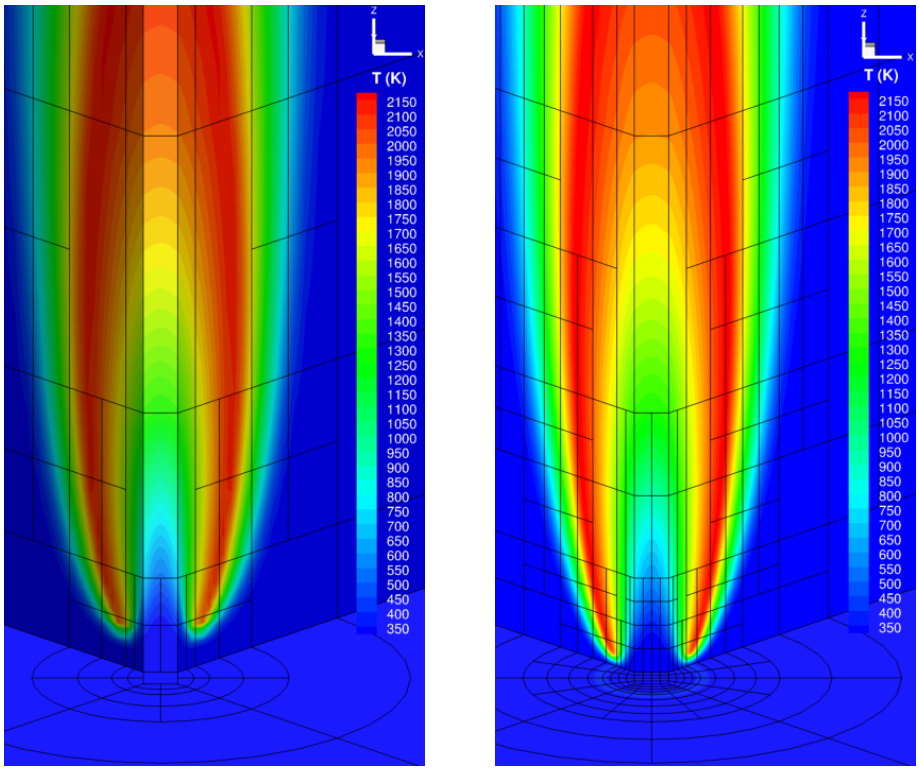


(a) 1st Level (96 Blocks) (b) 3rd Level (468 Blocks)

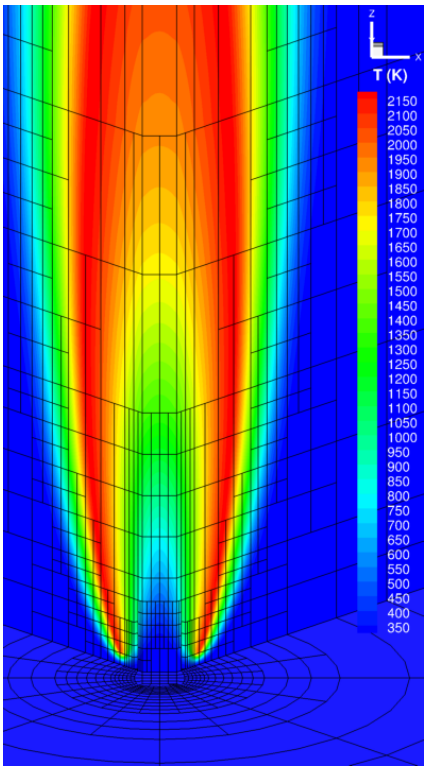


(c) 5th Level (1392 Blocks)

Figure 5.24: Solution of co-flow methane-air 2D axisymmetric laminar diffusion flame showing the computed isotherms and flame structure as well as grid blocks obtained on the (a) initial mesh; (b) AMR mesh with three levels of refinement; (c) AMR mesh with five levels of refinement.

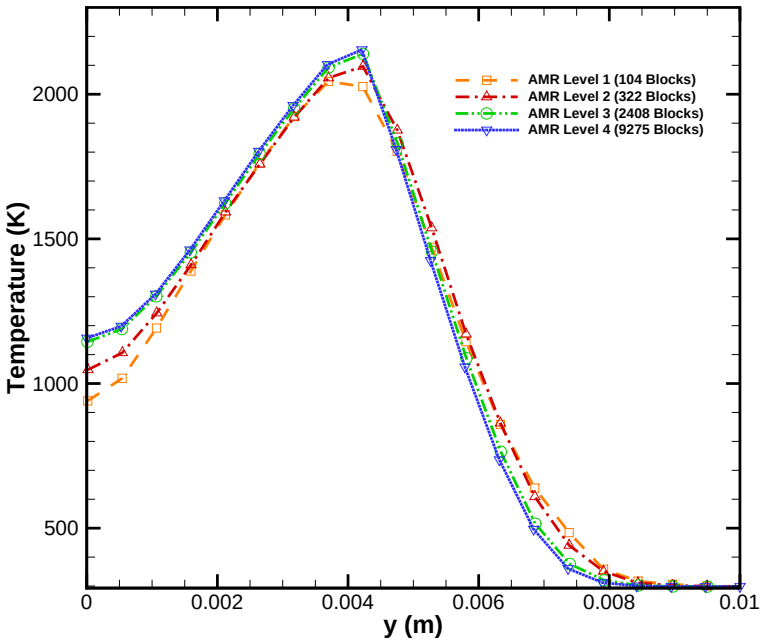


(a) 2nd Level (322 Blocks) (b) 3rd Level (2408 Blocks)

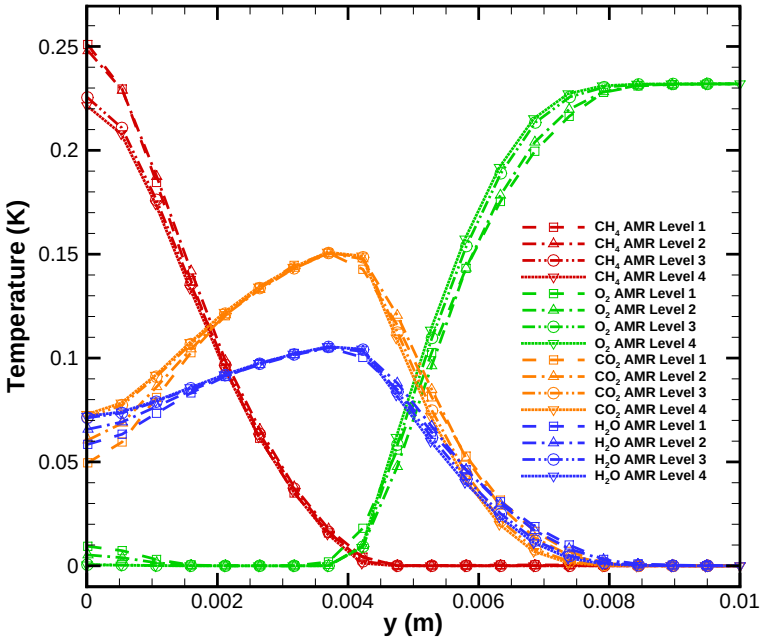


(c) 4th Level (9275 Blocks)

Figure 5.25: Solution of co-flow methane-air 3D laminar diffusion flame showing the computed isotherms and flame structure as well as grid blocks obtained on the AMR mesh with (a) two, (b) three, and (c) four, levels of refinement.



(a) Temperature across flame front



(b) Mass Species across flame front

Figure 5.26: Steady methane-air co-flow laminar diffusion flame 3D radial profiles of (a) temperature and (b) species mass fraction at an axial height of z=0.01 m at four levels of AMR grid refinement.

where α is the parameter controlling the amplitude of the velocity fluctuations and ω is the frequency of oscillation. In the experiments of Mohammed *et al.* [194] and Dworkin *et al.* [11] a range of cases were considered with varying velocity amplitudes; however, most of the published work has been presented for cases with velocity amplitudes of 30% and 50%. In this work, the velocity amplitude was selected to be 50% at a frequency of 20 Hz resulting in a fuel flow velocity that varies from 0.35–1.05 m/s over a period of 0.05 s to best compare with the experimental work presented by Mohammed *et al.* [194]. All the other parameters, geometry, boundary conditions were kept the same as for the previous steady flame simulation as summarized in Figure 5.17.

Starting with an initial solution based on the steady-state flame with the inlet fuel velocity held constant at 0.7 m/s, i.e., the solutions obtained in Section 5.4.2, a full five periods of the flame oscillations, or 0.25 s at 20 Hz, were evaluated to eliminate and thereby avoid any non-periodic solution content produced by the solution initialization process. Newton's method was used with a GMRES tolerance of 0.05, ILU(2) in 2D , ILU(0) in 3D, and at each time step the Newton iterations were converged two orders of magnitude, to a maximum of 10 Newton steps, where 5–7 is typical. The approximate Jacobian preconditioner was only updated for the first Newton step of each time-step, unless the number of GMRES iterations required increased.

For the 2D axisymmetric calculation, results of which are shown in Figure 5.27, a fixed time-step of 0.005 ms was used during the unsteady calculation and mesh refinement was carried out every 10 iterations or 0.05 ms based on the gradient of temperature, and 3 levels of mesh refinement (4 mesh levels) were used. The dynamic AMR typically produced solution grids in the range of approximately 440 to 460 blocks (14,080 to 14,720 cells) with an AMR refinement efficiency, η, of 96.3 to 94.2%. For the 3D calculation, Figure 5.28, the same fixed time-step of 0.005 ms was used during the unsteady calculation and mesh refinement was also carried out every 0.05 ms based on the gradient of temperature. Four levels of mesh refinement (5 mesh levels) were used. The dynamic AMR typically produced solution grids in the range of approximately 24,800 to 25,700 blocks (12,697,600 to 13,158,400 cells) with an AMR refinement efficiency, η, of 95.2 to 95.0%.

A side by side comparison of the temperature isotherms at five 0.01 s intervals of the 20 Hz periodic cycle are shown in Figure 5.29 for a cross-section (r–z plane) at y=0 of

the predicted 2D axisymmetric and 3D solutions. Similar to the steady laminar diffusion results presented above, the predicted 2D and 3D solutions agree very well providing further support for a valid and correct implementation of the proposed 3D finite-volume solution method.

The predicted flames shape and structure of the 2D and 3D simulations also match well with the experimental results of Mohammed *et al.* [194] as shown for comparison in Figure 5.30. The five cross-section iostherms correspond to 0.01 s of the 0.05 s (20 Hz) periodic fluctuations of the driven flame. The most significant difference is in the over prediction of temperature. However, this is related to the use of the simplified non-reversible one-step reaction mechanism used to model the methane-air chemistry in the present work, whereas a more detailed mechanism was used in the previous studies.

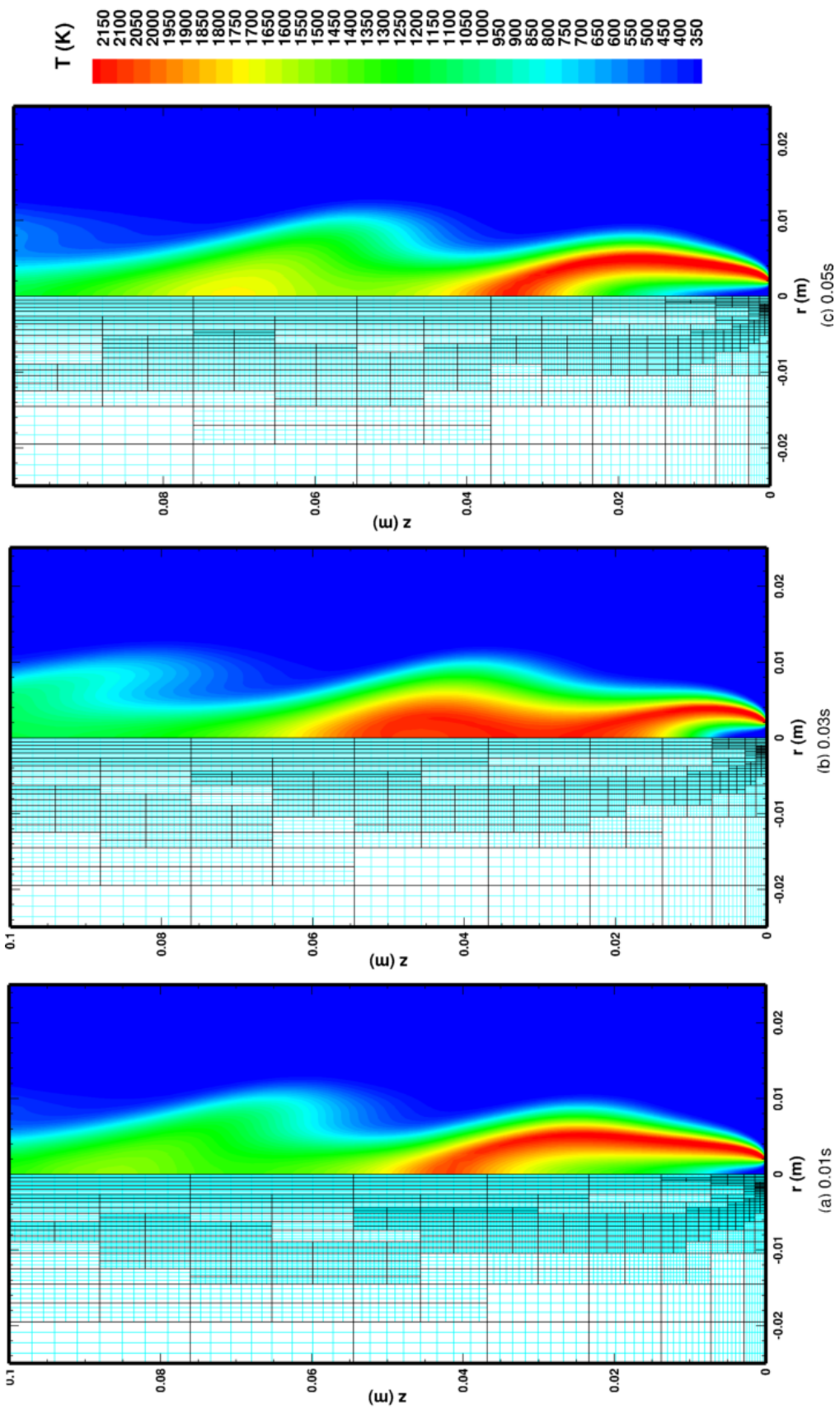
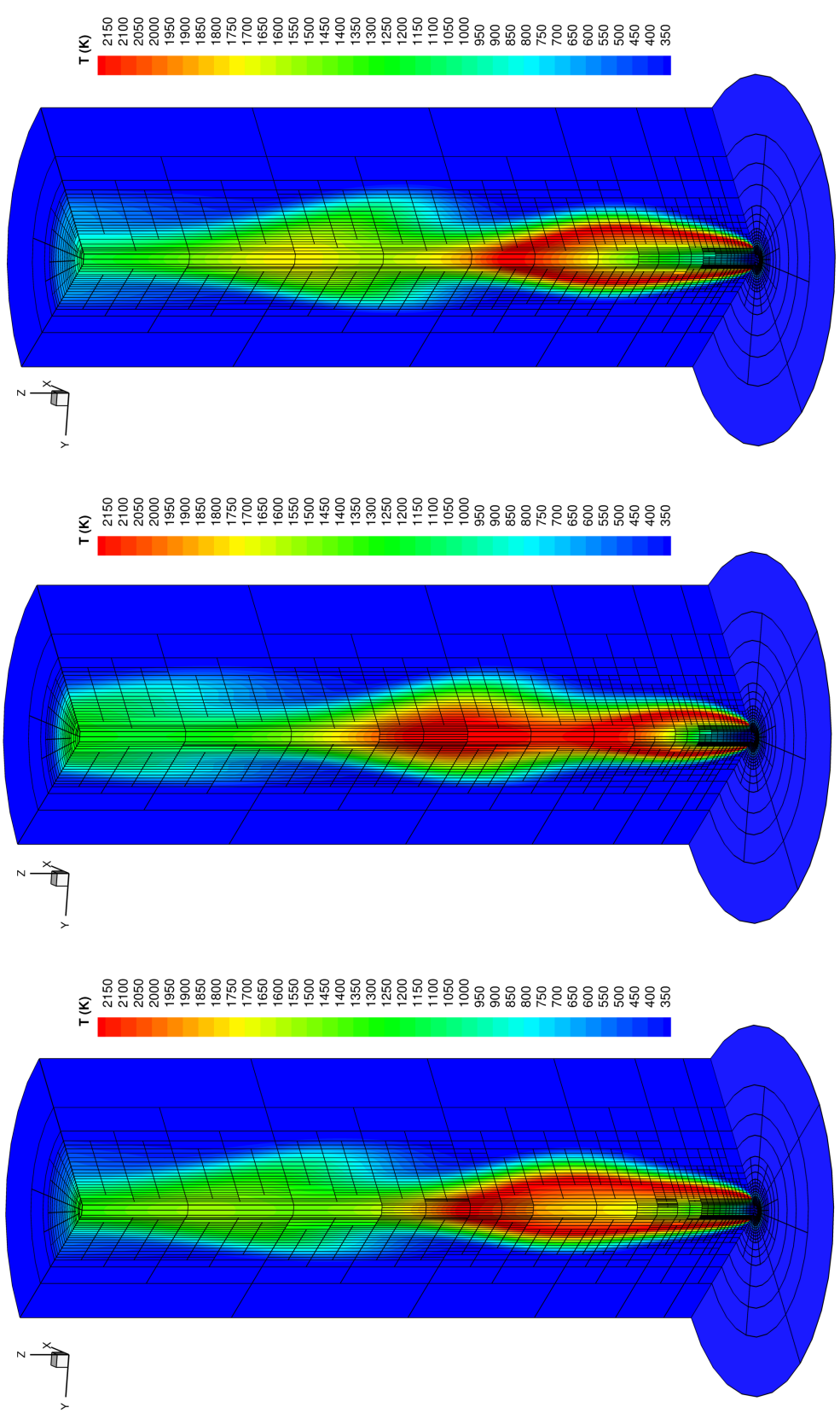
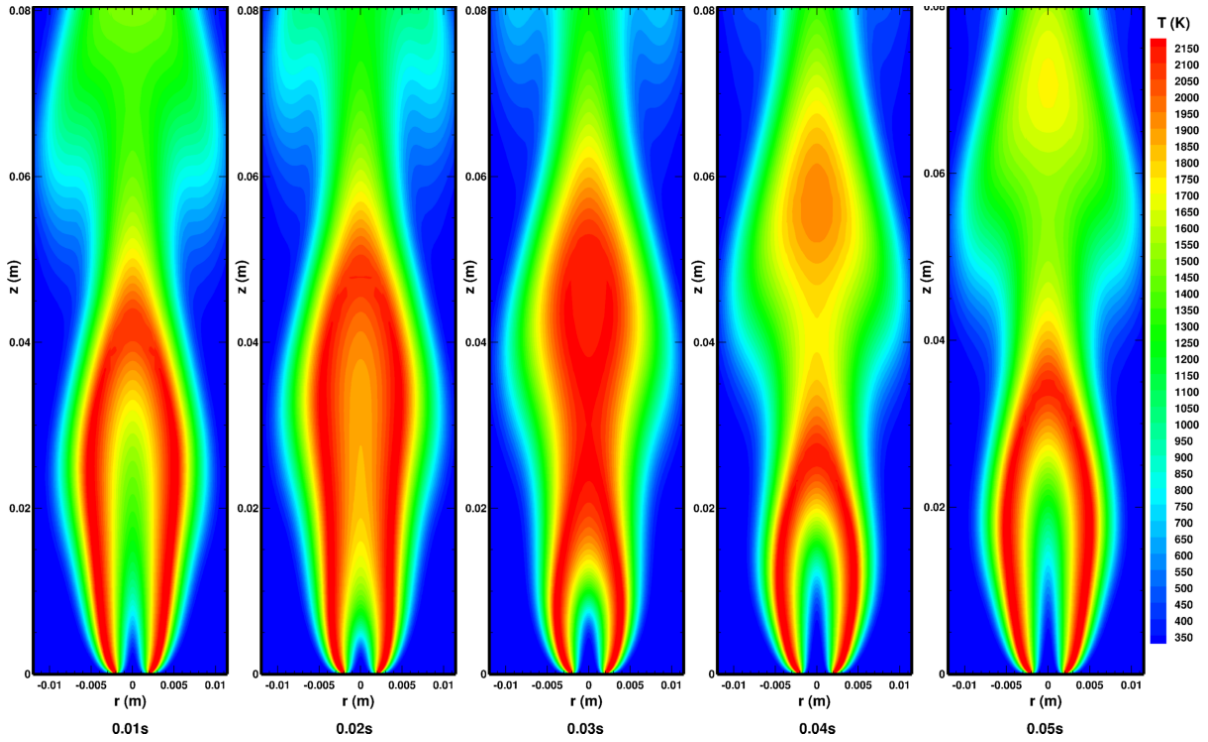


Figure 5.27: 2D axisymmetric solution of a time-varying methane-air co-flow laminar diffusion flame at 3 time intervals during its 0.05 s periodic cycle, with a) 456 block (14,592 cell), b) 444 block (14,208 cell), and c) 447 block (14,304 cell) grids with (4×8 cell) blocks each with 3 levels of mesh refinement. The right hand side shows the isotherms and the left shows the mesh refinement and adaption at the particular time.

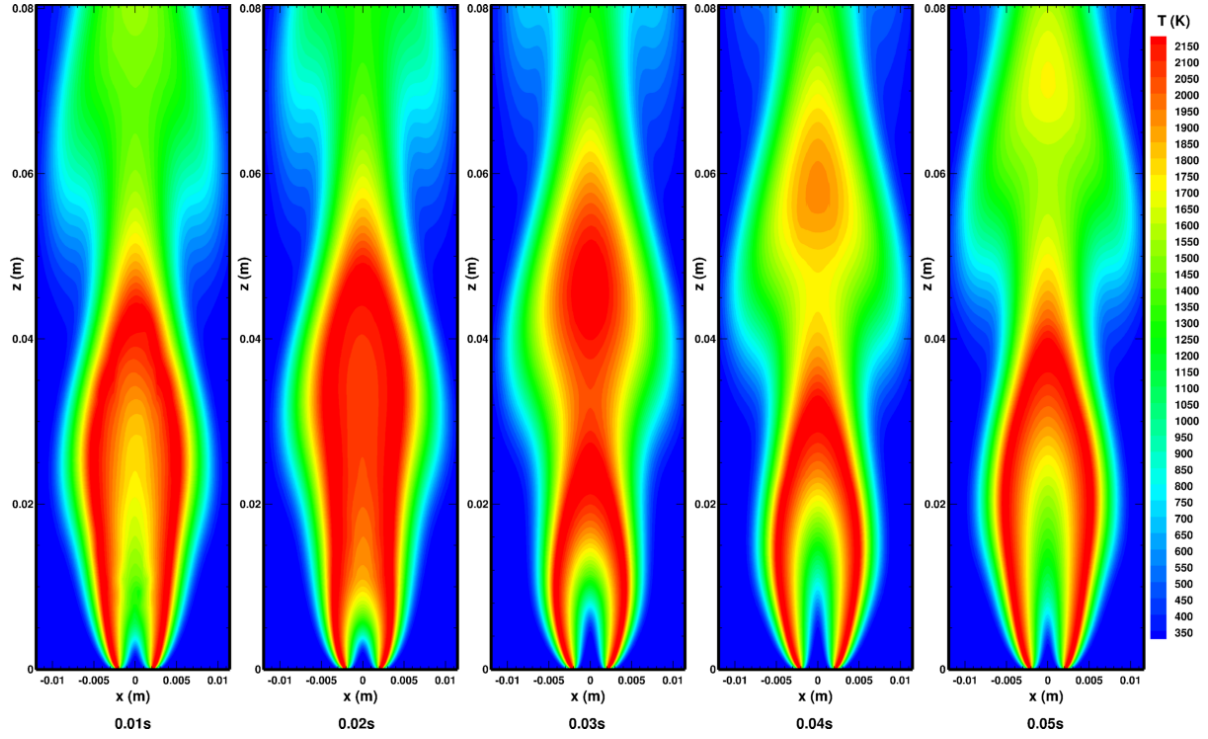


(a) 0.01 s (b) 0.03 s (c) 0.05 s

Figure 5.28: 3D solution of a time-varying methane-air co-flow laminar diffusion flame at 3 time intervals during its 0.05 s periodic cycle, with a) 25,592 block (13,103,104 cell), b) 24,927 (12,762,624 cell), and c) 25,137 block (12,870,144 cell) grids with (8×8×8 cell) blocks each with 4 levels of refinement.

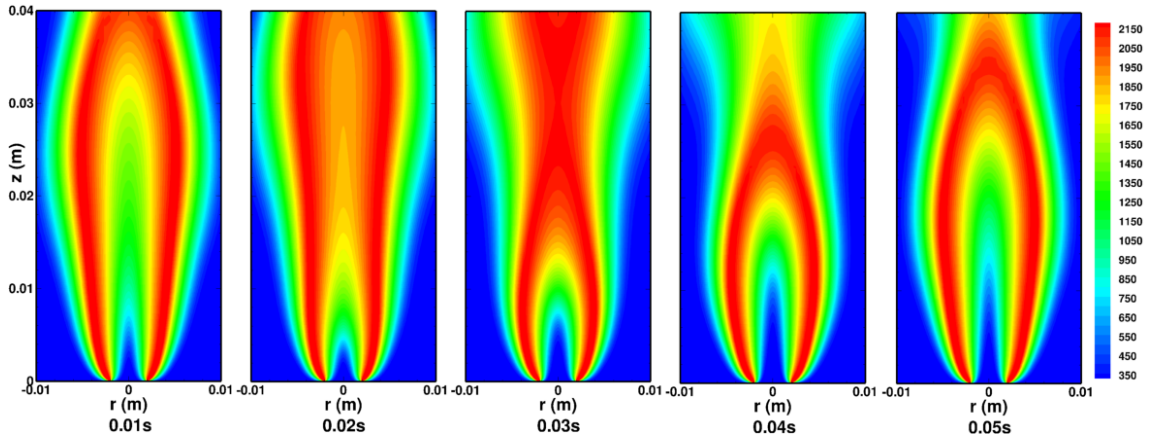


(a) 2D Axisymmetric Solution

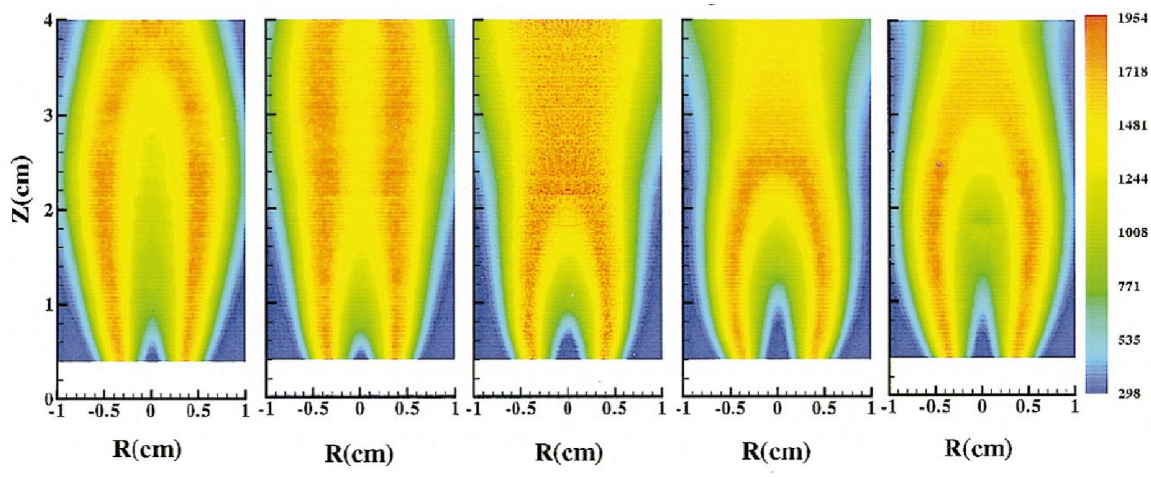


(b) 3D Solution Cross-Section at y=0

Figure 5.29: Comparison of time-varying methane-air co-flow laminar diffusion flame isotherms at five 0.01 s intervals of the 20 Hz periodic cycle for (a) 2D axisymmetric and (b) 3D solution procedures.



(a) Numerical Isotherms



(b) Experimental Isotherms

Figure 5.30: Comparison of a periodic time-varying methane-air co-flow laminar diffusion flame isotherm contours at five 0.01 s intervals from (a) numerical computation and (b) experimental results of Mohammed *et al.* [194].

5.5 3D Conical Laminar Premixed Flame

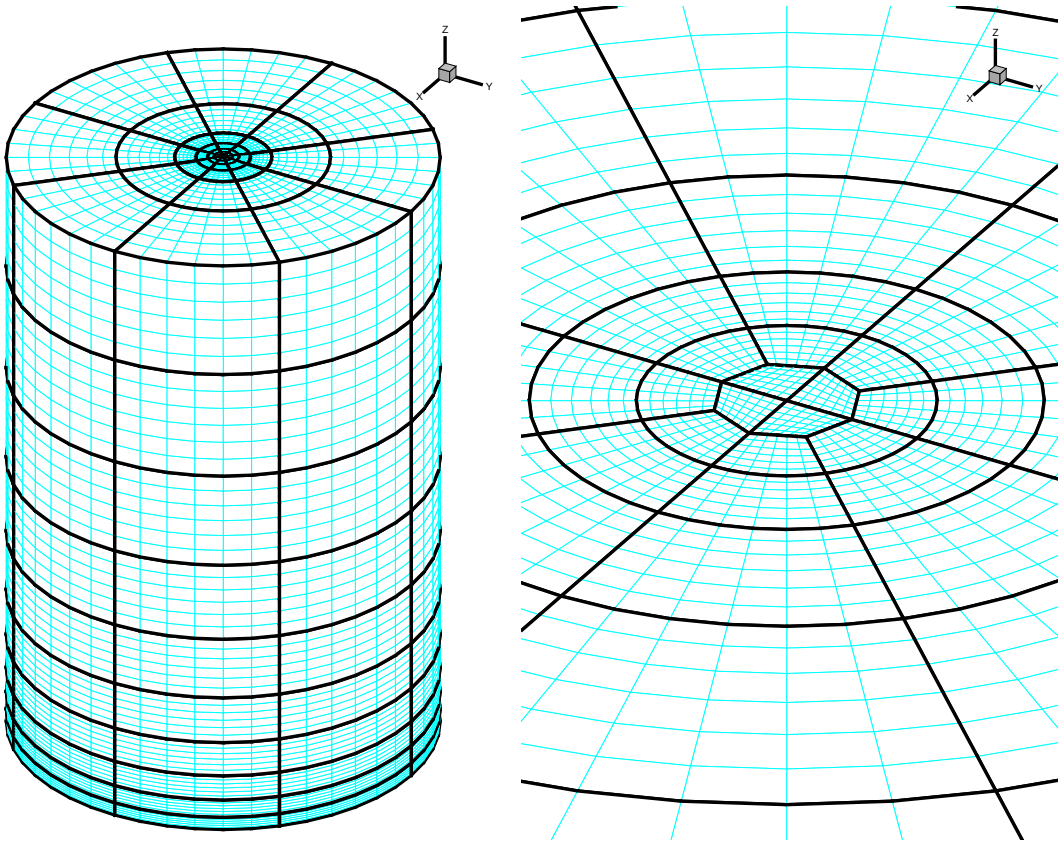
The previous reactive flow problem examined the well characterized and predictable behaviour of non-premixed flames where the fuel and oxidizer are simultaneously mixed through diffusion and convective processes igniting and burning all within the burner. In a premixed flame, the fuel and oxidizer are first thoroughly mixed at the molecular level before entering the combustion chamber and the resulting mixture is injected into the chamber where it is ignited and burnt. Conical methane-air premixed laminar flames, unlike diffusion flames, do not typically form a steady flame, but rather instead exhibit an oscillation or flickering of the flame edge and tip. Buoyancy-induced interactions between the hot products and the cold environment produce velocity fluctuations in the reactants which result in the flame flicker.

Various aspects of flame/buoyancy coupling in conical and V-shaped premixed flames have been investigated in previous experimental studies. This includes the flame flicker frequency as a function of flow velocity, pressure, and strength of the gravitational force [195,196]. Kostiuk and Cheng [197,198] have also shown that premixed flame oscillations with characteristic frequencies in the range of 10–20 Hz can be correlated to a wide range of system parameters. More recent experimental investigations into flame-intrinsic Kelvin–Helmholtz instabilities for inverted conical premixed flames by Guahk *et al.* [199] have shown that the Strouhal number, representing the dimensionless frequency of the oscillations, can be correlated with the flow Richardson number.

Shepherd *et al.* [200] have previously made comparisons of experimental measurements and numerical predictions of laminar premixed flame flicker. The low-Mach-number numerical solution method with AMR developed by Day and Bell [53] was used to obtain the unsteady numerical solutions. In this previous research, it was found that accurate predictions of the quasi-periodic flame flicker could be obtained given the correct flame conditions as dictated by the equivalence ratio and inlet mass flow rate. The premixed methane-air laminar flame examined herein attempts to reproduce the quasi-periodic behaviour observed in the previous combined experimental and numerical study of Shepherd *et al.* [200] and the experimental studies of Kostiuk and Cheng [196–198] using the proposed time-accurate parallel implicit AMR algorithm.

The flame configuration considered herein is very similar to that used for the laminar

diffusion flame cases, however in this case the fuel being injected through the inlet is replaced with a premixed methane-air mixture at a fixed equivalence ratio. Fully three-dimensional solutions were obtained using the same initial and boundary conditions as described previously and summarized in Figure 5.17; however, the size of the computational domain used is much larger with a height of 0.3 m and outer radius of 0.1 m to avoid the effects of far-field boundary data prescription on the predicted solutions. The initial mesh blocking and spacing is concentrated near the centerline, as depicted in Figure 5.31(b) to better capture the thin flame fronts associated with premixed flames. Premixed reactants with an equivalence ratio of $\phi = 0.8$ are injected into the domain through a 0.025 m diameter inlet with a parabolic velocity profile having a peak velocity of 0.73 m/s. The gas mixture in the interior of the domain is taken to be quiescent air at standard atmospheric conditions and for the premixed case there is no co-flow.



(a) 3D Initial Mesh (b) Closeup of centerline

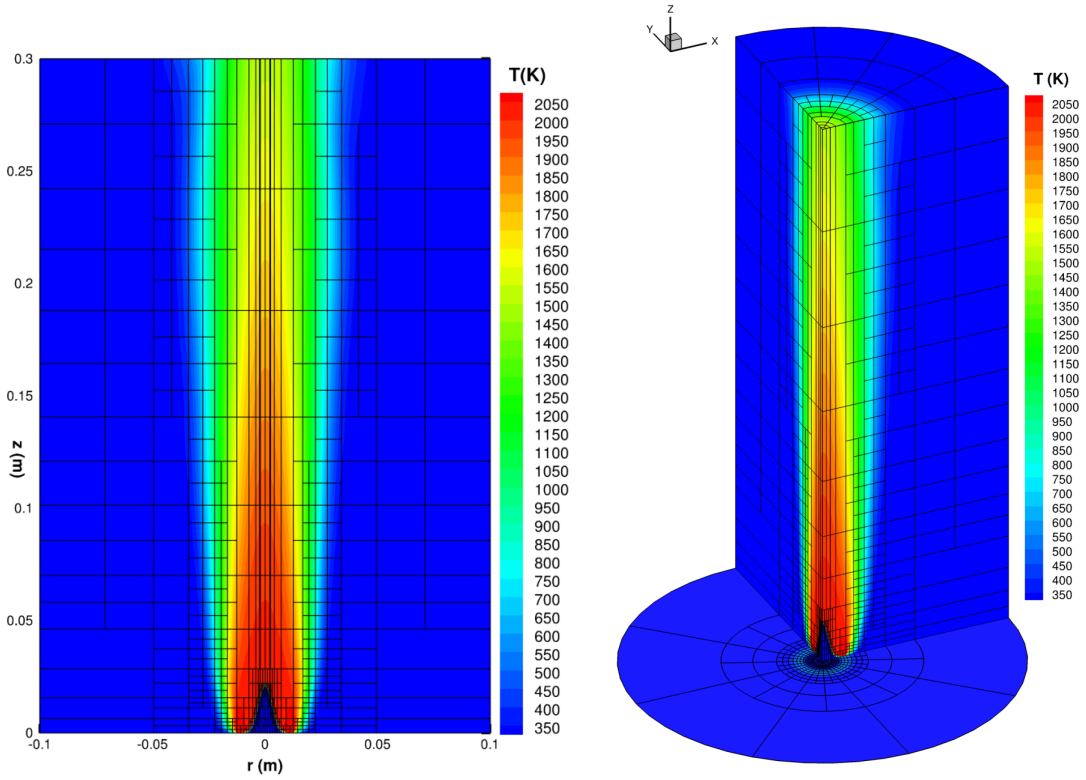
Figure 5.31: Initial 3D coarse 520 (6×6×6) blocks with 112,320 cell grid for premixed laminar flame solution.

5.5.1 Steady Flame in Absence of Gravity

A steady-state or time-invariant solution for the premixed flame was first considered. As the oscillations in conical premixed laminar flames are buoyancy driven or induced, in the absence of gravity, results in a steady non-flickering flame is produced. This has been demonstrated through previous experiments [195, 196] and is readily verified computationally by setting the source term associated with gravitational acceleration to zero.

Similar to the previous 3D diffusion flame simulations described above, the Roe flux function with the Venkatakrishnan limiter was used with low-Mach-number preconditioning with a reference Mach number, $M_{r_{min}}$, of 0.1. The Newton-Krylov-Schwarz algorithm with a GMRES tolerance of 0.01 and ILU(0) fill level was used to achieve a steady-state solution. The initial unrefined coarse mesh consisting of 520 (6×6×6) grid blocks and 112,320 computational cells is given in Figure 5.31. The resolution of this coarse initial mesh was however found to be insufficient to obtain a stable and steady premixed flame solution (the flame would collapse on the inlet and extinguish). It was found that a minimum of 3 levels of mesh refinements were required to achieve a stable and accurate solution. The final solution mesh with 4 levels of refinement based on the gradient of temperature is shown in Figure 5.32 and contains 8,430 blocks and 1,820,880 cells with a refinement efficiency of η=97%. The refined AMR mesh provides high resolution of the thin premixed flame interface as shown in the predicted isotherms of Figure 5.32.

The predicted steady-state laminar premixed flame solution of Figure 5.32 exhibits the expected features of the steady flame behaviour. When compared with a Schlieren image of a similar steady flame observed by Kostiuk and Cheng [196] in micro-gravity as shown in Figure 5.32(a), the density contour cross-sections at y=0 match reasonably well. The numerical flame treated the boundary condition at the fuel inlet by imposing a parabolic velocity profile without any consideration of the upstream geometry of the fuel tube and so the interaction between the flame and fuel inlet does not quite match exactly with the experiment. However, the actual conical flame and premixed flame interface between products and reactants is well captured.

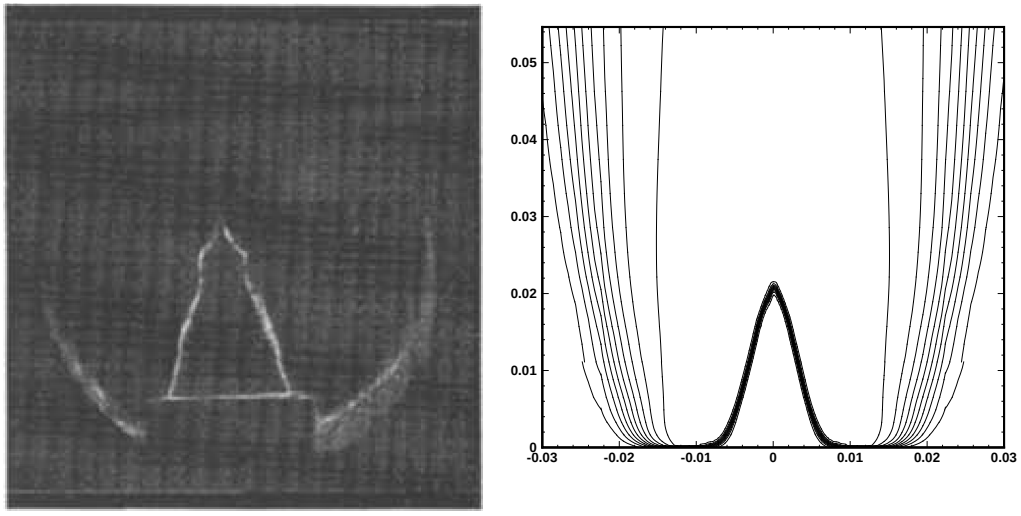


(a) 2D Cross-section (b) 3D 1/4-section

Figure 5.32: Solution of 3D steady methane-air premixed flame in the absence of gravity (a) cross-section at y=0 and (b) 3D quarter sections showing temperature isotherms, flame structure, and block boundaries obtained using 8,430 (6×6×6) blocks with 1,820,880 cells with four levels of mesh refinement.

5.5.2 Unsteady Flame with Buoyancy-Induced Oscillations

The unsteady version of the preceding premixed laminar flame with the buoyancy-induced oscillations was also considered by including a non-zero gravitational force equivalent to that on the Earth's surface. Starting from the zero-gravity "steady" solution discussed in Section 5.5.1 above and re-introducing the gravity source term, so as to "turn gravity back on", the predicted premixed flame naturally evolved into an unsteady oscillating flame. In this case, the time-accurate BDF2-NKS algorithm was used with a GMRES tolerance of 0.05 and ILU(0), where each time step of the Newton iteration was converged two orders of magnitude, to a maximum of 10 Newton steps. The approximate Jacobian preconditioner was only updated for the first Newton step of each time-step, unless the



(a) Schlieren image (b) 2D Cross-section with density contours

Figure 5.33: Comparison of (a) Schlieren images of experiments by Kostiuk and Cheng [196] and (b) numerical cross-section at y=0 density contours of a laminar premixed methane-air flame in the absence of gravity.

number of required GMRES iterations increased. A fixed time-step of 0.01 ms was used during the unsteady calculation and mesh refinement was carried out every 0.1 ms, 10 steps, based on the gradient of temperature, and 3 levels of mesh refinement (a maximum of 4 mesh levels) were used. As mentioned, the unsteady solution was initialized using the steady solution discussed in Section 5.5.1 and depicted in Figure 5.32 which contained 8,430 (6×6×6) blocks with 1,820,880 cells. As the solution evolved toward a quasi-periodic solution the dynamic AMR typically produced solution grids in the range of approximately 12,000 to 13,000 blocks (2,592,000 to 2,808,000 cells) with a refinement efficiency, η, of 95.5 to 95.1%. The calculation was run for 0.75 s to allow the solution to evolve toward a quasi-periodic solution in which the influences of the initial conditions and startup transients are effectively eliminated.

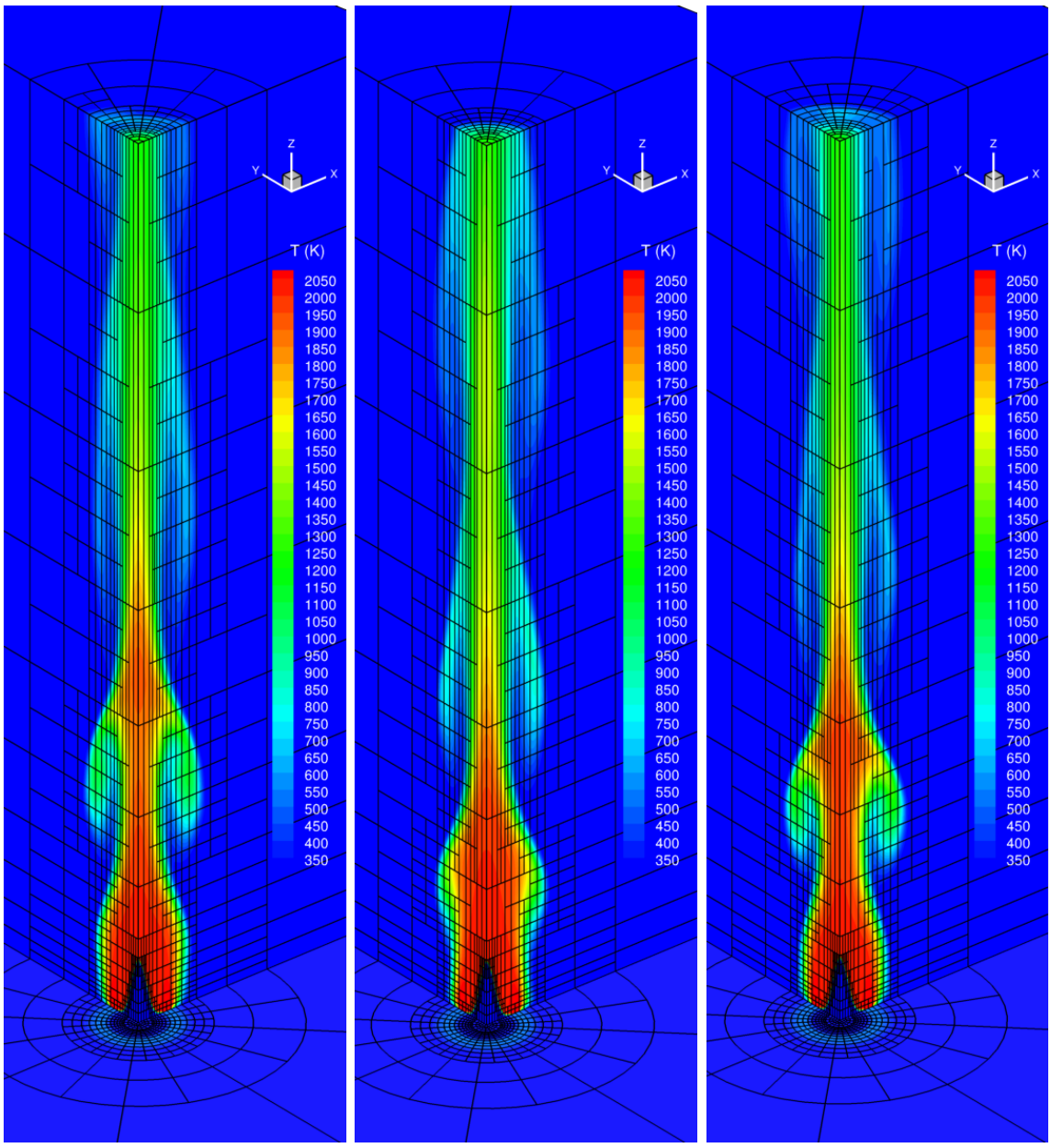
The resulting predicted solution for premixed flame-flicker, as shown in Figure 5.34, matches well with the 2D axisymmetric numerical results of Shepherd *et al.* [200] in terms of temperature, flame front size, and centerline velocities. When compared with experimental Schlieren imagery of Kostiuk and Cheng [196] for the unsteady flame as reproduced in Figure 5.35, current predictions of the density distribution cross-sections

at y=0 agree well with the Schlieren images. The buoyancy interaction of the hot products and cold ambient air that produces the unsteady oscillations is quite evident in the results of Figure 5.34.

Figure 5.36(a) shows six predicted isotherm cross-sections in the y=0 plane over a 0.1 s time period showing the formation and evolution of the flame edge vortices. The adaptive mesh refinement indicated by the block boundaries in Figure 5.36(b) would seem to accurately track and resolve the vortices as they are shed and propagate downstream of the flame. The predicted flame tip height is nominally 22 mm which matches well with Shepherd's *et al.* [200] experimental estimates of the measured flame height that was reported to be about 20 mm.

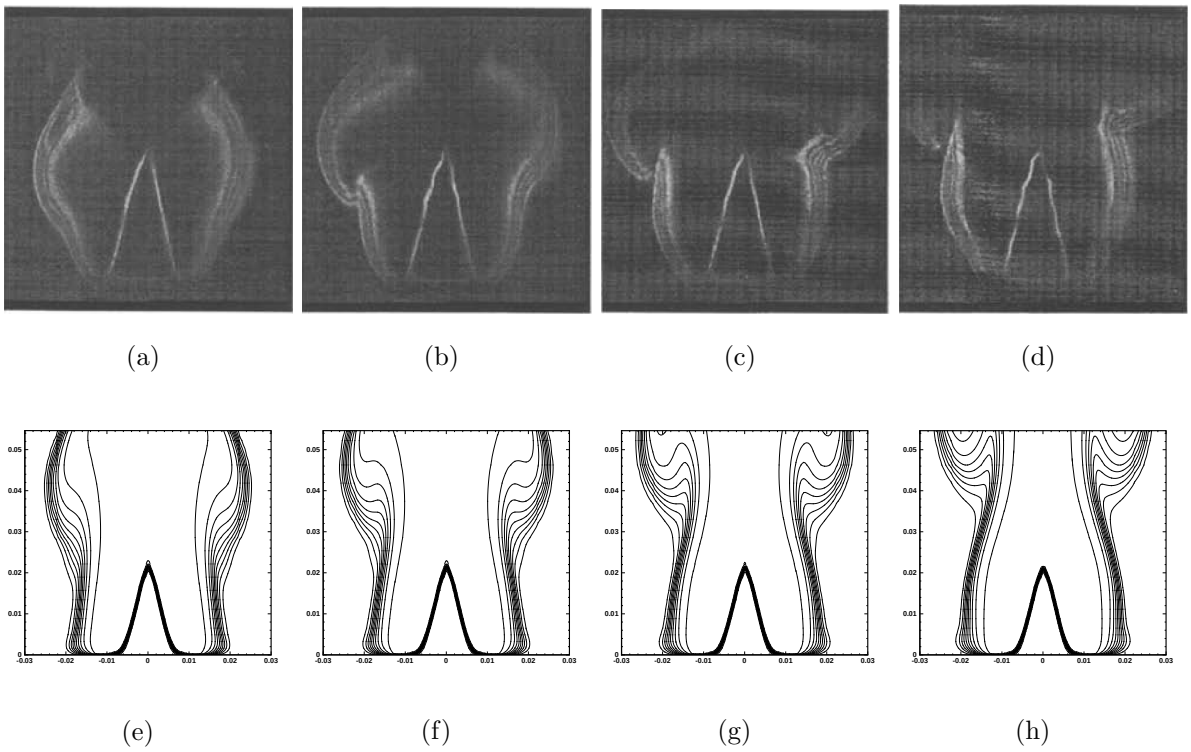
The primary oscillation frequency of the flame oscillations or flicker can be examined by considering the history of the the centerline axial component of velocity as shown in Figure 5.37. Applying a Fourier transform to the periodic component of the velocity time history, starting at 0.4 s, and plotting the amplitude of the transformed signal, or what is commonly referred to as power spectrum, versus signal frequency as given in Figure 5.38, a dominant peak at 10.4 Hz is observed. This predicted primary frequency of the flame oscillations agrees very well with Shepherd's *et al.* [200] measurement of 10.2 Hz and is consistent with Kostiuk and Cheng [197] characterization of premixed methane flame oscillations in the 10–20 Hz range.

Overall the combination of the time-accurate parallel implicit algorithm with dynamic AMR does a very good job at resolving the buoyancy driven instability. The adaptive mesh refinement, as mentioned, was required to provide sufficient resolution as to accurately resolve the thin flame front and allow the buoyancy induced oscillations to develop naturally. The agreement with previous numerical and experimental studies is rather good considering the limitations of the reduced methane-air chemical mechanisms.



(a) 0.4 s (b) 0.44 s (c) 0.48 s

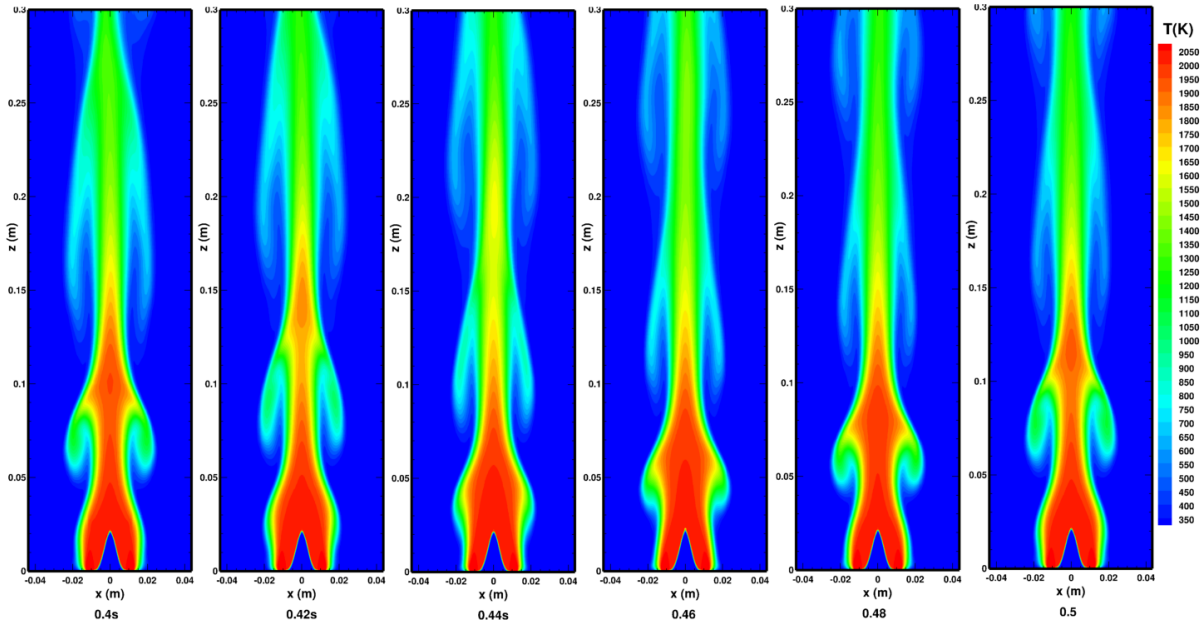
Figure 5.34: 3D solution of a time-varying methane-air co-flow laminar premixed flame at 3 time intervals, $t = 0.4, 0.44, 0.48$ s, during its approximately 10 Hz quasi-periodic cycle, with a) 12,707 block (2,744,712 cell), b) 12,539 block (2,708,424 cell), and c) 12,553 block (2,711,488 cell) grids with (6×6×6 cell) blocks each with 4 levels of mesh refinement.



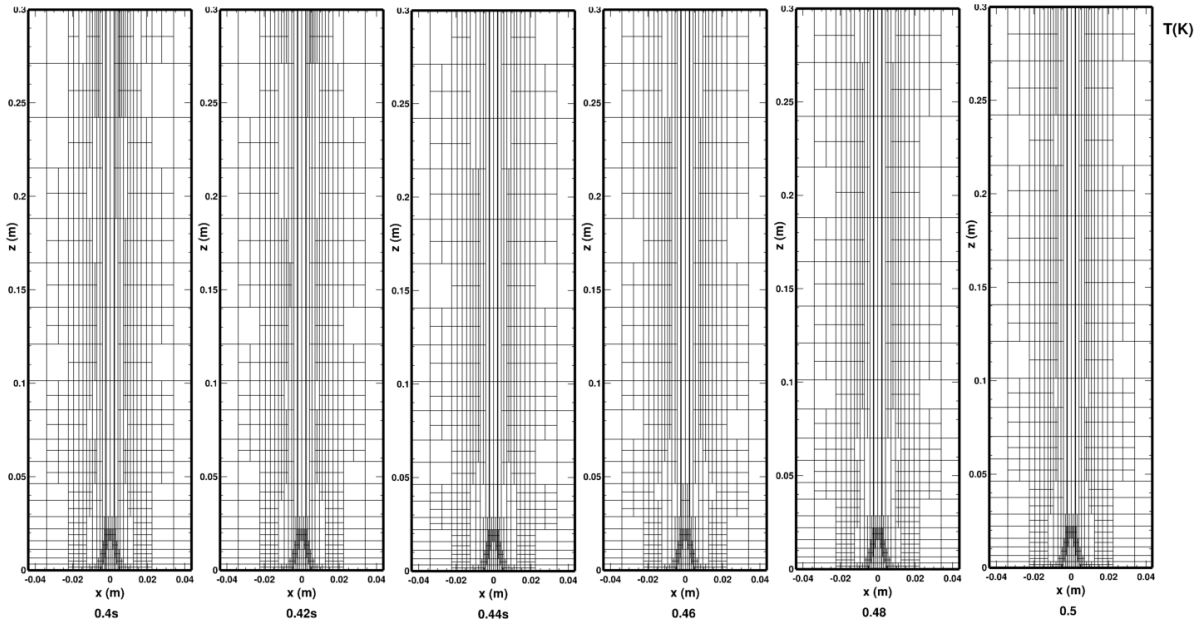
(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 5.35: Comparison of (a-d) Schlieren images of experiments by Kostiuk and Cheng [196] and (e-h) predicted numerical cross-sections of the distributions of flow density in the y=0 plane for $t = 0.44, 0.45, 0.46, 0.47$ s for unsteady laminar premixed methane air flame with gravity.



(a) Temperature Isotherms



(b) Block boundaries

Figure 5.36: 3D predicted solution cross-sections in y=0 plane of a methane-air premixed flame showing the (a) computed isotherms and (b) block boundaries with four levels of mesh refinement at 6 time intervals, $t = 0.4, 0.42, 0.44, 0.46, 0.49, 0.5$ s, showing the approximately 10 Hz buoyancy driven oscillations.

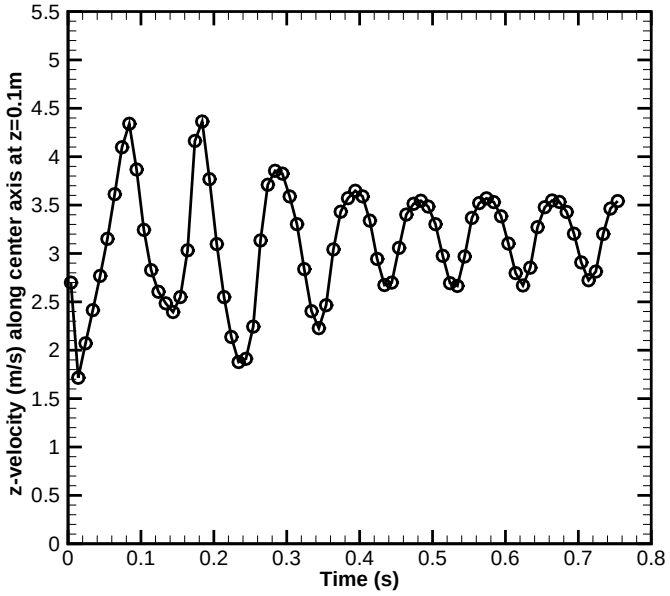


Figure 5.37: Conical premixed methane-air centerline flow velocity on the centerline at an axial height of z=0.1 m showing a quasi-periodic oscillation of approximately 10 Hz.

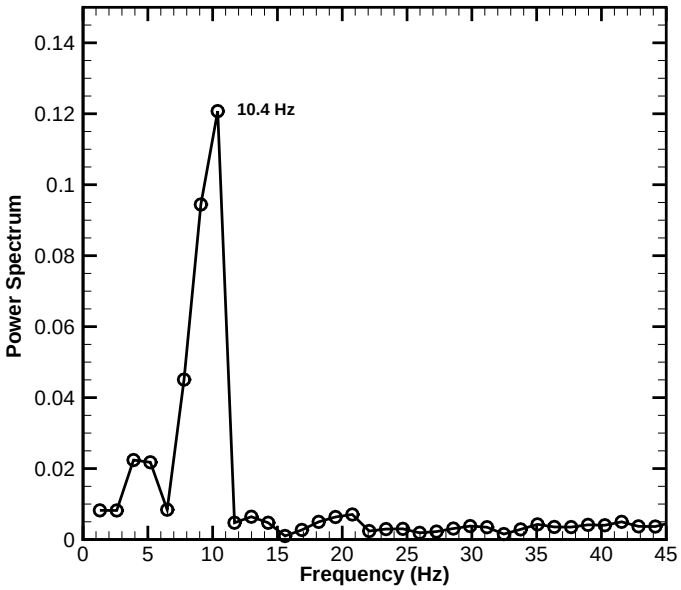


Figure 5.38: Fourier power spectrum or modes of the predicted time history of flow velocity on the centerline at an axial height of z=0.1 m showing a dominant frequency of 10.4 Hz.

Chapter 6

Numerical Results: Algorithm Performance

As stated in the introduction, the primary objective of this research was to develop an algorithm that reduces the time it takes to achieve accurate numerical solutions of laminar reactive flows. The numerical results presented in Chapter 5 have demonstrated the proposed algorithm's ability to produce accurate numerical predictions of two- and three-dimensional laminar non-premixed and premixed combusting flows for both steady and unsteady flames under low-Mach-number flow conditions. While some algorithm performance characteristics relating to the benefits of low-Mach-number preconditioning and refinement efficiency have been already discussed in the previous chapter, along with the numerical results, for the most part, the details of the performance of the proposed solution methodology were not discussed as the focus to this stage was on verification and validation.

This chapter deals specifically with the performance of the parallel implicit finite-volume AMR scheme as it pertains to computation time and parallel efficiency. Using the solution results from Chapter 5, the proposed algorithm's convergence rates, for steady flow problems, and overall solution time, for time-accurate predictions are compared to other contemporary approaches for steady and unsteady simulations. As the Newton-Krylov algorithm has many adjustable or tunable parameters requiring user specification that can have a large effect on solution time, a discussion of parameter selection is also included. As this is a parallel algorithm, parallel efficiency and scalability is also investi-

gated as deficiencies in these areas will dominate over serial performance at even modest numbers of processors and/or cores. As part of the parallel analysis, the performance and potential drawbacks of the additive Schwarz preconditioning technique are investigated.

6.1 Newton-Krylov-Schwarz Parameter Selection

Almost all numerical methods require some user defined input parameters, such as time step size and parameters controlling the mesh generation, which are typically used to balance algorithm accuracy and stability usually at the cost of some trade-off in performance. Newton-Krylov methods are no exception to this in that there are numerous adjustable parameters, such as the various inner and outer loop convergence tolerances, preconditioning control parameters, and others, as fully described in detail in Chapter 4 of this thesis.

A common stated detraction of Newton-Krylov methods is that there are almost too many adjustable parameters and determining an optimal set in some cases can be difficult for an inexperienced user. Multiple studies have been performed to investigate the effects of parameter selection on algorithm performance; however, a set of optimal parameters is often dependent on the equation system, system architecture, and specific flow regime of interest. Zingg and associates have published multiple studies investigating Newton-Krylov parameter selection for two- and three-dimensional steady aerodynamic flows [103, 112, 181, 201] as well as for unsteady flows [78, 105]. Keyes, Gropp and associates have also carried out considerable research into how Newton-Krylov-Schwarz parameter selection [108, 109, 111] affects overall algorithm performance, but make no claim that the optimal combination was found [101]. However, to the experienced user, this rich set of algorithm options can provide a great deal of architectural and application adaptivity [101].

Although a detailed investigation into all of the controlling parameters for the present algorithm is not considered in this thesis, this section provides some investigation and discussion into the key parameters and how they affected solution convergence rates for the test cases and flow problems considered in Chapter 5.

6.1.1 Nonlinear Convergence Tolerance for Newton Scheme

For the non-reacting cases, overall residual reductions of more than ten orders of magnitude could typically be achieved, as shown in Figures 5.8 and 5.13(b). However, for the laminar reactive flow cases considered, only about six or seven orders could be achieved before the solution would stall. This stall is most likely related to the finite-rate chemistry source terms behaviour as the mass fraction of species approaches zero. The species mass fractions, c_s, are required to satisfy both maximum and minimum principles and lie within the range $0 \leq c_s \leq 1$. This restricted range of validity leads to inaccuracies in the GMRES matrix-vector products generated using Fréchet derivative approximations as well as in the approximate Jacobian used as a preconditioner, particularly as $c_s \to 0$. The empirical curve fits used for the thermodynamic and transport data outlined in Chapter 2, subsection 2.2.3 may also be hampering convergence as the residual gets small. Recently, Veldhuizen *et al.* [117] have proposed using a modified globalized inexact projected Newton method based on the projected Newton approaches originally designed for nonlinear optimization problems with constraints to preserve species mass fraction positivity. While this approach seems quite promising, this was not implemented in this work due to time constraints. For the unsteady problems, this is also really a non-issue as full convergence is typically not required, and often not recommended as reducing the residual too much will lead to oversolving the system and unnecessarily increasing the computational cost of updating the solution at each time-step [78].

6.1.2 Linear Convergence Tolerance for GMRES Algorithm

The linear or GMRES convergence tolerance parameters govern how accurately the linear system is solved at each Newton step. A high tolerance is generally more computationally costly whereas too low of a tolerance can lead to an inaccurate solution update that may ultimately cause the outer Newton iterations to diverge. For many applications a GMRES reduction of one order, a tolerance of 0.1, has been shown to be sufficient [103].

In the non-reacting inviscid and viscous cases described in Chapter 4, a tolerance of 0.1 was indeed found to be sufficient. For the reacting cases however, a tighter tolerance of 0.05–0.01 was found to be necessary to avoid divergence of the Newton method. This is most likely related to the species mass fractions that are highly sensitive to minor

variations especially when approaching zero. For steady reactive flow calculations, a tolerance of 0.01 was typically required to avoid solution stall; however, for unsteady calculations this could be relaxed to a tolerance of 0.05 as the solution was "warm-starting" with a good initial estimate from the previous time step and the outer loop convergence was typically converged only a few orders of magnitude.

6.1.3 Local BILU Preconditioner Fill Level

In the variable-fill level BILU preconditioner acting as the local preconditioner for each partition as described previously in Section 4.2.4, the higher the level of fill used in constructing the approximate inverse, the more non-zero entries are retained at the cost of greater computational work and storage. As the level fill is increased, the approximate inverse approaches that of the exact inverse and further increasing the level of fill has no beneficial effect. To investigate the effects of the ILU(k) fill level, k, on performance for reacting flows, the steady co-flow methane-air laminar diffusion flame solutions described in Section 5.4 were re-computed for the two- and three- dimensional cases with varying fill levels k, from 0 to 4, with all other parameters remaining constant. The results for two different mesh resolutions in both two- and three-dimensions are summarized in Tables 6.1 and 6.2, respectively. For both mesh sizes, the best overall performance was achieved with a fill level, k, of 2 in 2D and 1 in 3D. In 2D, the overall effect on solution time is very minor, most likely due to the block sizes being fairly small. In 3D, the selection of fill level, k, does have a more significant impact on overall solution time than in 2D, however the effect is still relatively minor. The higher fill levels do reduce the total number of GMRES iterations required; however, not significantly enough to offset the extra computational cost of computing the preconditioner. Previous studies by Groth *et al.* [49] for 2D inviscid flows found that fill levels of $k = 3$ or $k = 4$ provided the best compromise; however, it appears that for 2D reactive flows, slightly lower levels of fill appear to be the most cost effective. In 3D, the lack of benefit from high ILU fill levels is consistent with other investigators [111, 201] and values of $k = 0$ and $k = 1$ appear to be most optimal.

Mesh size	ILU(k)	GMRES Iterations	NKS Iterations	CPU time (min)
3072 cells	0	42,933	3,241	12.47
	1	42,211	3,221	12.42
	2	40,115	3,196	12.35
	3	40,115	3,196	12.46
	4	40,115	3,196	12.52
12288 cells	0	57,368	5,027	40.70
	1	56,857	5,015	40.58
	2	55,513	4,965	40.05
	3	53,655	4,951	40.48
	4	52,964	4,998	40.93

Table 6.1: Convergence parameter results for varying the ILU(k) fill for the solution of a two-dimensional steady co-flow laminar diffusion flame with 96 (4×8) block, 3,072 cell and 96 (8×16) block, 12,288 cell meshes on 96 processor cores.

Mesh size	ILU(k)	GMRES Iterations	NKS Iterations	CPU time (min)
64,512 cells	0	25,394	2,720	143.02
	1	23,141	2,683	138.25
	2	22,268	2,716	141.56
	3	21,077	2,714	149.08
	4	20,254	2,714	164.94
217,728 cells	0	35,150	3,666	577.73
	1	31,873	3,669	567.75
	2	31,117	3,668	587.13
	3	28,922	3,586	592.43
	4	28,252	3,673	659.81

Table 6.2: Convergence parameter results for varying the ILU(k) fill for the solution of a three-dimensional steady co-flow laminar diffusion flame with 126 ($8 \times 8 \times 8$) block, 64,512 cell and 126 ($12 \times 12 \times 12$) block, 217,728 meshes on 126 processor cores.

6.1.4 Domain Overlap for Additive Schwarz Global Preconditioner

Domain overlap can help to offset the loss of overall implicitness of the Newton iterative solver introduced by the block-based additive Schwarz preconditioning. In the early stages of the parallel implicit algorithm development, Schwarz overlap was investigated in the implementation of the two-dimensional solution method. Overlaps ranging from zero to three were investigated using the bump flow cases outlined in Section 5.1.3 of Chapter 5. The increased overlap was found to reduce the number of GMRES iterations as expected; however, the overall CPU time was not reduced, and in fact was found to increase. The computational cost of solving the larger individual sub-domain blocks and the extra communication cost was not offset significantly enough by the reduction in iterations to ultimately reduce overall computation time. Gropp *et al.* also found no significant overall performance benefit from the use of domain overlap [111]. For the three-dimensional solution method, domain overlap was not investigated as the communications costs are generally even higher than for the two-dimensional case and little benefit could be achieved. As such, additive Schwarz preconditioning with no overlap has been used in all of the three-dimensional computations reported herein.

6.1.5 Summary of NKS Algorithm Parameter Selection

The specific NKS parameters used for performing the non-reactive and reactive flow problems Chapter 5 have been presented previously. The selection of these parameters were based on the investigations outlined in the preceding sections. Table 6.3 provides a summary of the key parameters found here to work best for the computation of the range of laminar reactive flows considered in this thesis.

6.2 Steady Solution Performance Compared to Explicit Time-Marching

As one of the primary goals of implementing the parallel NKS solver was to reduce the time required to achieve an accurate solution, this section compares the convergence

	2D Steady	3D Steady	2D Unsteady	3D Unsteady
GMRES Tolerance (η)	0.01	0.01	0.05	0.05
ILU Fill Level (f)	2	1	2	1
Newton Tolerance	1×10^{-6}	1×10^{-6}	1×10^{-2}	1×10^{-2}
Jacobian Update	every iteration	every iteration	1st iteration	1st iteration

Table 6.3: Summary of NKS parameters for steady and unsteady laminar reactive flows.

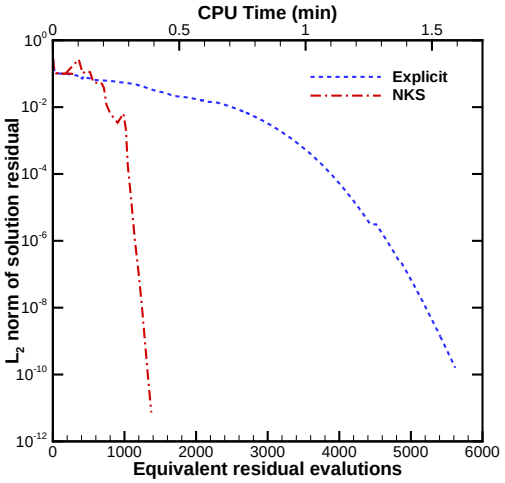
rates and total wall-clock times of the NKS solver versus an explicit 4-stage optimally smoothing solver that has been used extensively in previous work described by Northrup and Groth [63] and Gao *et al.* [14].

The comparisons were done using five of the representative inviscid, viscous, and reacting cases described in Chapter 5. Both solution methods use the same spatial discretization reconstruction and block based AMR approach, i.e., the same right hand side, the only difference between the simulations being the time-stepping algorithm. The CFL number used for the explicit solution was 0.5 and the NKS parameters are described for each case in Chapter 5. Both cases were run using the same HPC system, as described in Section 3.3.4 of Chapter 3 with the identical number of processor cores.

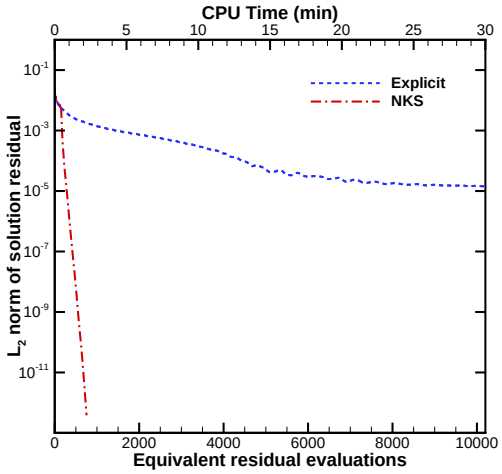
Figures 6.1 and 6.2 compare the convergence histories of density versus overall CPU time and equivalent residual, right hand side evaluations, for each of the five cases. The convergence histories for the other flow variables are similar. The use of equivalent residual evaluations provides a means for non-dimensionalizing the convergence history plots, allowing easier comparison of different solution methods across various computer platforms. The number of residual evaluations should remain relatively constant across differing systems whereas straight CPU time will vary considerably depending on CPU generation, model, and architecture.

As both the explicit and implicit solution methods were carried out using the same compute resources the absolute numbers for the computational cost are not of critical importance, but rather the relative convergence rates. When examining the convergence histories of Figures 6.1 and 6.2 it is readily evident that in all cases the NKS solver significantly outperforms the explicit method by reducing the time to achieve a solution dramatically, as should be expected for a steady flow problem. The improvement ranges

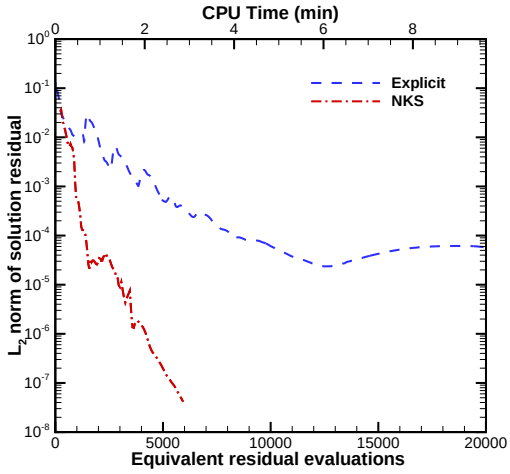
from 5 to 10 times reduction in solution time and in the viscous and reacting solutions the NKS solver was able to converge solution error to a far higher tolerance than the explicit methods that generally "stalled" after only a few orders. As mentioned in Section 6.1, the NKS parameters used were not explicitly optimally tuned for these specific cases, so these significant performance gains can be considered to be relatively conservative.



(a) 2D Inviscid Bump Flow

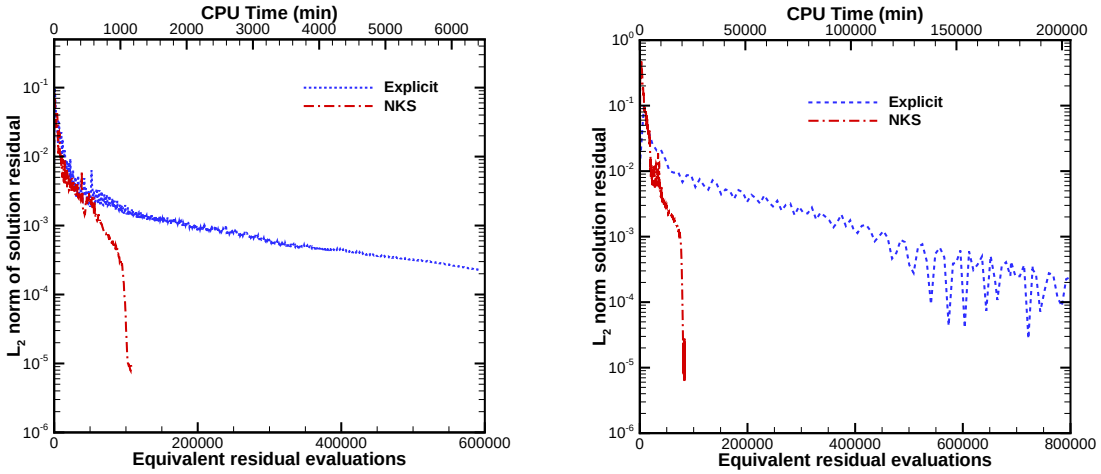


(b) 2D Viscous Flat Plate



(c) 1D Premixed Flame

Figure 6.1: Comparison of convergence of the solution residuals for the Newton-Krylov-Schwarz and explicit 4-stage optimally smoothing scheme as applied to; (a) 2D inviscid bump flow (Section 5.1.3), (b) 2D viscous flat plate initial mesh (Section 5.2.1), (c) 1D premixed flame (Section 5.3).



(a) 2D Laminar Diffusion Flame

(b) 3D Laminar Diffusion Flame

Figure 6.2: Comparison of convergence of the solution residuals for the Newton-Krylov-Schwarz and explicit 4-stage optimally smoothing scheme as applied to; (a) 2D laminar diffusion flame initial mesh (Section 5.4.2), (b) 3D laminar diffusion flame initial mesh (Section 5.4.2).

6.3 Unsteady Solution Performance Compared to Explicit Time-Marching

The performance of the proposed time-accurate parallel implicit BDF2-NKS algorithm has also been compared with two contemporary time-marching schemes, the results of which are now discussed. In particular, the parallel implicit scheme was compared to an explicit two-step, Runge-Kutta, time-marching scheme, RK2, without the temporal low-Mach-number preconditioning applied as that would break time accuracy, as well as the same BDF2 implicit time scheme implemented within a dual-time stepping procedure using an explicit multi-stage optimally-smoothing scheme as the pseudo-time stepping method, referred herein as BDF2-DTS. The time-step of the Runge-Kutta scheme is limited by the CFL condition, whereas the implicit BDF2 schemes are not. Nevertheless, the solution accuracy of the implicit methods are affected by the inner loop convergence tolerance.

Unlike for steady problems, where a direct comparison of convergence histories shows the

algorithms performance, as shown in Section 6.2, for time-accurate problems the only fair comparison is to compare overall solution CPU time at an equivalent solution accuracy for the same simulation or problem. The methodology used herein for comparison is similar to that used by Tabesh *et al.* [78] when comparing computational costs of various time-marching methods.

The performance is investigated using the solution of the 3D driven laminar diffusion flame problem as discussed previously in Section 5.4.3 of Chapter 5. As the solution is periodic it provides a rather good test case for comparing the time-marching schemes. The accuracy of the solutions is assessed by comparing the solution of the axial component of the center-line velocity, w, over one full oscillation, which for this case is 0.05 s. As there is no analytical solution for this case, the solution error, w_{error} is calculated with reference to a computed solution, $w_{i,ref}$ shown in Figure 6.3, calculated using a very small time step. Equation (6.1) shows the error calculation where N is the total number of time steps.

$$w_{error} = \sqrt{\frac{\sum_{i=0}^{N} (w_i - w_{i,ref})^2}{N}} \tag{6.1}$$

A 126 (8×8×8 cell) block mesh, as shown in Figure 5.18(b) with 64,512 computational cells was used without any dynamic adaption to ensure consistent solutions with the different time-marching schemes. Initially the problem was run for 10 full periods (0.5 s) to remove any initial condition hysteresis effects and ensure a period solution was achieved. The BDF2 schemes were run with physical time steps, Δt, of 0.1, 0.05, 0.025, 0.01, and 0.005 ms corresponding to 500, 1000, 2000, 5000, and 10000 steps per period respectively. The RK2 solutions were run with CFL's of 0.3, 0.2 and 0.1, as 0.3 was found to be the largest stable value for this method. For the NKS-BDF2 a Newton tolerance of 0.01 with a GMRES tolerance of 0.01 was used.

The performance results depicted in Figure 6.4 provide a comparison of the various algorithm solution error, w_{error}, versus the associated computational time. It is clear that the proposed BDF2-NKS algorithm outperforms the other methods requiring considerably less computational time to achieve the same solution accuracy. The RK2 scheme is considerably handicapped by having to take very small time steps to maintain stability (i.e., there is a limited range of valid time steps) and also not being able to utilize the low-

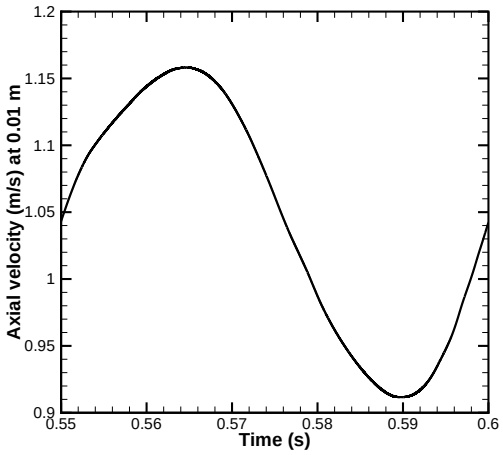


Figure 6.3: Time-accurate driven 3D laminar diffusion flame centerline velocity w at z=0.1 m for one period at 20 Hz oscillation (0.05 s).

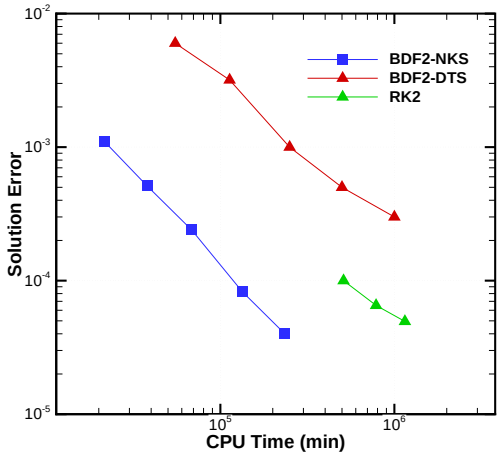


Figure 6.4: Unsteady algorithm performance comparisons of BDF2-NKS, BDF2-DTS, and RK2 for the solution of a time-accurate 3D driven laminar diffusion flame.

Mach-number preconditioning of the numerical flux to maintain solution accuracy. The BDF2-DTS scheme does benefit from the greater stability of the implicit time scheme allowing it to use much large time-steps; however, it requires a far larger number of inner loop iterations of the multi-stage optimally smoothing scheme which completely offsets any possible gains in computational savings. Much like the performance of the steady solutions, the unsteady NKS solver outperforms the other methods by significantly reducing the number of residual evaluations that are required. For the reactive flow cases, the cost of the residual evaluation is very high due to the thermally perfect gas relationships and the chemical source terms. Thus the residual evaluation cost dominates over the extra costs associated with the NKS, i.e., preconditioner and GMRES overhead, resulting in the very significant performance gains in overall solution time.

6.4 Parallel Performance

In modern HPC, the parallel performance and scalability of a numerical methods is an extremely important consideration. Since single core serial CPU performance has

remained relatively flat due to power consumption issues, manufacturers have turned to multi-core and hybrid architectures to provide increased computational performance. This has resulted in modern HPC clusters having ever increasing core counts ranging from the thousands to over a million in emerging multi-petaflop systems [113, 202]. Thus an algorithms ability to scale to a large number of processor elements is essential in order to use modern HPC systems efficiently and effectively especially for large and computationally expensive problems.

As such, the parallel performance of the proposed algorithm is assessed herein by examining two common metrics, the parallel speedup, S_p , and efficiency, E_p, which are defined as

$$S_p = \frac{t_1}{t_p} \tag{6.2}$$

$$E_p = \frac{S_p}{p} \tag{6.3}$$

where t_1 is the total wall time to solve the problem with 1 processor, and t_p is the total wall times required to solve the problem with p processors. These performance measures are examined for both strong and weak scaling problems. The effect of the Schwarz preconditioner is also investigated as, in this implementation, it is directly tied to the parallel domain decomposition and the amount of preconditioning increases with the number of processing cores used to solve the problem at hand.

6.4.1 Strong Scaling of Parallel Implicit Algorithm

Strong scaling determines an algorithm's ability to scale proportionately with more processors for a fixed size problem. For strong scaling, the problem size is held fixed while the number of processors used to perform the computation is varied. Ideally, if the number of processors used to solve the problem is increased by a factor of 10, then the solution time should be reduced by the same factor of 10. Strong scaling is also typically the most challenging for most algorithms as the ratio of communication to computation increases as the number of processor cores increases. This is of course the inverse of what typically promotes good parallel scalability.

The strong scaling performance of the proposed parallel AMR implicit finite-volume algorithm was examined using the solution of the 3D steady laminar diffusion flame described previously in Section 5.4. A non-adapted mesh of 6,400 ($8 \times 8 \times 8$ cell) blocks totaling 3,276,800 cells with a fixed number of Newton iterations was used to ensure that equivalent work was used for each solution. As the Schwarz preconditioner in this implementation is tied to the domain decomposition, i.e., the number of blocks, using the same number of blocks regardless the number of processors, maintains the same level of Schwarz preconditioning. This allows one to investigate the algorithms parallel scalability separate from the effects of the Schwarz preconditioner, which will be investigated independently in Section 6.4.3 of Chapter 6 to follow.

Scaling studies were performed on three separate HPC platforms to determine the effects of the processor, network, and algorithm on the scalability. Figure 6.5(a) hows the results of the proposed NKS algorithm on both DDR Infiniband and Gigabit Ethernet on the SciNet Intel x86_64 cluster using up to 800 nodes which have 8 cores each. At 6,400 cores the algorithm is still achieving 80% efficiency for Infiniband and remarkably, even with the much higher latency Ethernet, achieving almost 50% for this particular laminar reactive flow problem.

Figure 6.5(b) shows the scaling results from running on the SciNet IBM Power6 Cluster with DDR Infiniband using up to 100 nodes with 64 threads per node. The same 3D steady laminar diffusion flame case was solved using the NKS algorithm as well as the explicit scheme described in the convergence comparisons in Section 6.2. The parallel efficiency of the NKS algorithm is very similar to that obtained on the Intel Infiniband cluster, which is to be expected as the network performance of the two clusters is very similar. The explicit solution algorithm achieves an efficiency of over 70% at 6,400 cores; however, is not quite as efficient as the implicit NKS algorithm. This difference can be attributed to the NKS requiring much less, but more computational expensive, iterations than the explicit solution method. Thus the ratio of computation to communication favors the NKS algorithm, increasing the overall strong scaling efficiency.

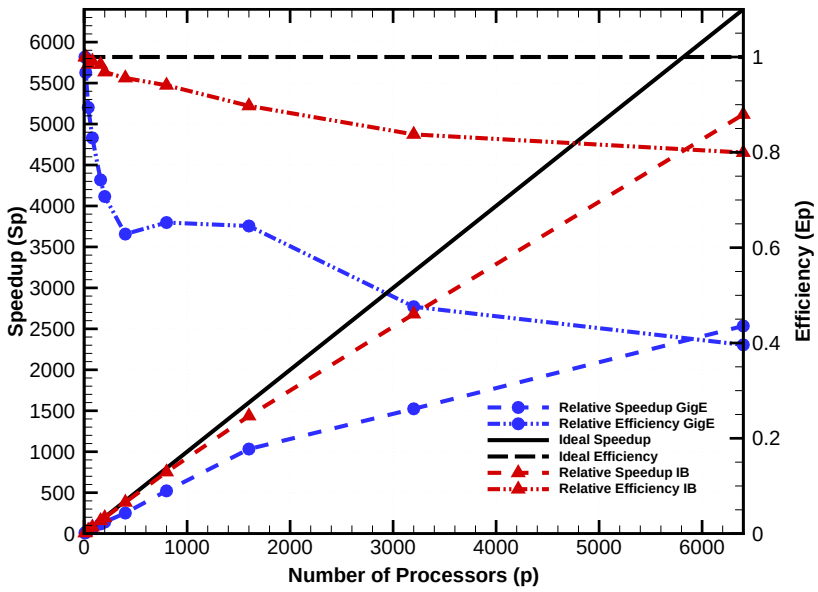
Scaling studies were also carried out on an IBM BlueGene/Q (BGQ) supercomputer [173, 174] which consists of 2,048 low-power 16 core CPUs (32,768 cores) connected together with a highly scalable proprietary 5D torus. The same 3D methane-air laminar diffusion flame case was used; however, the problem size was increased to 32,256 ($8 \times 8 \times 8$ cell)

blocks totaling 16,512,072 cells. Figure 6.6 shows the very impressive strong scaling of the NKS implicit algorithm achieving almost 95% efficiency at 32,256 cores. The high dimensionality of the BGQ 5D Torus results in very low point-to-point latencies allowing the algorithm to scale extremely well, even with a very high number of cores.

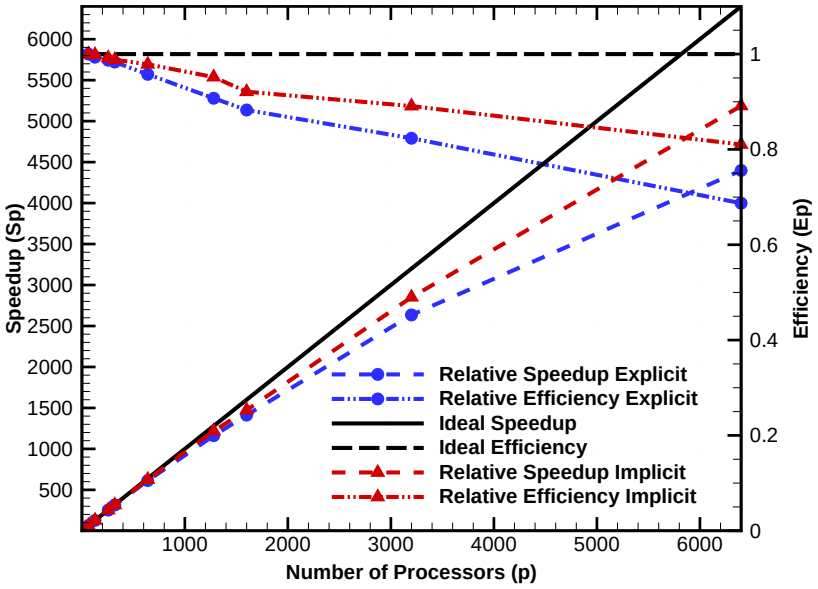
6.4.2 Weak Scaling of Parallel Implicit Algorithm

Weak scaling examines an algorithm's ability to scale as the problem size is increased in step with the number of processing cores. For a weak scaling analysis, the problem size is scaled proportionately to the number of processors and in the ideal case the computation time would remain fixed or constant. This is generally a more practical scaling test as it represents the typical case were more processors are required to solve larger and larger problems in a reasonable amount of time. For most solution methods, it is usually easier to achiever high performance for weak scaling than for strong scaling as the communication and work per processor core remains essentially constant.

In the weak scaling evaluations of the parallel implicit AMR finite-volume method, the 3D steady laminar flame solution was again considered. However, the mesh is no longer constant but scales linearly with the number of processor cores being used. The mesh is adjusted so as to maintain one ($8 \times 8 \times 8$ cell) block per core starting with 25 blocks (12,800 cells) through 6,400 blocks (3,276,800 cells). Figure 6.7 shows the weak scaling results for the parallel implicit algorithm as well as an explicit scheme described in the convergence comparisons in Section 6.2 on the SciNet Intel x86_64 cluster using DDR Infiniband. The parallel implicit algorithm achieves a very high relative efficiency of over 90% even up to 6,400 cores and the explicit case also performs well achieving a relative efficiency of over 80% for 6,400 cores. The weak scaling results are consistent with the strong scaling results of Section 6.4.1, with the parallel implicit algorithm outperforming the explicit case, which is expected as the ratio of communication to computation favors the implicit scheme. The weak scaling relative efficiencies are higher than those of the strong scaling, but again this is expected as the communication and work per core ratio is constant in weak scaling, whereas for the strong scaling cases, inter-core communication increases.



(a) Intel x86 Cluster



(b) IBM Power6 Cluster

Figure 6.5: Strong scaling parallel performance of the Newton algorithm for laminar diffusion flame calculation on a mesh consisting of 6,400 $8 \times 8 \times 8$ cell solution blocks (3,276,800 cells) showing the relative parallel speed-up on an (a) Intel x86 Ethernet and DDR Infniband Cluster and (b) IBM Power6 with DDR Infiniband Cluster.

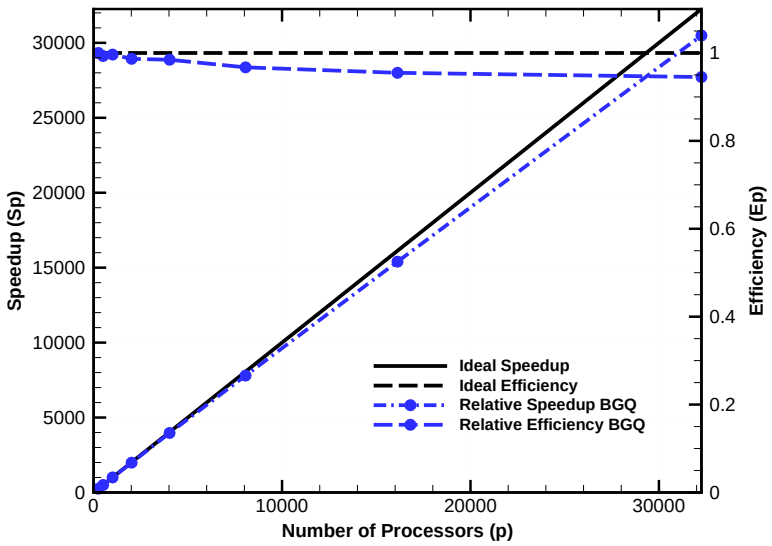


Figure 6.6: Strong scaling parallel performance of the Newton algorithm for laminar diffusion flame calculation on a mesh consisting of 32,256 $8\times8\times8$ cell solution blocks (16,515,072 cells) showing the relative parallel speed-up on a Blue Gene/Q supercomputer.

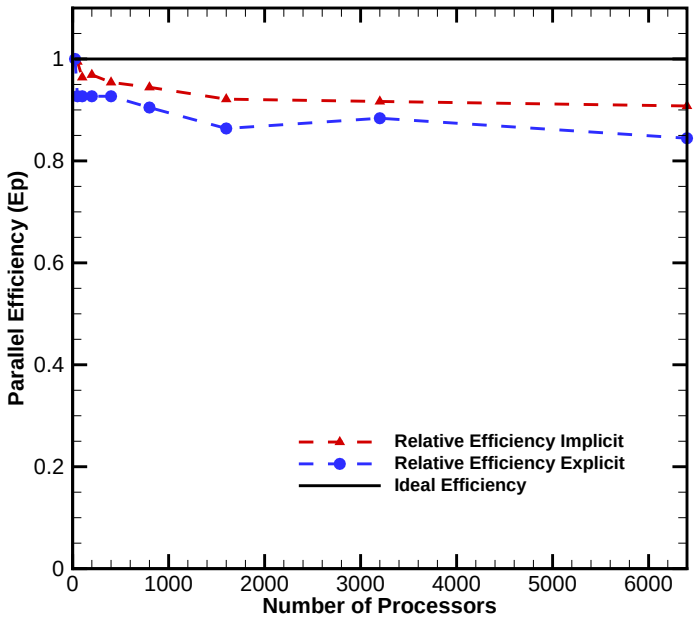


Figure 6.7: Weak scaling parallel performance of the Newton algorithm for laminar diffusion flame calculation on a mesh consisting of $8\times8\times8$ cell solution blocks, with 1 block per processor core.

6.4.3 Effects of Additive Schwarz Preconditioning

The strong parallel scaling simulation results described above in Section 6.4.1 were performed using a fixed number of iterations and a fixed domain decomposition, thus the number of total blocks were held constant, regardless the number of processor cores being used. These results portray accurately the algorithms ability to scale well with the increases in communication; however, they do not assess or include the expected degradation in performance of the parallel implicit algorithm caused by the loss of coupling or implicitness resulting from the domain decomposition procedure and Schwarz preconditioning. Typically the decomposition changes as the number of processors are increased as further subdivisions of the domain are required. For explicit calculations this typically does not change the underlying algorithm or convergence rates, it just adds the increased overhead involved with communication at the boundaries. However in the proposed parallel implicit algorithm, where the domain decomposition procedure is also used as the global Schwarz preconditioner, the algorithms convergence can be deleteriously affected by the decomposition. As more blocks are used, the less accurate the approximate inverse that results from the Schwarz preconditioner. As a result, typically a greater number of GMRES iterations to converge the problem are required, resulting in a higher computational cost. This was well illustrated in previous work by Groth *et al.* [49] for 2D inviscid flow problems.

As the Schwarz preconditioner is tied to the domain decomposition in this implementation, the effect of the preconditioner can be investigated by adjusting the partitioning (i.e. the number of blocks) for a mesh with a fixed number of cells. As the number of blocks increases the number of cells per block decreases and thus the global Schwarz preconditioner subdivides the global linear system into smaller and smaller local systems as shown graphically in Figure 4.1 of Chapter 4.

Steady Two-Dimensional Supersonic Flow Past a Cylinder

The effect of the Schwarz preconditioner is first investigated for the solution of inviscid two-dimensional supersonic flow past a cylinder as described previously in Section 5.1.2 of Chapter 5. Numerical solutions were calculated on a 128×128 mesh consisting of 16,384 computational cells and several different blockings (partitionings) of the computational

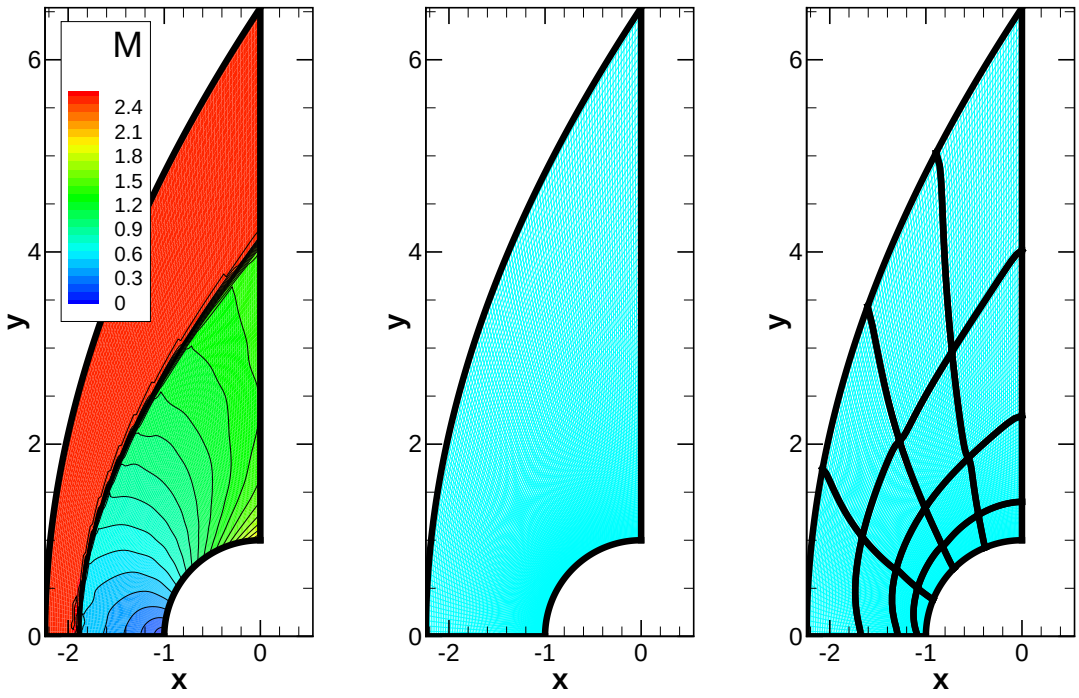
domain are examined in order to investigate the influence of the Schwarz preconditioning. A single-block grid consisting of one 128×128 solution block is considered, along with 4-, 16-, and 64-block grids composed of 4 64×64, 16 32×32, and 64 16×16 solutions blocks. The computed Mach number distribution for the $M_\infty = 2.5 = 2.5$ blunt-body flow is shown in Figure 6.8(a) along with single- and 16-block computational meshes in Figures 6.8(b) and 6.8(c) respectively for comparison. The convergence results of the parallel NKS algorithm are given for the 1-, 4-, and 16-block cases in Figure 6.9(a), showing the effects that the higher level of Schwarz preconditioning has on solution convergence and thus computational time required. The single block with no Schwarz preconditioning converges the most rapidly and as the Schwarz preconditioning is increased from 4 to 16 blocks the amount of time to reach convergence increases by approximately 30%. Figure 6.9(b) shows that the algorithm still maintains good parallel scaling, even for the higher number of block partitions and despite the fact that the ratio of communication to computation is less favorable in this relatively small two-dimensional problem.

Subsonic Two-Dimensional Laminar Boundary-Layer Flow Past a Flat Plate

The effect of the Schwarz preconditioning is next investigated for the solution of two-dimensional viscous flow over a flat plate on a 64×128 (8,192 cell) grid, as described previously in Section 5.2.1 of Chapter 5. Figure 6.10(a) shows the 3 different grid partitions with 3, 12, and 48 blocks, and Figure 6.10(b) shows the effects of the higher level of Schwarz preconditioning has on solution convergence and thus computational time required. For this case, the number of GMRES iterations required increases about 10% between 3 and 12 blocks and overall solution time about 40%. From 12 to 48 blocks, the effect is much less substantial suggesting that the increased Schwarz preconditioning has less of an impact on solution time as the number of blocks is increased. Figure 6.10(c) shows that the algorithm still maintains good parallel scaling, even for the higher number of block partitions, even though the ratio of communication is computation is less favorable in this relatively small two-dimensional problem.

Three-Dimensional Co-Flow Laminar Diffusion Flames

Finally, to fully asses the influence of the Schwarz preconditioning and the resulting degradation in overall performance a more representative combustion case, the solution

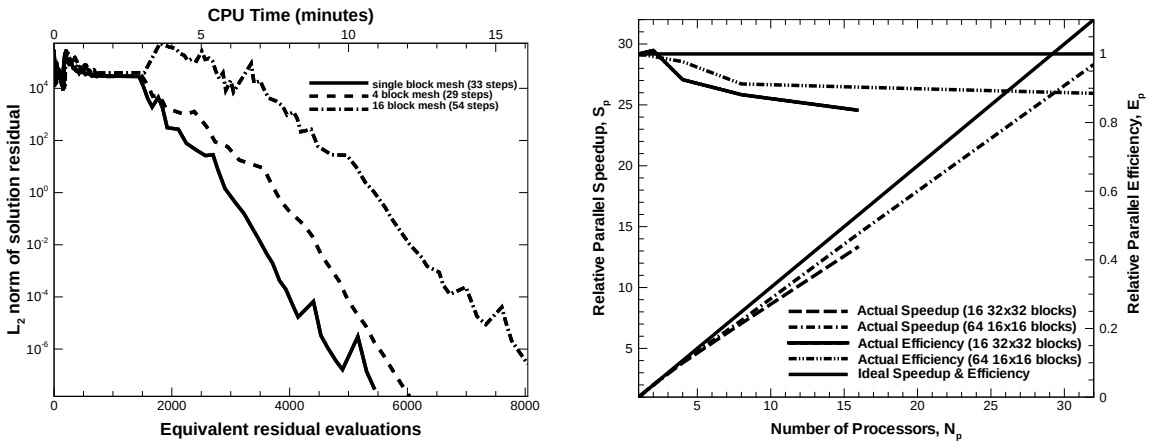


(a) (b) (c)

Figure 6.8: Computed solution of parallel NKS algorithm for supersonic flow past a circular cylinder; Mach number $M_\infty = 2.5$; showing (a) Mach number distribution and (b) single (128×128) and (c) 16 (32×32) block, 16,384 cell, meshes used in computations.

of a steady three-dimensional co-flow methane-air laminar diffusion flame, as described previously in Section 5.4.2, is re-investigated. The overall solution was converged four orders of magnitude using a GMRES tolerance of 0.01 and ILU(0). An initial 13 ($12 \times 12 \times 72$ cell) block grid with 134,784 cells total is subdivided into 26, 52, 104, 208, 312, and 624 blocks as shown in Figure 6.11 with the 624 block portioning having only $6 \times 6 \times 6$ cells per block. All other algorithm and grid parameters, besides the initial block partitioning, were held constant.

Figure 6.12(a) shows the convergence histories for all 7 cases computed using 1 block per processor core. As is expected the 13-block case, with the least amount of Schwarz preconditioning, converges the most rapidly and the 624 block case, with the most partitions converges most slowly. The spread however between these two extremes is only about 20%, which is much better than 50% seen in the previous two-dimensional cases. When

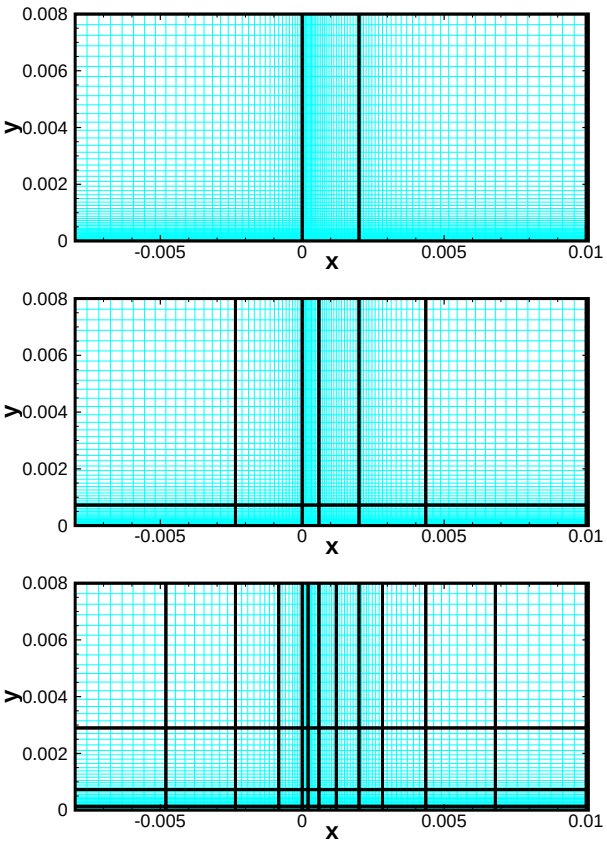


(a) (b)

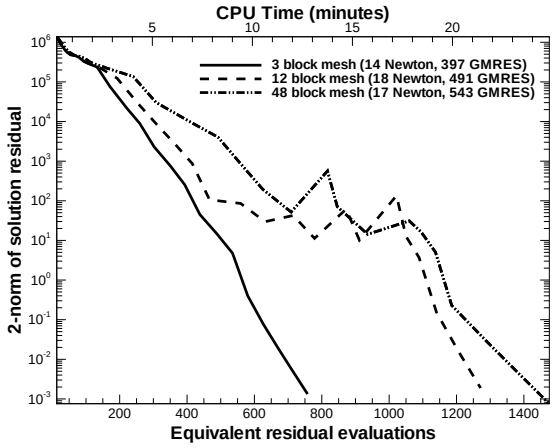
Figure 6.9: Performance of parallel NKS algorithm for supersonic flow past a circular cylinder; Mach number M = 2.5; (a) convergence history showing 2-norm of density residual as a function of the number of equivalent residual evaluations and total processor time and (b) strong parallel scaling.

looking at the solver statistics given in Table 6.4, the overall NKS iterations are relatively constant; however the total number of GMRES iterations increases by about 20%. This is consistent with what is expected. As the global GMRES problem becomes less well conditioned with greater subdivision of the problem, a greater number of iterations is required to achieve a specified convergence tolerance. Similar trends were also observed in the previous two-dimensional cases. Despite these effects, Figure 6.12(b) shows the strong parallel scaling for this case. Even with the combined effects of the Schwarz preconditioner and parallel communication overhead, the parallel efficiency with 624 cores remains at just under 80%.

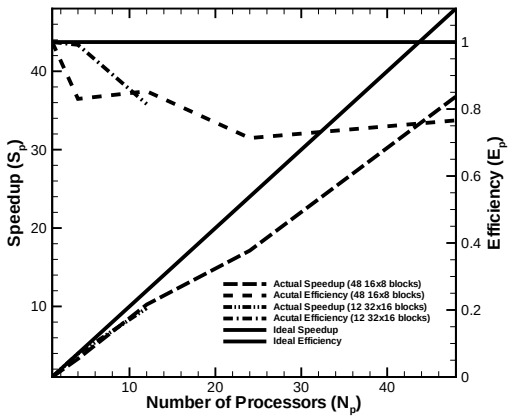
In summary, it would seem that overall the use of Schwarz preconditioning has a less detrimental effect on solution performance for the 3D reactive case than for the 2D inviscid and viscous flow cases. The two-dimensional inviscid and viscous cases started with very little Schwarz preconditioning as they were only initially partitioned with one and three blocks respectively, while the minimum number of partitions for the three-dimensional case was 13. Unfortunately, due to the topologies of the 3D grids considered, 13 is the minimum that could be accommodated. Also the 3D reactive flow case has a much more expensive residual evaluation enabling greater parallel efficiency, due to the increased computation to communication ratio, ultimately reducing the overall time to



(a)

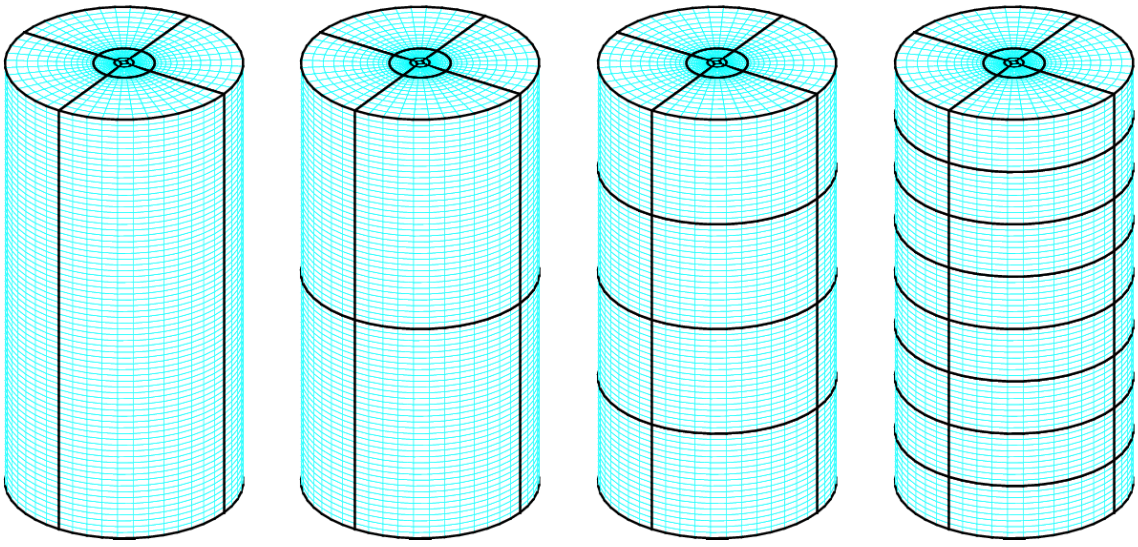


(b)

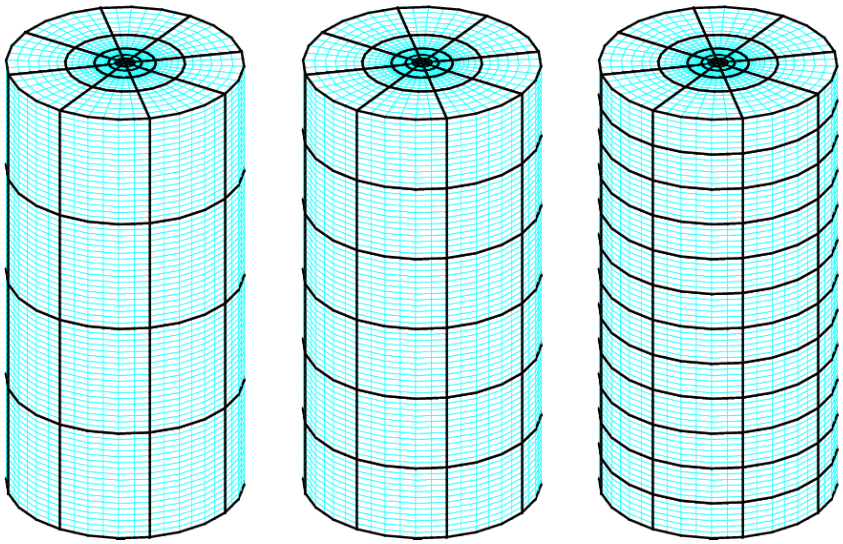


(c)

Figure 6.10: Viscous flow over a two-dimensional flat plate computed on the same 64×128 mesh with (a) 3 different domain decompositions (3, 12, and 48 blocks) showing the effects of Schwarz preconditioner on (b) solution convergence and (c) strong parallel scaling.



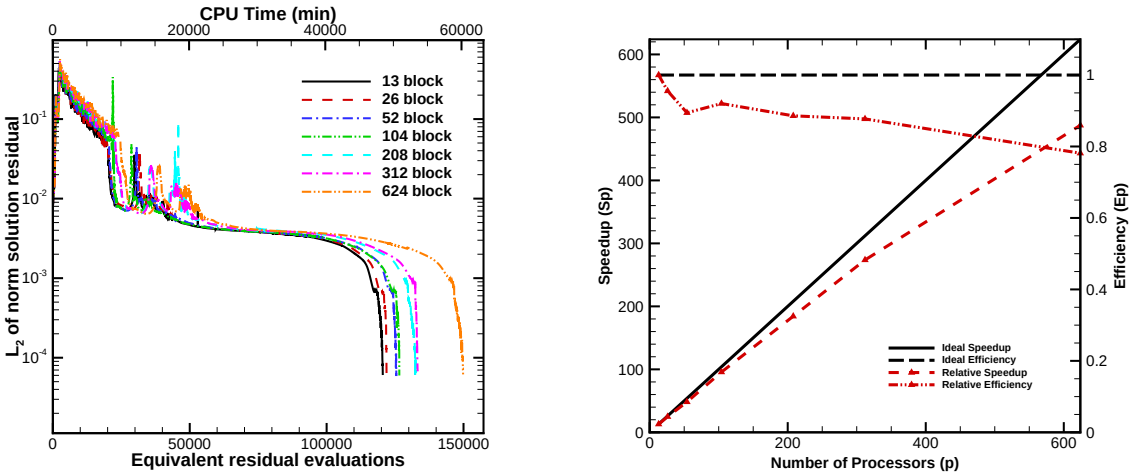
(a) 13 blocks (b) 26 blocks (c) 52 blocks (d) 104 blocks



(e) 208 blocks (f) 312 blocks (g) 624 blocks

Figure 6.11: 3D laminar diffusion flame 134,784 cells mesh partitioned (a) 13, (b) 26, (c) 52, (d) 104, (e) 208 , (f) 312, and (g) 624 blocks.

achieve a solution. While it is felt that the results are quite representative of typical combustion flows it would seem important to confirm these findings for a wider range of reactive flow problems and geometries in future follow-on studies.



(a) Convergence histories

(b) Strong scaling

Figure 6.12: 3D Steady Laminar Diffusion flame computed with 134,784 cells and partitioned with 13, 26, 52, 104, 208, 312, and 624 blocks showing (a) the effect of the Schwarz preconditioner on convergence rate and (b) the associated strong parallel scaling.

Blocks	NKS Iterations	GMRES Iterations	CPU time (min)	Wall time (min)
13	4,501	37,155	48,446	3,726
26	4,521	37,548	48,988	1,884
52	4,573	37,622	50,421	969
104	4,552	38,105	50,871	489
208	4,596	41,313	53,186	254
312	4,567	42,625	53,572	171
624	4,640	46,527	60,236	96

Table 6.4: Summary of NKS algorithm solution statistics for the same 134,784 cells mesh with varying levels of Schwarz preconditioning determined by the number of blocks.

Chapter 7

Conclusions

As stated in the introduction of Chapter 1 of the thesis, the primary objective of this research was the development of a numerical method that will significantly reduce the time it takes to achieve accurate solutions of physically complex steady and unsteady laminar reactive flows. This was achieved by developing a new parallel implicit AMR scheme and applying it to the solution of a range of steady and unsteady methane-air laminar diffusion and premixed flames.

The parallel implicit formulation makes use of a dual-time-stepping like approach with an implicit second-order backward discretization of the physical time. A Jacobian-free inexact Newton method with a preconditioned generalized minimal residual (GMRES) algorithm is used to solve the system of nonlinear algebraic equations arising from the temporal and spatial discretization procedures. An additive Schwarz global preconditioner is used in conjunction with block incomplete LU type local preconditioners for each sub-domain. This Newton-Krylov-Schwarz algorithm was developed to work in conjunction with dynamic solution-directed block-based mesh adaptation according to physics-based refinement criteria. The underlying block-based octree data structure enables an efficient parallel implementation via domain decomposition.

The algorithm, in conjunction with a density-based solution method in which the fully compressible form of the Navier-Stokes equations are solved in a tightly-coupled manner, was applied to the solution of steady and driven laminar diffusion flames. The use of the compressible form of the Navier-Stokes equations naturally allows for the large density variations associated with reactive flows. It also means that, unlike many other

approaches, the solution method is well prepared for the accurate treatment of flows with time varying pressure and the prediction of thermoacoustic phenomena. Solutions were obtained for both 2D axisymmetric and 3D coordinate frames and good agreement was demonstrated between the two sets of results and with other published experimental and numerical results. The application of low-Mach-number preconditioning was found to be necessary and important for controlling dissipation in low-Mach-number flows. Premixed time-accurate methane-air laminar flames were also investigated and the proposed parallel implicit AMR finite-volume method did a very good job of predicting the frequency of buoyancy-driven oscillations that occur under terrestrial gravity. The prediction of shedded or flame-flicker vortices for the unsteady premixed flame showcased the ability of the solution directed adaptive mesh refinement to track and resolve detailed flow features.

The parallel implicit algorithm's performance was assessed and was found to drastically reduce the computational time it takes to achieve a solution in all cases investigated compared to standard explicit time marching schemes. For steady flows, the computational time required to convergence a solution was reduced by a factor 5-10 times. Additionally time-accurate combustion solutions were obtained 6 times faster than comparative schemes at the same level of accuracy. Parallel communication performance was assessed and scaling efficiencies over 80% were achieved at over 6,000 cores. The Schwarz preconditioner was investigated and found to decrease the algorithm's performance as the amount of preconditioning, greater and smaller blocks, is increased. For 3D reactive flows, the reduction however was not as severe as with the 2D model problems and any loss in efficiency was found to be much less than the benefits of being able to solve the problem of interest using more processing cores, at least for the cases considered.

The combination of the time-accurate parallel implicit algorithm with dynamic AMR did a very good job at efficiently solving laminar combustion problems and fulfilling the primary research objective to significantly reduce the time it takes to achieve accurate solutions. It is felt that, with the proposed algorithm combined with today's available and future HPC resources, provide the opportunity to bring to bear unprecedented computational capabilities to problems related to turbulent reactive flow, including thermoacoustic phenomena and combustion instabilities.

7.1 Contributions

The following is a summary of original contributions arising as a result of this research:

- A 3D block-based AMR framework was extended and adapted to allow for dynamic refinement and coarsening of the mesh for tracking the solution features of unsteady flows.

- A new parallel implicit Newton-Krylov-Schwarz algorithm for 2D and 3D unsteady reactive laminar flows governed by the fully compressible form of the Navier-Stokes equations was developed that is fully compatible with the aforementioned AMR scheme.

- The first application of the combination of a fully parallel implicit Newton-Krylov-Schwarz algorithm with adaptive mesh refinement for reacting flows was performed. Both steady and unsteady laminar diffusion and premixed flames were considered.

- The proposed algorithm was demonstrated to be considerably faster than a comparable explicit scheme for all steady and unsteady flow problems considered.

- The algorithm demonstrated the capability to solve very large problems and scale to well over 6,000 processor cores. The Schwarz preconditioner was shown to have an effect on scaling, but still have good parallel scaling performance of 80% at over 600 processing cores for 3D combustion solutions.

7.2 Recommendations for Future Work

The parallel implicit block-based AMR scheme developed in this thesis provides a solid basis going forward for future research related to both algorithm design and development as well as combustion research. The overall approach is very promising; however, detailed investigation into algorithm components could still provide significant reductions in solution times and possibly increase overall robustness. A few key ares of further investigation are outlined below that would be natural extensions to this research.

Globalization

A common issue when employing Newton-Krylov methods for the solution of steady-state problems is the issue of globalization whereby a good startup algorithm is invariably required in order to increase the radius of convergence and ensure global convergence of the NKS method. [80, 92, 94, 101]. This has been partially addressed in the thesis by using the SER methods outlined in Section 4.5. Nevertheless, for the steady reactive flow problems studied here, significant time was spent in this startup phase. Future investigation into alternative strategies, such as the work by Zingg and co-researchers [97, 203, 204], would likely lead to greater robustness and improved performance.

Preconditioners

As mentioned by Knoll and Keyes [101], the overall efficiency of Newton-Krylov schemes is predominantly determined by the choice of preconditioner. For the parallel Newton-Krylov method adopted herein this is even more pronounced. The combination of a global Schwarz and local ILU(f) preconditioner has worked well for the situations considered here; however, there was still a measurable degradation caused by the Schwarz preconditioner, as shown in Section 6.4.3 of Chapter 6.

It is recommended that future research include investigations into the use of multigrid as a preconditioner [205, 206] as well as multi-level approaches such as those proposed by Cai *et al.* [207, 208] to counteract the effect of the Schwarz preconditioning.

Higher Order Schemes

For unsteady time-accurate flows, the BDF2 temporal scheme was found to perform quite well when combined with the NKS algorithm. However, for fine solutions relatively small time steps were still required for stability so investigations into higher order temporal schemes may be of benefit. Tabesh and Zingg [78] found that this was indeed the case, when comparing BDF2 with a 4th order explicit single-diagonal implicit Runge-Kutta (ESDIRK) scheme for unsteady aerodynamic flows.

Along similar lines, the higher order spatial discretization employed was only second order and thus to resolve fine features, such as the flame fronts in the reactive flow cases,

a large number of computational cells were required even with the benefits of AMR. An investigation of higher order spatial discretizations, such as the those considered in the recent work of Ivan and Groth [209, 210], coupled with the parallel implicit AMR algorithm may lead to further reductions in overall time to achieve a solution.

AMR

The dynamic 3D adaption scheme has proved very effective at resolving the solution features of the problems solved in Chapter 5. Further efficiencies may be gained however from transitioning from an octree data structure where each block must be refined into 8 children blocks, to a binary tree that would allow non-isotropic direction-dependent refinement [51, 160, 161].

Turbulence and Detailed Chemistry

The combustion modelling used in this research was relatively unsophisticated. The study was restricted to simple reduced chemical kinetics and laminar flows. An obvious next step would be to extend the algorithm to solutions with detailed chemistry as well as extend it for use with turbulent combustion flow regimes. Both of these application areas are indeed already being studied as Charest *et al.* [120] used the parallel implicit AMR algorithm as the basis for the solution of detailed 2D axisymmetric sooting flames and Northrup *et al.* [211] has done some preliminary work extending the algorithm for Large Eddy Simulations (LES) of reactive flows.

References

[1] C. M. Benkovitz, M. T. Scholtz, J. Pacyna, L. Tarrason, J. Dignon, E. C. Voldner, P. A. Spiro, J. A. Logan, and T. E. Graedel. Global gridded inventories of anthropogenic emissions of sulfur and nitrogen. *Journal of Geophysical Research: Atmospheres*, 101(D22):29239–29253, December 1996.

[2] C. K. Westbrook, Y. Mizobuchi, T. J. Poinsot, P. J. Smith, and J. Warnatz. Computational combustion. *Proceedings of the Combustion Institute*, 30:125–157, 2005.

[3] A. M. Eaton, L. D. Smoot, S. C. Hill, and C. N. Eatough. Components, formulations, solutions, evaluation, and application of comprehensive combustion models. *Progress in Energy and Combustion Science*, 25:387–436, 1999.

[4] D. Veynante and L. Vervisch. Turbulent combustion modeling. *Progress in Energy and Combustion Science*, 28:193–266, 2002.

[5] H. Pitsch. Large-eddy simulation of turbulent combustion. *Annual Review of Fluid Mechanics*, 38:453–482, 2006.

[6] Los Alamos National Laboratory. KIVA. http://www.lanl.gov/orgs/t/t3/codes/kiva.shtml.

[7] OpenCFD Ltd (ESI Group). OpenFOAM. http://www.openfoam.com.

[8] M.D. Smooke, R.E. Mitchell, and D.E. Keyes. Numerical solution of two-dimensional axisymmetric laminar diffusion flames. *Combustion Science and Technology*, 67:85–122, 1989.

[9] M.D. Smooke. The computation of laminar flames. In *Proceedings of Combustion Institute*, volume 34, pages 65–98, 2013.

[10] B. A. V. Bennett, C. S. McEnally, L. D. Pfefferle, and M. D. Smooke. Computational and experimental study of axisymmetric coflow partially premixed methane/air flames. *Combustion and Flame*, 123:522–546, 2000.

[11] S. B. Dworkin, B. C. Connelly, A. M. Schaffer, B. A. V. Bennett, M. B. Long, M. D. Smooke, M. P. Puccio, B. McAndrews, and J. H. Miller. Computational and experimental study of a forced, time-dependent, methane-air coflow diffusion flame. In *Thirty First Symposium (International) on Combustion*, pages 971–978, Pittsburgh, 2007. The Combustion Institute.

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

[13] V. Moureau, G. Lartigue, Y. Sommerer, C. Angelberger, O. Colin, and T. Poinsot. Numerical methods for unsteady compressible multi-component reacting flows on fixed and moving grids. *Journal of Computational Physics*, 202:710–736, 2005.

[14] X. Gao, S. A. Northrup, and C. P. T. Groth. Parallel solution-adaptive method for two-dimensional non-premixed combusting flows. *Progress in Computational Fluid Dynamics*, 11(2):76–95, 2011.

[15] J. H. Chen. Petascale direct numerical simulation of turbulent combustion – fundamental insights towards predictive models. In *Proceedings of Combustion Institute*, volume 33, pages 99–123, 2011.

[16] D. E. Keyes. A science-based case for large-scale simulation volume 1. Report prepared for the Office of Science, U. S. Department of Energy, July 2003.

[17] D. E. Keyes. A science-based case for large-scale simulation volume 2. Report prepared for the Office of Science, U. S. Department of Energy, September 2004.

[18] S. K. Godunov. Finite-difference method for numerical computations of discontinuous solutions of the equations of fluid dynamics. *Matematicheskii Sbornik*, 47:271–306, 1959.

[19] M. J. Berger. Adaptive mesh refinement for hyperbolic partial differential equations. *Journal of Computational Physics*, 53:484–512, 1984.

[20] M. J. Berger. Data structures for adaptive grid generation. *SIAM Journal for Scientific and Statistical Computing*, 7(3):904–916, 1986.

[21] M. J. Berger and P. Colella. Local adaptive mesh refinement for shock hydrodynamics. *Journal of Computational Physics*, 82:67–84, 1989.

[22] M. J. Berger and R. J. LeVeque. An adaptive Cartesian mesh algorithm for the Euler equations in arbitrary geometries. Paper 89-1930, AIAA, June 1989.

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

[24] K. G. Powell, P. L. Roe, and J. Quirk. Adaptive-mesh algorithms for computational fluid dynamics. In M. Y. Hussaini, A. Kumar, and M. D. Salas, editors, *Algorithmic Trends in Computational Fluid Dynmaics*, pages 303–337. Springer-Verlag, New York, 1993.

[25] D. De Zeeuw and K. G. Powell. An adaptively refined Cartesian mesh solver for the Euler equations. *Journal of Computational Physics*, 104:56–68, 1993.

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

[27] J. J. Quirk and U. R. Hanebutte. A parallel adaptive mesh refinement algorithm. Report 93-63, ICASE, August 1993.

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

[29] A. S. Almgren, T. Buttke, and P. Colella. A fast adaptive vortex method in three dimensions. *Journal of Computational Physics*, 113:177–200, 1994.

[30] J.B. Bell, M.J. Berger, J.S. Saltzman, and M. Welcome. A three-dimensional adaptive mesh refinement for hyperbolic conservation laws. *SIAM Journal on Scientific Computing*, 15:127–138, 1994.

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

[32] R. B. Pember, J. B. Bell, P. Colella, W. Y. Crutchfield, and M. L. Welcome. An adaptive Cartesian grid method for unsteady compressible flow in irregular regions. *Journal of Computational Physics*, 120:278–304, 1995.

[33] W. J. Coirier and K. G. Powell. An accuracy assessment of Cartesian-mesh approaches for the Euler equations. *Journal of Computational Physics*, 117:121–131, 1995.

[34] W. J. Coirier and K. G. Powell. Solution-adaptive Cartesian cell approach for viscous and inviscid flows. *AIAA Journal*, 34(5):938–945, May 1996.

[35] M. J. Aftosmis, M. J. Berger, and J. E. Melton. Robust and efficient Cartesian mesh generation for component-base geometry. *AIAA Journal*, 36(6):952–960, 1998.

[36] A. S. Almgren, J. B. Bell, P. Colella, L. H. Howell, and M. J. Welcome. A conservative adaptive projection method for the variable density incompressible Navier-Stokes equations. *Journal of Computational Physics*, 142:1–46, 1998.

[37] R. B. Pember, L.H. Howell, J.B. Bell, P. Colella, W. Y. Crutchfield, M. A. Fiveland, and J.P. Jessee. An adaptive projection method for unsteady, low Mach number combustion. *Combustion Science and Technology*, 140:123–168, 1998.

[38] J.P. Jessee, W.A. Fiveland, L.H. Howell, P. Colella, and R.B. Pember. An adaptive mesh refinement algorithm for the radiative transport equation. *Journal of Computational Physics*, 139(2):380–398, 1998.

[39] P. Colella, M. Dorr, and D. Wake. Numerical solution of plasma-fluid equations using locally refined grids. *Journal of Computational Physics*, 152:550–583, 1999.

[40] L.H. Howell, R.B. Pember, P. Colella, J.P. Jessee, and W. A. Fiveland. A conservative adaptive-mesh algorithm for unsteady, combined-mode heat transfer using the discrete ordinates method. *Numerical Heat Transfer*, 35:407–430, 1999.

[41] C. P. T. Groth, D. L. De Zeeuw, K. G. Powell, T. I. Gombosi, and Q. F. Stout. A parallel solution-adaptive scheme for ideal magnetohydrodynamics. Paper 99-3273, AIAA, June 1999.

[42] C. P. T. Groth, D. L. De Zeeuw, T. I. Gombosi, and K. G. Powell. Global three-dimensional MHD simulation of a space weather event: CME formation, interplanetary propagation, and and interaction with the magnetosphere. *Journal of Geophysical Research*, 105(A11):25,053–25,078, 2000.

[43] M. J. Aftosmis, M. J. Berger, and G. Adomavicius. A parallel multilevel method for adaptively refined Cartesian grids with embedded boundaries. Paper 2000-0808, AIAA, January 2000.

[44] M. J. Aftosmis, M. J. Berger, and S. M. Murman. Applications of space-filling curves to Cartesian methods for CFD. Paper 2004-1232, AIAA, January 2004.

[45] J. S. Sachdev, C. P. T. Groth, and J. J. Gottlieb. A parallel solution-adaptive scheme for predicting multi-phase core flows in solid propellant rocket motors. *International Journal of Computational Fluid Dynamics*, 19(2):157–175, 2005.

[46] X. Gao and C. P. T. Groth. A parallel adaptive mesh refinement algorithm for predicting turbulent non-premixed combusting flows. *International Journal of Computational Fluid Dynamics*, 20(5):349–357, 2006.

[47] X. Gao and C. P. T. Groth. A parallel solution-adaptive method for three-dimensional turbulent non-premixed combusting flows. *Journal of Computational Physics*, 229(5):3250–3275, 2010.

[48] C. P. T. Groth, D. L. De Zeeuw, T. I. Gombosi, and K. G. Powell. Three-dimensional MHD simulation of coronal mass ejections. *Advances in Space Research*, 26(5):793–800, 2000.

[49] C. P. T. Groth and S. A. Northrup. Parallel implicit adaptive mesh refinement scheme for body-fitted multi-block mesh. Paper 2005-5333, AIAA, June 2005.

[50] J. S. Sachdev and C. P. T. Groth. A mesh adjustment scheme for embedded boundaries. *Communications in Computational Physics*, 2(6):1095–1124, 2007.

[51] Z. J. Zhang and C. P. T. Groth. Parallel high-order anisotropic block-based adaptive mesh refinement finite-volume scheme. Paper 2011-3695, AIAA, June 2011.

[52] B. van der Holst and R. Keppens. Hybrid block-AMR in Cartesian and curvilinear coordinates: MHD applications. *Journal of Computational Physics*, 226:925–946, 2007.

[53] M. S. Day and J. B. Bell. Numerical simulation of laminar reacting flows with complex chemistry. *Combustion Theory and Modelling*, 4(4):535–556, 2000.

[54] J.B. Bell, M.S. Day, A.S. Almgren, M. J. Lijewski, and C. A. Rendleman. A parallel adaptive projection method for low Mach number flows. *International Journal for Numerical Methods in Fluids*, 40:209–216, 2002.

[55] J. B. Bell. AMR for low Mach number reacting flow. http://repositories.cdlib.org/lbnl/ LBNL-54351, 2004.

[56] J. B. Bell, M. S. Day, J. F. Grcar, M. J. Lijewski, J. F. Driscoll, and S. A. Filatyev. Numerical simulation of a laboratory-scale turbulent slot flame. In *Proceedings of the Combustion Institute*, volume 31, pages 1299–1307, 2007.

[57] J. B. Bell, R. K. Cheng, M. S. Day, and M. Leijewski. Interaction of turbulence and chemistry in a low-swirl burner. *Journal of Physics: Conference Series*, 125:012027, 2008.

[58] A. Aspden, J.B. Bell, and M.S. Day. Characterization of low Lewis number flames. In *Proceedings of Combustion Institute*, volume 33(1), pages 1463–1471, 2011.

[59] A. Aspden, J.B. Bell, and M.S. Day. Lewis number effects in distributed flames. In *Proceedings of Combustion Institute*, volume 33(1), pages 1473–1480, 2011.

[60] B. A. V. Bennett, J. Fielding, R. J. Mauro, M. B. Long, and M. D. Smooke. 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*, 3:657–687, 1998.

[61] B.A.V. Bennett and M.D. Smooke. Local rectangular refinement with application to nonreacting and reacting fluid flow problems. *Journal of Computational Physics*, 151:684–727, 1999.

[62] S. A. Northrup. A Parallel Adaptive-Mesh Refinement Scheme for Predicting Laminar Diffusion Flames. Master's thesis, University of Toronto, 2004.

[63] S. A. Northrup and C. P. T. Groth. Solution of laminar diffusion flames using a parallel adaptive mesh refinement algorithm. Paper 2005–0547, AIAA, January 2005.

[64] X. Gao. *A Parallel Solution-Adaptive Method for Turbulent Non-Premixed Combusting Flows*. PhD thesis, University of Toronto, September 2008.

[65] H. Paillère, K. G. Powell, and D. L. De Zeeuw. A wave-model-based refinement criterion for adaptive-grid computation of compressible flows. Paper 92-0322, AIAA, January 1992.

[66] M. B. Giles and N. A. Pierce. Adjoint equations in CFD: duality, boundary conditions and solution behavior. Paper 97-1850, AIAA, January 1997.

[67] M. B. Giles and N. A. Pierce. Improved lift and drag estimates using adjoint Euler equations. Paper 99-3293, AIAA, January 1999.

[68] M.B. Giles and N.A. Pierce. Adjoint error correction for integral outputs. In T. Barth and H. Deconinck, editors, *Error Estimation and Adaptive Discretization Methods in Computational Fluid Dynamics*, volume 25 of *Lecture Notes in Computer Science and Engineering*, pages 47–96. Springer-Verlag, Berlin, 2002.

[69] D. A. Venditti and D. L. Darmofal. Adjoint error estimation and grid adaptation for functional outputs: Application to quasi-one-dimensional flow. *Journal of Computational Physics*, 164:204– 227, 2000.

[70] D. A. Venditti and D. L. Darmofal. Grid adaptation for functional outputs: Application to two-dimensional inviscid flows. *Journal of Computational Physics*, 176:40– 69, 2002.

[71] D. A. Venditti and D. L. Darmofal. Anisotropic grid adaptation for functional outputs: Application to two-dimensional viscous flows. *Journal of Computational Physics*, 187:22– 46, 2003.

[72] V. Heuveline and R. Rannacher. Duality-based adaptivity in the *hp*-finite element method. *Journal of Numerical Math*, 11:1–18, 2003.

[73] R. Rannacher and B. Vexler. A priori error estimates for the finite element discretization of elliptic parameter identification problems with pointwise measurements. 44:1844–1863, 2005.

[74] M. Nemec and M. J. Aftosmis. Adjoint error estimation and adaptive refinement for embedded-boundary Cartesian meshes. Paper 2007-4187, AIAA, June 2007.

[75] M. Nemec and M. J. Aftosmis. Adjoint sensitivity computations for an embedded-boundary Cartesian mesh method. *Journal of Computational Physics*, 227(4):2724 – 2742, 2008.

[76] M. Nemec, M. J. Aftosmis, and M. Wintzer. Adjoint-based adaptive mesh refinement for complex geometries. Paper 2008-0725, AIAA, June 2008.

[77] H. Lomax, T. H. Pulliam, and D. W. Zingg. *Fundamentals of Computational Fluid Dynamics*. Scientific Computation. Springer, Berlin, 2001.

[78] M. Tabesh and D. W. Zingg. Efficient implict time marching methods using a Newton-Krylov algorithm. Paper 2009-164, AIAA, January 2009.

[79] V. Venkatakrishnan. Viscous computations using a direct solver. *Computers & Fluids*, 18:191–204, 1990.

[80] E. J. Nielsen, W. K. Anderson, R. W. Walters, and D. E. Keyes. Application of Newton-Krylov methodology to a three-dimensional unstructured Euler code. Paper 95-1733-CP, AIAA, June 1995.

[81] W. K. Anderson, R. D. Rausch, and D. L. Bonhaus. Implicit/multigrid algorithms for incompressible turbulent flows on unstructured grids. *Journal of Computational Physics*, 128:391–408, 1996.

[82] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, 2nd edition, 2003.

[83] Y. Saad and H. A. van der Vorst. Iterative solution of linear systems in the 20-th century. *CompAppMath*, 123:1–33, 2000.

[84] H. A. van der Vorst. Bi-CGSTAB: A fast and smoothly converging variant of bi-cg for the solution of nonsymmetric linear systems. *SIAM Journal for Scientific and Statistical Computing*, 13:631–644, 1992.

[85] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear equations. *SIAM Journal for Scientific and Statistical Computing*, 7(3):856–869, 1986.

[86] Y. Saad. Krylov subspace methods on supercomputers. *SIAM Journal for Scientific and Statistical Computing*, 10(6):1200–1232, 1989.

[87] P. N. Brown and Y. Saad. Hybrid Krylov methods for nonlinear systems of equations. *SIAM Journal for Scientific and Statistical Computing*, 11(3):450–481, 1990.

[88] Y. Saad. *Iterative Methods for Sparse Linear Systems*. PWS Publishing Company, Boston, 1996.

[89] J. E. Hicken and D. W. Zingg. A simplified and flexible variant of GCROT for solving nonsymmetric linear systems. *SIAM Journal on Scientific Computing*, 32(3):1672–1694, 2010.

[90] D. W. Zingg, M. Nemec, and T. H. Pulliam. A comparative evaluation of genetic and gradient-based algorithms applied to aerodynamic optimization. *European Journal of Computational Mechanics*, pages 103–126, 2008.

[91] L. B. Wigton, N. J. Yu, and D. P. Young. GMRES acceleration of computational fluid dynamics codes. Paper 85-1494, AIAA, July 1985.

[92] V. Venkatakrishnan and D. J. Mavriplis. Implicit solvers for unstructured meshes. *Journal of Computational Physics*, 105:83–91, 1993.

[93] T. J. Barth and S. W. Linton. An unstructured mesh Newton solver for compressible fluid flow and its parallel implementation. Paper 95-0221, AIAA, January 1995.

[94] A. Pueyo and D. W. Zingg. An efficient Newton-GMRES solver for aerodynamic computations. Paper 97-1955, AIAA, June 1997.

[95] H. Luo, J. D. Baum, and R. Löhner. A fast, matrix-free implicit method for compressible flows on unstructured grids. *Journal of Computational Physics*, 146:664–690, 1998.

[96] P. Wong and D.W. Zingg. A Newton-Krylov algorithm for turbulent aerodynamic flows on unstructured grids. Conference Paper 50th Annual Conference Aerodynamics Symposium, CASI, 2003.

[97] T. Chisholm and D.W. Zingg. Start-up issues in a Newton-Krylov algorithm for turbulent aerodynamic flows. Paper 2003-3708, AIAA, June 2003.

[98] D. A. Knoll and P. R. McHugh. Enhance nonlinear iterative techniques applied to nonequilibrium plasma flow. *SIAM Journal on Scientific Computing*, 19(1):291–301, 1998.

[99] J. M. C. Pereira, T. C. Hayashi, N. P. C. Marques, and J. C. F. Pereira. A coupled explicit Newton-Krylov matrix free numerical method for combustion in inert porous media. In *Proceedings of the 2nd International Conference on Computational Heat and Mass Transfer*, Federal University of Rio de Janeiro, Brazil, October 22-26, 2001.

[100] X. Yuan, S. C. Jardinb, and D. E. Keyes. Numerical simulation of four-field extended magnetohydrodynamics in dynamically adaptive curvilinear coordinates via Newton-Krylov-Schwarz. *Journal of Computational Physics*, 231(17):5822–5853, 2012.

[101] D. A. Knoll and D. E. Keyes. Jacobian-free Newton-Krylov methods: A survey of approaches and applications. *Journal of Computational Physics*, 193:357–397, 2004.

[102] A. Chapman, Y. Saad, and L. Wigton. High-order ILU preconditioners for CFD problems. *International Journal for Numerical Methods in Fluids*, 33:767–788, 2000.

[103] A. Pueyo and D. W. Zingg. An efficient Newton-GMRES solver for aerodynamic computations. *AIAA Journal*, 36:1991–1997, 1998.

[104] A. Jameson. Time dependent calculations using multigrid, with applications to unsteady flows past airfoils and wings. Paper 1991-1596, AIAA, June 1991.

[105] S. Isono and D. W. Zingg. A Runge-Kutta-Newton-Krylov algorithm for fourth-order implicit time marching applied to unsteady flows. Paper 2004–0433, AIAA, January 2004.

[106] Y. Saad and M. Sosonkina. Distributed Schur complement techniques for general sparse linear systems. *SIAM Journal on Scientific Computing*, 21(4):1337–1357, 1999.

[107] X.-C. Cai, W. D. Gropp, David E. Keyes, and M. D. Tidriri. Newton-Krylov-Schwarz methods in CFD. In *Proceedings of the International Workshop on Numerical Methods for the Navier-Stokes Equations*, pages 17–30, Braunschweig, 1995.

[108] David E. Keyes. Aerodynamic applications of Newton-Krylov-Schwarz solvers. In *14th International Conference on Numerical Methods in Fluid Dynamics*, volume 453, pages 1–20, Berlin, 1995. Springer.

[109] W. D. Gropp, D. E. Keyes, L. C. McInnes, and M. D. Tidriri. Globalized Newton-Krylov-Schwarz algorithms and software for parallel implicit CFD. Technical Report 98-24, ICASE, July 1998.

[110] X.-C. Cai, W. D. Gropp, D. E. Keyes, R. G. Melvin, and D. P. Young. Parallel Newton-Krylov-Schwarz algorithms for the transonic full potential equations. *SIAM Journal on Scientific Computing*, 19(1):246–265, 1998.

[111] W. D. Gropp, D. K. Kaushik, D. E. Keyes, and B. F. Smith. High-performance parallel implicit CFD. *Parallel Computing*, 27:337–362, 2001.

[112] J. E. Hicken and D. W. Zingg. A parallel Newton-Krylov solver for the Euler equations discretized using simultaneous approximation terms. *AIAA Journal*, 46:2773–2786, 2008.

[113] P. Beckman. Looking toward exascale computing. In *Parallel and Distirbuted Computing, Applications and Technologies*. IEEE, 2008.

[114] D. A. Knoll, P. R. McHugh, and D. E. Keyes. Newton-Krylov methods for low-Mach-number compressible combustion. *AIAA Journal*, 34(5):961–967, 1996.

[115] P. R. McHugh, D. A. Knoll, and D. E. Keyes. Application of Newton-Krylov-Schwarz algorithm to low-Mach-number compressible combustion. *AIAA Journal*, 36(2):290–292, 1998.

[116] R. R. Dobbins and M. D. Smooke. A fully implicit, compact finite difference method for the numerical solution of unsteady laminar flames. *Flow, Turbulence and Combustion*, 85:763–799, 2010.

[117] S. van Veldhuizen, C. Vuik, and C. R. Kleijn. On projected Newton-Krylov solvers for instationary laminar reacting gas flows. *Journal of Computational Physics*, 229:1724–1738, 2010.

[118] J.N. Shadid and R.S. Tuminaro. A comparison of preconditioned nonsymmetric krylov methods on a large-scale mimd machine. *SIAM Journal on Scientific Computing*, 15:440–459, 1994.

[119] J.N. Shadid, R.S. Tuminaro, and H.F. Walker. An inexact Newton method for fully coupled solution of the NavierStokes equations with heat and mass transport. *Journal of Computational Physics*, 137:155–185, 1997.

[120] M. R. J. Charest, C. P. T. Groth, and Ö. L. Gülder. A computational framework for predicting laminar reactive flows with soot formation. *Combustion Theory and Modelling*, 14(6):793–825, 2010.

[121] M. R. J. Charest, C. P. T. Groth, and Ö. L. Gülder. Solution of the equation of radiative transfer using a Newton-Krylov approach and adaptive mesh refinement. *Journal of Computational Physics*, 231:3023–3040, 2012.

[122] S. M. Candel and T. J. Poinsot. Flame stetch and the balance equation for the flame area. *Combustion Science and Technology*, 23:8009–815, 1990.

[123] O. Colin, F. Ducros, D. Veynante, and T. Poinsot. A thickened flame model for large eddy simulations of turbulent premixed combustion. *Physics of Fluids*, 12:1843–1863, 2000.

[124] S. Gordon and B. J. McBride. Computer program for calculation of complex chemical equilibrium compositions and applications I. analysis. Reference Publication 1311, NASA, 1994.

[125] B. J. McBride and S. Gordon. Computer program for calculation of complex chemical equilibrium compositions and applications II. users manual and program description. Reference Publication 1311, NASA, 1996.

[126] C. R. Wilke. A viscosity equation for gas mixtures. *Journal of Chemical Physics*, 18:517–519, 1950.

[127] G. Dixon-Lewis. Computer modeling of combustion reactions in flowing systems with transport. In W. C. Gardiner, editor, *Combustion Chemistry*, pages 21–126. Springer-Verlag, New York, 1984.

[128] I. Glassman. *Combustion*. Academic Press, 3rd edition, 1996.

[129] J. D. Anderson. *Hypersonic and High Temperature Gas Dynamics*. McGraw-Hill, New York, 1989.

[130] G. P. Smith, D. M. Golden, M. Frenklach, N. W. Moriarty, B. Eiteneer, M. Goldenberg, C. T. Bowman, R. K. Hanson, S. Song, W. C. Gardiner, V. V. Lissianski, and Z. Qin. GRI-Mech 3.0. http://www.me.berkeley.edu/gri_mech/.

[131] C. K. Westbrook and F. L. Dryer. Simplified reaction mechanisms for the oxidation of hydrocarbon fuels in flames. *Combustion Science and Technology*, 27:31–43, 1981.

[132] E. F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. Scientific Computation. Springer, Germany, 1999.

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

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

[135] J. J. Gottlieb and C. P. T. Groth. Assessment of Riemann solvers for unsteady one-dimensional inviscid flows of perfect gases. *Journal of Computational Physics*, 78:437–458, 1988.

[136] P. L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43:357–372, 1981.

[137] B. Einfeldt. On Godunov-type methods for gas dynamics. *SIAM Journal on Numerical Analysis*, 25:294–318, 1988.

[138] E. F. Toro, M. Spruce, and W. Speares. Restoration of the contact surface in the HLL-Riemann solver. *Shock Waves*, 4(1):25–34, 1994.

[139] T. J. Linde. *A Three-Dimensional Adaptive Multifluid MHD Model of the Heliosphere*. PhD thesis, University of Michigan, May 1998.

[140] M.-S. Liou. A sequel to AUSM, part II: AUSM$^+$-up for all apeeds. *Journal of Computational Physics*, 214:137–170, 2006.

[141] J.-S. Shuen, M.-S. Liou, and B. van Leer. Inviscid flux-splitting algorithms for real gases with non-equilibrium chemistry. *Journal of Computational Physics*, 90:371–395, 1990.

[142] R. W. Walters, P. Cinnella, D. C. Slack, and D. Halt. Characteristic-based algorithms for flows in thermochemical nonequilibrium. *AIAA Journal*, 30:1304–1313, 1992.

[143] B. van Leer. Upwind and high-resolution methods for compressible flow: From donor cell to residual distribution schemes. *Communications in Computational Physics*, 1(2):192–206, 2006.

[144] J. P. Boris and D. L. Book. Flux-corrected transport. i. SHASTA, a fluid transport algorithm that works. *Journal of Computational Physics*, 11:38–69, 1973.

[145] B. van Leer. Towards the ultimate conservative difference scheme. II. Monotonicity and conservation combined in a second order scheme. *Journal of Computational Physics*, 14:361–370, 1974.

[146] A. Harten. High resolution schemes for hyperbolic conservation laws. *Journal of Computational Physics*, 49:357–393, 1983.

[147] V. Venkatakrishnan. On the accuracy of limiters and convergence to steady state solutions. Paper 93-0880, AIAA, January 1993.

[148] T. J. Barth. Recent developments in high order k-exact reconstruction on unstructured meshes. Paper 93-0668, AIAA, January 1993.

[149] J. M. Weiss and W. A. Smith. Preconditioning applied to variable and constant density flows. *AIAA Journal*, 33(11):2050–2057, 1995.

[150] E. Turkel. Preconditioning techniques in computational fluid dynamics. *Annual Review of Fluid Mechanics*, 31:385–416, 1999.

[151] C. Depcik and B. van Leer. In search of an optimal local Navier-Stokes preconditioner. Paper 2003-3703, AIAA, June 2003.

[152] S. Venkateswaran and C. L. Merkle. Dual time stepping and preconditioning for unsteady computations. Paper 95–0078, AIAA, January 1995.

[153] S. Venkateswaran and C. L. Merkle. Analysis of preconditioning methods for the Euler and Navier–Stokes equations. Lecture Notes 1.1, Von Karman Institute, March 1999.

[154] P. Geuzaine. *An Implicit Upwind Finite-Volume Method for Compressible Turbulent Flows on Unstructured Meshes*. PhD thesis, Université de Liège, 1999.

[155] S. R. Mathur and J. Y. Murthy. A pressure-based method for unstructured meshes. *Numerical Heat Transfer*, 31:191–215, 1997.

[156] M.C. Thompson and J.H. Ferziger. An adaptive multigrid technique for the incompressible Navier-Stokes equations. *Journal of Computational Physics*, 82(1):94–121, 1989.

[157] J. S. Sachdev, C. P. T. Groth, and J. J. Gottlieb. Parallel solution-adaptive scheme for multi-phase core flows in rocket motors. Paper 2003–4106, AIAA, June 2003.

[158] X. Gao and C. P. T. Groth. Parallel adaptive mesh refinement scheme for three-dimensional turbulent non-premixed combustion. Paper 2008-1017, AIAA, January 2008.

[159] J. G. McDonald and C. P. T. Groth. Numerical modeling of micron-scale flows using the gaussian moment closure. Paper 2005-5035, AIAA, June 2005.

[160] Z. J. Zhang. Parallel anisotropic block-based adaptive mesh refinement finite-volume scheme. Master's thesis, University of Toronto, 2011.

[161] M. J. Williamschen and C. P. T. Groth. Parallel anisotropic block-based adaptive mesh refinement algorithm for three-dimensional flows. Paper, AIAA, June 2013.

[162] R. L. Davis and J. F. Dannenhoffer. Decomposition and parallelization strategies for adaptive grid-embedding techniques. *International Journal of Computational Fluid Dynamics*, 1:79–93, 1993.

[163] M. Sun and K. Takayama. Conservative smoothing on an adaptive quadrilateral grid. *Journal of Computational Physics*, 150:143–180, 1999.

[164] F. Ham, F.-S. Lien, and A. B. Strong. A Cartesian grid method with transient anisotropic adaption. *Journal of Computational Physics*, 179:469–494, 2002.

[165] W.A. Keats and F.-S. Lien. Two-dimensional anisotropic Cartesian mesh adaptation for the compressible Euler equations. *IJNMF*, 46:1099–1125, 2004.

[166] CGNS. CFD general notation system. http://www.grc.nasa.gov/www/cgns/.

[167] B. Smith, P. Bjorstad, and W. Gropp. *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press, 1996.

[168] C.-W. Ou and S. Ranka. Thoughts on the Chimera method of simulation of three-dimensional viscous flow. In *Proceedings of the 5th Symposium on the Frontiers of Massively Parallel Computation*, pages 367–374, McLean, VA, February 1995, 1995. IEEE Computer Society Press.

[169] J.R. Pilkington and S. B. Baden. Dynamic partitioning of non-uniform structured workloads with space-filling curves. Paper 3, IEEE Transaction on Parallel and Distributed Systems, 1996.

[170] B. Stroustrup. *The C++ Programming Language*. Addison-Wesley, New York, 1985.

[171] W. Gropp, E. Lusk, and A. Skjellum. *Using MPI*. MIT Press, Cambridge, Massachussets, 1999.

[172] C. Loken, D. Gruner, L. Groer, R. Peltier, N. Bunn, M. Craig, T. Henriques, C. Yu. J. Dempsey, Joseph C., J. Dursi J. Chong, S. Northrup, J. Pinto, N. Knecht, and R. Van Zon. SciNet: Lessons learned from building a power-efficient top-20 system and data centre. *Journal of Physics: Conference Series*, 256(1), 2010.

[173] R. Haring, M. Ohmacht, T. Fox, M. Gschwind, D. Satterfield, K. Sugavanam amnd P. Coteus, P. Heidelberger, M. Blumrich, R. Wisniewski, A. Gara, G. Chiu, P. Boyle, N. Christ, and K. Changhoan. The IBM Blue Gene/Q compute chip. *Micro, IEEE*, 32(2):48–60, 2012.

[174] Chen D., N. Eisley, P. Heidelberger, R. Senger, Y. Sugawara, S. Kumar, V. Salapura, D. Satterfield, B. Stienmacher-Burrow, and J. Parker. The IBM Blue Gene/Q interconnection fabric. *Micro, IEEE*, 32(1):32–43, 2012.

[175] A. Jameson. Solution of the Euler equations by a multigrid method. *Applied Mathematics and Computation*, 13:327–356, 1983.

[176] J. J. Alonso, L. Martinelli, and A. Jameson. Multigrid unsteady Navier-Stokes calculations with aeroelastic applications. Paper 95-0048, AIAA, January 1995.

[177] L. Dubuc, F. Cantariti, M. Woodgate, B. Gribben, K. J. Badcock, and B. E. Richards. Solution of the unsteady Euler equations using an implicit dual-time method. *AIAA Journal*, 36(8):1417–1424, August 1998.

[178] J. C. Chassaing, G. A. Gerolymos, and I. Vallet. Reynolds-stress model dual-time-stepping computation of unsteady three-dimensional flows. *AIAA Journal*, 41(10):1882–1895, October 2003.

[179] P. D. Boom and D. W. Zingg. Time-accurate flow simulations using an efficient Newton-Krylov-Schur approach with high-order temporal and spatial discretization. Paper 2013-0383, AIAA, January 2013.

[180] R. S. Dembo, S. C. Eisenstat, and T. Steihaug. Inexact Newton methods. *SIAM Journal on Numerical Analysis*, 19(2):400–408, 1982.

[181] T. T Chisholm and D. W. Zingg. A Jacobian-free Newton-Krylov algorithm for compressible turbulent fluid flows. *Journal of Computational Physics*, 228:3490–3507, 2009.

[182] H. A. Schwarz. Gesammelte mathematische abhandlungen. In *Vierteljahrsschrift der Naturforschenden Gesellschaft in Zürich*, volume 15, pages 272–286, 1870.

[183] J. A. Meijerink and H. A. van der Voorst. An iterative solution method for linear systems of which the coefficeint matrix is a symmetric M-matrix. *Mathematics of Computation*, 31:148–162, 1977.

[184] T. T. Chisholm. *A Fully Coupled Newton-Krylov solver with a one equation turbulence model*. PhD thesis, University of Toronto, 2006.

[185] E. Chow and M. A. Heroux. An object-oriented framework for block preconditioning. *Assosciation for Computatonal Machinery Transactions on Mathematical Software*, pages 159–183, 1998.

[186] Y. Saad. Sparskit 2. http://www-users.cs.umn.edu/ ∼ saad/software/SPARSKIT/, 2000.

[187] Y. Saad. A flexible inner-outer preconditioned gmres. *SIAM Journal on Scientific Computing*, 14(2):461–469, 1993.

[188] N. Nethercote and J. Seward. Valgrind: A framework for heavyweight dynamic binary instrumentation, 2007.

[189] W. A. Mulder and B. van Leer. Experiments with implicit upwind methods for the Euler equations. *Journal of Computational Physics*, 59:232–246, 1985.

[190] J.S. Sachdev. *Parallel Solution-Adaptive Method for Predicting Solid Propellant Rocket Motor Core Flows*. PhD thesis, University of Toronto, March 2007.

[191] J. F. Lynn. *Multigrid Solution of the Euler Equations with Local Preconditioning*. PhD thesis, University of Michigan, 1995.

[192] H. Schlichting. *Boundary-Layer Theory*. McGraw-Hill, Toronto, 7th edition, 1979.

[193] Reaction Design. CHEMKIN. http://www.reactiondesign.com/products/open/chemkin.html.

[194] R. K. Mohammed, M. A. Tanoff, M. D. Smooke, A. M. Schaffer, and M. B. Long. Computational and experimental study of a forced, time-varying, axisymmetric, laminar diffusion flame. In *Twenty-Seventh Symposium (International) on Combustion*, pages 693–702, Pittsburgh, 1998. The Combustion Institute.

[195] D. Durox, F. Baillot, P. Scoufflaire, and R. Prudhomme. Some effects of gravity on the behaviour of premixed flames. *Combustion and Flame*, 82:66–74, 1990.

[196] L. W. Kostiuk and R. K. Cheng. Imaging of premixed flames in microgravity. *Experiments in Fluids*, 18:59–68, 1994.

[197] L. W. Kostiuk and R. K. Cheng. The coupling of conical wrinkled laminar flames with gravity. *Combustion and Flame*, 103:27–40, 1995.

[198] R. K. Cheng, B. Bedat, and L. W. Kostiuk. Effects of buoyancy on lean premixed v-flames part i: laminar and turbulent flame structures. *Combustion and Flame*, 116:360–375, 1999.

[199] Y. T. Guahk, D. K. Lee, K. C. Oh, and H. D. Shin. Flame-intrinsic Kelvin–Helmholtz instability of flickering premixed flames. *Energy and Fuels*, 23:3875–3884, 2009.

[200] I. G. Shepherd, M. S. Day, and R. K. Cheng. The dynamics of flame flicker in conical premixed flames: An experimental and numerical study. LBNL Report LBNL-59249, 2005.

[201] P. Wong and D.W. Zingg. Three-dimensional aerodynamic computations on unstructured grids using a Newton-Krylov approach. *Computers & Fluids*, 37(2):107–120, 2008.

[202] V. Kindratenko and P. Trancoso. Trends in high performance computing. *Computing in Science & Engineering*, 13(3):92–95, 2011.

[203] J. E. Hicken and D. W. Zingg. Globalization strategies for inexact-Newton solvers. Paper 2009-4139, AIAA, June 2009.

[204] J. E. Hicken, J. E. Buckley, M. Osusky, and D. W. Zingg. Dissipation-based continuation: a globalization for inexact-Newton solvers. Paper 2011-2337, AIAA, June 2011.

[205] C. W. Oosterlee and T. Washio. Krylov subspace acceleration of nonlinear multigrid with applications to recirculating flows. *SIAM Journal on Scientific Computing*, 21(5):1670–1690, 2000.

[206] D. J. Mavriplis. An assessment of linear versus non-linear multigrid methods for unstructured mesh solvers. *Journal of Computational Physics*, 175:302–325, 2002.

[207] E. E. Prudencio and X.-C. Cai. Parallel multilevel restricted Schwarz preconditioners with pollution removing for PDE-constrained optimization. *SIAM Journal on Scientific Computing*, 29(1):964–985, 2007.

[208] F.-N. Hwang and X.-C. Cai. A new class of parallel two-level nonliner Schwarz preconditioned inexact Newton algorithms. *Computer Methods in Applied Mechanics and Engineering*, 196:1603–1611, 2007.

[209] L. Ivan. *Development of High-Order CENO Finite-Volume Schemes with Block-Based Adaptive Mesh Refinement*. PhD thesis, University of Toronto, 2010.

[210] L. Ivan and C. P. T. Groth. High-order solution-adaptive central essentially non-oscillatory CENO method for viscous flows. Paper 2011-0367, AIAA, January 2011.

[211] S. A. Northrup and C. P. T. Groth. Parallel implicit AMR scheme for unsteady reactive flows. 18th Annual Conference of the CFD Society of Canada, London, Ontario, Canada, May 17–19, 2010, May 2010.