DISSERTATION

# TIME INTEGRATION FOR COMPLEX FLUID DYNAMICS

Submitted by

Joshua C. Christopher

Department of Mechanical Engineering

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Fall 2021

## ABSTRACT

## TIME INTEGRATION FOR COMPLEX FLUID DYNAMICS

Efficient and accurate simulation of turbulent combusting flows in complex geometry remains a challenging and computationally expensive proposition. A significant source of computational expense is in the integration of the temporal domain, where small time steps are required for the accurate resolution of chemical reactions and long solution times are needed for many practical applications. To address the small step sizes, a fourth-order implicit-explicit additive Runge-Kutta (ARK4) method is developed to integrate the stiff chemical reactions implicitly while advancing the convective and diffusive physics explicitly in time. Applications involving complex geometry, stiff reaction mechanisms, and high-order spatial discretizations are challenged by stability issues in the numerical solution of the nonlinear problem that arises from the implicit treatment of the stiff term. Techniques for maintaining a physical thermodynamic state during the numerical solution of the nonlinear problem, such as placing constraints on the nonlinear solver and the use of a nonlinear optimizer to find valid thermodynamic states, are proposed and tested. Verification and validation are performed for the new adaptive ARK4 method using lean premixed flames burning hydrogen, showing preservation of 4th-order error convergence and recovery of literature results. ARK4 is then applied to solve lean, premixed $C_3H_8$-air combustion in a bluff-body combustor geometry. In the two-dimensional case, ARK4 provides a $70\times$ speedup over the standard explicit four-stage Runge-Kutta method and, for the three-dimensional case, three-orders-of-magnitude-larger time step sizes are achieved. To further increase the computational scaling of the algorithms, parallel-in-time (PinT) techniques are explored. PinT has the dual benefit of providing parallelization to long temporal domains as well as taking advantage of hardware trends towards more concurrency in modern high-performance computing platforms. Specifically, the multigrid reduction-in-time (MGRIT) method is adapted and enhanced by adding adaptive mesh refinement (AMR) in time.

This creates a space-time algorithm with efficient solution-adaptive grids. The new MGRIT+AMR algorithm is first verified and validated using problems dominated by diffusion or characterized by time periodicity, such as Couette flow and Stokes second problem. The adaptive space-time parallel algorithm demonstrates up to a 13.7$\times$ speedup over a time-sequential algorithm for the same solution accuracy. However, MGRIT has difficulties when applied to solve practical fluid flows, such as turbulence, governed by strong hyperbolic partial differential equations. To overcome this challenge, the multigrid operations are modified and applied in a novel way by exploiting the space-time localization of fine turbulence scales. With these new operators, the coarse-scale errors are advected out of the temporal domain while the fine-scale dynamics iterate to equilibrium. This leads to rapid convergence of the bulk flow, which is important for computing macroscopic properties useful for engineering purposes. The novel multigrid operations are applied to the compressible inviscid Taylor-Green vortex flow and the convergence of the low-frequency modes is achieved within a few iterations. Future work will be focused on a performance study for practical highly turbulent flows.

## ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Gao, for the guidance she has given me and all that I have learned from her. Through her feedback, discussions, and passion for research, I have grown as a student, researcher, and person. Likewise, I would like to thank Dr. Guzik for his excellent counsel during my time as a student.

I would like to thank my doctoral committee, Dr. Bangerth and Dr. Marchese, for their feedback and suggestions. For their help and guidance with XBraid and parallel-in-time, I would like to thank Dr. Falgout and Dr. Schroder.

I am forever indebted for the love and support from my parents and my brother. I appreciate the many times they came to visit and explore the Colorado mountains with me.

I would like to thank my colleagues Landon, Jordan, Nate S., Tyler, Justin, Noah, Scott, Nate O., Yijun, Sean, Chris, Erik, and Andy for the discussions and friendship.

I am grateful for the financial support from Dr. Gao and Colorado State University during the course of my studies and research.

# TABLE OF CONTENTS

# LIST OF TABLES

LIST OF FIGURES

# LIST OF SYMBOLS

## Alphanumeric

| | |
|---|---|
| $A_r$ | Pre-exponential factor in the rate constant for the $r^{th}$ reaction, depends on reaction |
| $c_n$ | Mass fraction of the $n^{th}$ species |
| $c_{p,n}$ | Specific heat capacity at constant pressure of the $n^{th}$ species, $\mathrm{J\,kg^{-1}\,K^{-1}}$ |
| $D$ | Number of dimensions |
| $D_n$ | Mass diffusion coefficient of the $n^{th}$ species, $\mathrm{m^2\,s^{-1}}$ |
| $E_{a,r}$ | Activation energy for the $r^{th}$ reaction, $\mathrm{cal\,mol^{-1}}$ |
| $\vec{\mathbf{F}}$ | Convective flux dyad, i.e., $[\mathbf{F}_x\,,\mathbf{F}_y\,,\mathbf{F}_z]$ |
| $\vec{\boldsymbol{\mathcal{G}}}$ | Mapped diffusion flux dyad, i.e., $[\boldsymbol{\mathcal{G}}_\xi\,,\boldsymbol{\mathcal{G}}_\eta\,,\boldsymbol{\mathcal{G}}_\zeta]$ |
| $G_n$ | Molar Gibbs free energy of the $n^{th}$ species, $\mathrm{J\,mol^{-1}}$ |
| $H_n$ | Molar specific enthalpy of the $n^{th}$ species, $\mathrm{J\,mol^{-1}}$ |
| $h_n$ | Total specific enthalpy of the $n^{th}$ species, $\mathrm{J\,kg^{-1}}$ |
| $k_{\mathrm{f},r}$ | Forward reaction rate for the $r^{th}$ reaction, depends on reaction |
| $k_{\mathrm{b},r}$ | Backward reaction rate for the $r^{th}$ reaction, depends on reaction |
| $K_{\mathrm{eq},r}$ | Equilibrium constant for the $r^{th}$ reaction, depends on reaction |
| $M_n$ | Molecular weight of the $n^{th}$ species, $\mathrm{kg\,mol^{-1}}$ |
| $N_r$ | Total number of reactions |
| $N_s$ | Total number of species |
| $p$ | Static pressure, $\mathrm{Pa}$ |
| $p_{\mathrm{atm}}$ | Atmospheric pressure, $\mathrm{Pa}$ |
| $R_n$ | Universal gas constant of the $n^{th}$ species, $\mathrm{J\,kg^{-1}\,K^{-1}}$ |
| $R_u$ | Molar universal gas constant, $8.314\,511\,\mathrm{J\,mol^{-1}\,K^{-1}}$ |
| $R_c$ | Molar universal gas constant (in $E_a$ units), $\mathrm{cal\,mol^{-1}\,K^{-1}}$ |
| $\mathbf{S}$ | Source term variable vector |

|  |  |
|---|---|
| $S_n$ | Molar specific entropy of the $n^{th}$ species, $\mathrm{J\,K^{-1}\,mol^{-1}}$ |
| $T$ | Temperature, $\mathrm{K}$ |
| $t$ | Time, $\mathrm{s}$ |
| $\mathbf{U}$ | Conservative variable vector |
| $u_d$ | Velocity component in the $d^{th}$ direction, $\mathrm{m\,s^{-1}}$ |
| $\mathbf{W}$ | Native primitive variable vector |
| $\tilde{\mathbf{W}}$ | Nonnative primitive variable vector |
| $x_d$ | Position in physical space in the $d^{th}$ direction |
| $[X_n]$ | Molar concentration of the $n^{th}$ species, $\mathrm{mol\,m^{-3}}$ |

**Dimensionless**

|  |  |
|---|---|
| Le | Lewis number |
| Ma | Mach number |
| Pr | Prandtl number |
| Re | Reynolds number |
| Sc | Schmidt number |

**Greek**

|  |  |
|---|---|
| $\alpha_{n,r}$ | Enhanced third-body efficiency of the $n^{th}$ species for the $r^{th}$ reaction |
| $\beta_r$ | Temperature exponent in the rate constant for the $r^{th}$ reaction |
| $\chi_n$ | Mole fraction of the $n^{th}$ species |
| $\dot{\omega}_n$ | Chemical production rate of the $n^{th}$ species, $\mathrm{s^{-1}}$ |
| $\kappa$ | Thermal conductivity, $\mathrm{W\,m^{-1}\,K^{-1}}$ |
| $\mu$ | Dynamic viscosity, $\mathrm{kg\,m^{-1}\,s^{-1}}$ |
| $\nu^{''}_{n,r}$ | Product stoichiometric coefficient of the $n^{th}$ species for the $r^{th}$ reaction |
| $\nu^{'}_{n,r}$ | Reactant stoichiometric coefficient of the $n^{th}$ species for the $r^{th}$ reaction |
| $\rho$ | Fluid density, $\mathrm{kg\,m^{-3}}$ |
| $\tau_n$ | Characteristic chemical destruction time of the $n^{th}$ species, $\mathrm{s}$ |
| $\xi_d$ | Position in computational space in the $d^{th}$ direction |

**Modifiers**

|  |  |
|---|---|
| $\lfloor \bullet \rceil$ | Round to the nearest integer |
| $\mathcal{O}\left(\Delta x^m\right)$ | Represents an error of the $m^{th}$ order of magnitude |
| $\vec{\bullet}$ | Denotes a vector |
| $\vec{\nabla}_\xi$ | Gradient or divergence operators in computational space |
| $\vec{\nabla}_x$ | Gradient or divergence operators in physical space |

# Chapter 1

# Introduction

## 1.1 Motivations and Objectives

While traditional combustion engines will continue to be used for power generation and vehicle propulsion for the next two or three decades, cleaner combustion engine technology must be developed to help slow down global warming and climate change. Accurately understanding the interactions among multiple physical phenomena, such as thermodynamics, turbulence, chemical reactions, and scalar transport, over a wide range of spatial and temporal scales, is difficult but necessary for engineering design. Computation has become the third pillar alongside experiment and theory. Moreover, computational techniques have advantages in investigating complex systems operating under extreme conditions which present challenges to physical experiments. Therefore, high-fidelity and high-performance simulations using advanced computational fluid dynamics (CFD) methods play an important role in modern engineering design cycles. Nevertheless, efficient and accurate modeling of practical combusting flows remains a difficult task for CFD.

One of the primary difficulties is the fast time scales of chemical reactions compared to the advective and diffusive ones presented by the multiple physical processes. The disparity in time scales leads to a stiff system of partial differential equations (PDEs) to solve numerically [1–3]. When using an explicit time marching method, such as the standard fourth-order explicit Runge-Kutta (ERK4) method, the solution can only be advanced with an exceptionally small time-step size due to the stability constraint, and the CFD simulation can easily take an intractable amount of CPU hours (e.g. months) to obtain accurate and meaningful results for statistical analysis that can be used to derive physical insights for new designs. Evidently, efficient and accurate time integration are desired for CFD modeling of stiff combustion problems for practical applications. Therefore, fourth-order additive Runge-Kutta (ARK4) methods are implemented and applied to increase the time step size by implicitly evolving the stiff chemical term while explicitly advancing the advection and diffusion terms of complex reacting fluid flows with practical combustor geometry.

Another difficulty for CFD modeling of complex reacting flows is the long numerical integration time required for the thermo-fluid dynamics to develop sufficiently. Reducing the run time for such simulations is of great importance for computational combustion engineering. Many strategies have been developed over the years, e.g., adaptive mesh refinement (AMR) [2, 4–14], high-order methods [7, 15–18], and spatially parallel CFD algorithms [4, 12, 13, 19–28]. As computer power increases, further speedup can be made possible through the parallelization of time stepping after spatial parallelization has saturated, provided there is sufficient space-time parallelism in the algorithms. This motivates the second part of the thesis research to be dedicated to the investigation of parallel-in-time (PinT) algorithms. In particular, multigrid reduction-in-time (MGRIT) appears well positioned to take advantage of this need by providing additional computational scaling and parallelization. Nevertheless, multigrid-based time-parallelization strategies have difficulties in convergence for hyperbolic PDEs, so new techniques must be developed to fully utilize time-parallel algorithms on future computing hardware.

To address these challenges, the following objectives are established in this thesis research:

1. Implement the ARK4 method with AMR in Chord to integrate the stiff chemical reactions implicitly in time while advancing the advection and diffusion flux explicitly. This allows for a significant increase in the step size taken by time integration without sacrificing the accuracy of the solution, leading to a reduction in time-to-solution.

2. Develop strategies for robustness and optimization to mitigate unphysical phenomena of the dynamical system. In particular, numerical errors such as those from the nonlinear solver can cause species mass fractions and temperature to be out of physical bounds, leading to unphysical thermodynamic states and divergence of the solution state.

3. Apply the ARK4+AMR algorithm to solve combusting flows with stiff reaction mechanisms occurring in complex geometry. The application evaluates the effectiveness of the integrated

numerical framework. Additionally, the effects of the geometry of the computational domain, the number of levels in AMR, and the types of fuels on the time scales are studied.

4. Add subcycling to MGRIT in XBraid. The addition of space-time adaptivity to MGRIT allows for additional parallel scaling in time. Furthermore, high performance computing (HPC) platforms are trending towards more concurrency, therefore enabling parallelization in the temporal domain will take advantage of this hardware trend.

5. Apply the adaptive, space-time parallel algorithm to turbulence for further computational efficiency. By exploiting the space-time localization of the fine scales of turbulent flows, multigrid is creatively used to cater to the multi-scale nature of turbulence.

These objectives are achieved by leveraging Chord and XBraid, and novel contributions are made to advance numerical algorithms. Chord [3,16,29–34] is the in-house solver developed by the CFD & Propulsion Laboratory at Colorado State University. Chord is a high-order finite-volume method (FVM) that solves the fully-coupled, compressible, reacting Navier-Stokes equations on structured AMR grids. Mapped multiblock (MMB) techniques enable Chord to solve flows with complex geometries such as the bluff-body combustor while effectively taking advantage of structured AMR grids. XBraid [35] is an open-source implementation of MGRIT. It has demonstrated speedups for a variety of problems, including a 2D unsteady flow vortex-shedding [36], a 3D Taylor-Green example with a moderate Reynolds number, eddy-current problems [37], optimization and machine learning [38, 39], 1D Burgers equation [40], linear hyperbolic problems [41], moving meshes [42], power systems [43, 44], linearized elasticity [45], elliptic problems [46, 47], time-fractional equations [48], and adjoint problems [38, 49]. Background information on the two primary methods upon which this study is built, namely ARK4 and PinT, is first reviewed. The HPC architectures used for this work is briefly described.

## 1.2 Additive Runge-Kutta Methods

Implicit-explicit (ImEx) time integration approaches have proven to be very promising for solution efficiency by partitioning the physics into stiff and non-stiff terms, then treating each with implicit and explicit methods, respectively. For example, Kennedy and Carpenter [50] introduced additive Runge-Kutta (ARK) methods for one-dimensional convection-diffusion-reaction (CDR) equations where convection and diffusion are solved explicitly and the reacting source term is solved implicitly. Zhang et al. [8] applied an ARK method to solve scalar convection-diffusion equations with AMR. Very recently, Chaplin [51] utilized the ARK method for simulating low Mach-number compressible flows ranging from inviscid gas dynamics to compressible Navier-Stokes with reactions in simple geometry. These studies demonstrate the computational efficiency of ARK over ERK4 for stiff reactions. There are many other ImEx methods, such as operator splitting [52–54], the SIMPLE algorithm and its successors [55, 56] and additive linear multi-step methods [57]; however, the focus of the present work is on the ARK family of ImEx methods. Readers are referred to those references for more information on general ImEx methods. The objective of the present study is to develop an efficient and accurate solution technique for solving stiff PDEs by integrating a fourth-order ARK, specifically, the 2-ARK$_4$(3)6L[2]SA method, with AMR on MMB grids in Chord. The 2-ARK$_4$(3)6L[2]SA method is introduced in Chapter 3. Then, the resulting algorithm is applied to solve complex, multidimensional combusting flows governed by the compressible Navier-Stokes equations coupled with chemical reactions in physical configurations with complex geometry.

## 1.3 Parallel-in-Time

Parallel-in-time methods have recently become an active research area and various approaches for parallel-in-time integration are available. To name a few, some methods are based on multiple shooting, waveform relaxation, domain decomposition, space-time multigrid methods, and direct time parallel methods [58]. In this study, the MGRIT algorithm [46] is employed, because of (i) demonstrated scalability [46, 47], (ii) non-intrusiveness and an ability to couple with many codes, and (iii) demonstrated success for some fluids problems [36]. Other space-time adaptive methods have been developed for parabolic equations, such as space-time multigrid with isogeometric analysis [59]. The space-time domain is decomposed into time-slices and connected using a discontinuous Galerkin technique. Steinbach et al. [60,61] construct a 4D finite element discretization in space and time and solve the system with GMRES and Kaczmarz relaxation. However, these methods require the entire formulation to be built around their respective finite-element discretizations. The non-intrusive nature of MGRIT allows for its coupling with existing discretizations such as those employed by structured AMR methods.

The present dissertation has two goals in this direction of research. The first goal is to add the space-time adaptivity to XBraid with the structured AMR Chombo library [62] for the CFD application code, Chord, and to demonstrate the validity, convergence, and performance of the resulting algorithm by solving the compressible Navier-Stokes equations. The second goal is to explore the application of this adaptive space-time parallel algorithm to multiscale turbulence.

This first goal is achieved by implementing the structured AMR in XBraid to create a unified methodology for constructing meshes and performing relaxation, prolongation, and restriction on space-time AMR grid hierarchies. This new feature combines the additional computational savings from subcycling with the base speedups seen from the existing temporal parallelization. The coupled algorithm is verified and validated by a transient Couette flow and a time-periodic Stokes second problem.

The second goal is achieved by applying the adaptive MGRIT algorithm to solve turbulent flows. To overcome the inherent mathematical issue on MGRIT for hyperbolic PDEs, a novel operator separating the high-frequency modes from low-frequency ones is developed. This makes use of the space-time localization of the the high-frequency modes in a turbulent flow and enables rapid convergence of the low-frequency large-scale dynamics of the flow. This new approach is tested on an infinite-Reynolds-number compressible Taylor-Green vortex problem.

## 1.4 High performance computing

The computing resources from three HPC centers were used in the course of this dissertation. The primary development and testing phases were carried out on the CFD & Propulsion Laboratory's Atlantis high performance compute server. This sever consists of nine compute nodes split into two partitions. The first partition consists of four nodes, each containing 20 Intel Sandy Bridge CPUs and 128 GB of memory, and the second partition consists of five nodes, each containing 24 Intel Haswell cores and 64 GB of memory. Cases run on Atlantis varied from single node tests to using the entire machine.

Large-scale testing and simulations of ARK4 cases were performed on Centennial and Onyx, managed by the Army Research Laboratory (ARL) Department of Defense (DoD) Supercomputing Resource Center (DSRC) and the U.S. Engineer Research & Development Center (ERDC) DSRC, respectively. Additional hardware information and development environment details can be found at https://centers.hpc.mil/systems/unclassified.html. Up to 4096 CPU cores were used for 3D cases with reacting turbulence.

Lastly, the PinT testing and production cases were run on Quartz and Ruby, managed by Lawrence Livermore National Laboratory (LLNL). The number of CPUs used varies based on the fine space-time resolution of the case, with the finest cases employing up to 4096 cores. Additional information on the technical details and capabilities Quartz and Ruby can be found at https://hpc.llnl.gov/hardware/platforms/Quartz and https://hpc.llnl.gov/hardware/platforms/ruby.

## 1.5 Dissertation Organization

This dissertation is structured as follows. For completeness and convenience, the mathematical modeling of the complex reacting flows and the underlying numerical framework where the fourth-order ARK (denoted by ARK4 throughout this dissertation) is implemented are described in Chapter 2. Chapter 3 reviews the relevant ARK4 method. Chapter 4 describes the solution algorithm for a single level grid. Chapter 5 presents the integration of ARK4 with AMR. Verification and validation of this algorithm is performed in Chapter 6 through a grid convergence study and comparisons of the solutions obtained by ARK4 and ERK4 for the reacting shock bubble problem. The validated algorithm is applied to solve combusting flows in a bluff-body combustor. Results on computational performance and speedup provided by ARK4 are reported and discussed in Chapter 7.

Chapter 8 introduces XBraid and MGRIT, highlighting the new adaptive space-time algorithm on the high-level algorithmic structure and challenges, and discusses the coupling of the AMR and MGRIT algorithms. Chapter 9 tests the resulting algorithm and presents its results on convergence and performance against the time-sequential algorithm. Chapter 10 proposes the modifications to the multigrid operators to support the application to infinite-Reynolds turbulent flows and demonstrates the results of the Taylor-Green case. Finally, Chapter 11 concludes the dissertation research, summarizes the original contributions, and points out the directions for future work.

# Chapter 2

# Mathematical and Numerical Modeling

## 2.1 Governing Equations

The system of PDEs governing compressible combustion consists of the Navier-Stokes and a set of species transport equations [3, 32] and specifically the mapped multiblock version [63–66] is used for this study to represent complex geometry on a Cartesian grid with AMR. To provide a brief background, the system of equations is given here. Grid mapping is used to transform from physical space, $\vec{x}$, to computational space, $\vec{\xi}$, where $\vec{\xi} = \vec{\xi}(\vec{x})$. The grid is assumed to not deform over time. The transformation grid metrics, $\mathrm{N}^{\mathrm{T}}$, and metric Jacobian, $J$, are defined as

$$\mathrm{N} = J\,(\vec{\nabla}_x \vec{\xi})^{\mathrm{T}}\,, \quad \mathrm{N}^{\mathrm{T}} = J\,\vec{\nabla}_x \vec{\xi}\,, \quad J \equiv \det(\vec{\nabla}_\xi \vec{x})\,,$$

where $\mathrm{T}$ denotes the transpose operation.

Applying the coordinate transformation to the continuity, momentum, energy, and species transport equations yields the governing equations for a compressible, thermally-perfect, reacting multispecies fluid

$$\frac{\partial}{\partial t}(J\rho) + \vec{\nabla}_\xi \cdot \left(\mathrm{N}^{\mathrm{T}} \rho \vec{u}\right) = 0\,, \tag{2.1}$$

$$\frac{\partial}{\partial t}(J\rho\vec{u}) + \vec{\nabla}_\xi \cdot \left(\mathrm{N}^{\mathrm{T}}(\rho\vec{u}\vec{u} + p\vec{\vec{I}})\right) = \vec{\nabla}_\xi \cdot (\mathrm{N}^{\mathrm{T}}\vec{\vec{\mathcal{T}}})\,, \tag{2.2}$$

$$\frac{\partial}{\partial t}(J\rho e) + \vec{\nabla}_\xi \cdot \left(\mathrm{N}^{\mathrm{T}} \rho\vec{u}(e + \frac{p}{\rho})\right) = \vec{\nabla}_\xi \cdot \left(\mathrm{N}^{\mathrm{T}}(\vec{\vec{\mathcal{T}}} \cdot \vec{u})\right) - \vec{\nabla}_\xi \cdot \left(\mathrm{N}^{\mathrm{T}} \vec{\mathcal{Q}}\right)\,, \tag{2.3}$$

$$\frac{\partial}{\partial t}(J\rho c_n) + \vec{\nabla}_\xi \cdot \left(\mathrm{N}^{\mathrm{T}} \rho c_n \vec{u}\right) = -\,\vec{\nabla}_\xi \cdot \left(\mathrm{N}^{\mathrm{T}}\, \vec{\mathcal{J}}_n\right) + J\rho\dot{\omega}_n\,, \quad n = 1\,\ldots\,N_s\,, \tag{2.4}$$

where $\rho$ is the density, $\vec{u}$ is the velocity vector, and $p$ is the pressure of the gaseous mixture. A total of $N_s$ species comprise the gaseous mixture, with $N_s$ transport equations. The ideal gas law provides the relation between density, pressure, and temperature for the mixture. $\vec{\vec{I}}$ is the identity tensor, and $e = |\vec{u}|^2/2 + \sum_{n=1}^{N_s} c_n h_n - p/\rho$ is the total specific energy, where $c_n$ and $h_n$ are the mass fraction and the specific enthalpy for species $n$. The calculation of the specific absolute enthalpy $h_n$ can be found in Gao et al. [32]. Essentially, the species enthalpy, thermal conductivity, and viscosity are calculated from a polynomial fit described by McBride et al. [67–69] and the Joint-Army-Navy-Air Force (JANAF) thermo-chemical tables [70].

To close the system, the molecular stress, heat flux, and species diffusion must be approximated [13]. The molecular stress, $\vec{\vec{\mathcal{T}}}$, is linearly proportional to the strain rate based on the Newtonian fluid assumption

$$\vec{\vec{\mathcal{T}}} = 2\mu\left(\vec{\vec{S}} - \frac{1}{3}J^{-1}\vec{\vec{I}}\,\vec{\nabla}_\xi \cdot \left(\mathrm{N}^{\mathrm{T}}\vec{u}\right)\right), \tag{2.5}$$

with the strain rate tensor, $\vec{\vec{S}}$, given by

$$\vec{\vec{S}} = \frac{1}{2}\left((\vec{\nabla}_\xi\vec{u})\,\left(\frac{\mathrm{N}^{\mathrm{T}}}{J}\right) + \left((\vec{\nabla}_\xi\vec{u})\,\left(\frac{\mathrm{N}^{\mathrm{T}}}{J}\right)\right)^{\mathrm{T}}\right). \tag{2.6}$$

Fourier's law is used to model the molecular heat flux, $\vec{\mathcal{Q}}$

$$\vec{\mathcal{Q}} = -\left(\kappa\frac{\mathrm{N}}{J}\vec{\nabla}_\xi T - \sum_{n=1}^{N_s}\left(h_n\,\vec{\mathcal{J}}_n\right)\right), \tag{2.7}$$

where $\kappa$ is the thermal conductivity coefficient and $\vec{\mathcal{J}}_n$ is the mass diffusion of species $n$

$$\vec{\mathcal{J}}_n = -\rho D_n\frac{\mathrm{N}}{J}\vec{\nabla}_\xi c_n\,. \tag{2.8}$$

The molecular diffusivity $D_n$ for species $n$ can be computed from the dynamic viscosity $\mu_n$ using the Schmidt number $\mathrm{Sc}$ as $D_n = \mu_n/(\rho\mathrm{Sc})$ or through the given Lewis number $\mathrm{Le}$ and the heat capacity at constant pressure $c_p$ by the relation of $D_n = \kappa/\rho c_p\mathrm{Le}$. Bulk viscosity is assumed to be negligible and there are no body forces present.

The reacting source term is based on the finite rate chemistry model described by Gao, Owen et al. [3, 11–13, 34]. For convenience, they are briefly described here since the chemical source Jacobian is required in the study and its derivation is dependent on the model of chemical source production rate. The mean reaction rate for species $n$ is calculated from the general form of the law of mass action [71] with

$$\dot{\omega}_n = \frac{M_n}{\rho} \sum_{r=1}^{N_r} \left( \nu^{''}_{n,r} - \nu^{'}_{n,r} \right) \left( \sum_{j=1}^{N_s} \alpha_{j,r} [X_j] \right) \left[ k_{f_r} \prod_{i=1}^{N_s} \left( [X_i] \right)^{\nu^{'}_{i,r}} - k_{b_r} \prod_{i=1}^{N_s} \left( [X_i] \right)^{\nu^{''}_{i,r}} \right] , \tag{2.9}$$

where $M_n$ is the molar mass of species $n$, $[X_n] = \rho c_n / M_n$ is the molar concentration of the $n$th species, $N_r$ is the number of chemical reaction steps, $\nu^{''}_{i,k}$ are the stoichiometric coefficients for the products, $\nu^{'}_{i,k}$ are the stoichiometric coefficients for the reactants, and $\alpha_{j,r}$ is the third-body coefficients specified in the reaction mechanism. The forward reaction rates are computed with the Arrhenius form $k_{f,r} = A T^{\beta_r} \exp\left( -E_{a,r} / R_u T \right)$ and, for reversible reactions, the backward reaction rate is $k_{\mathrm{b},r} = k_{\mathrm{f},r} / K_{\mathrm{eq},r}$ which is computed from the equilibrium rate

$$K_{\mathrm{eq},r} = \exp\left[ \sum_{n=1}^{N_s} \nu_{n,r} \left( \frac{-G_n}{R_u T} \right) \right] \left( \frac{p_{\mathrm{atm}}}{R_u T} \right)^{\sum_{n=1}^{N_s} \nu_{n,r}} , \tag{2.10}$$

with $\nu_{n,r}$ the change in products and reactants, $\nu_{n,r} = \nu^{''}_{n,r} - \nu'_{n,r}$, and Gibb's free energy [67]

$$\begin{aligned} \frac{G_n}{R_u T} = & - \frac{a_{1,n}}{2T} + a_{2,n} \left( 1 + \ln T \right) + a_{3,n} T \left( 1 - \ln T \right) \\ & - a_{4,n} \frac{T^2}{2} - a_{5,n} \frac{T^3}{6} - a_{6,n} \frac{T^4}{12} - a_{7,n} \frac{T^5}{20} + a_{8,n} - a_{9,n} T \, , \end{aligned} \tag{2.11}$$

where $R_u$ is the universal gas constant and the coefficients $a_{i,n}$ are the polynomial coefficients for species $n$ as provided by McBride, Gordon, and Reno [67]. Various reaction mechanisms are used in the present study and are introduced as they appear in Section 6.2 and Chapter 7.

## 2.2 Numerical Framework

The numerical framework supporting the current work is Chord [3, 16, 29–32, 34, 72]. Chord, built upon the Chombo parallel AMR library [62], features: a) fourth-order accuracy in space and time for smooth flows, b) adaptive mesh refinement in space and time, c) parallel scaling to over $1 \times 10^5$ CPU cores using a flat MPI-everywhere approach, and d) mapped grids for representing complex geometry. Generalized curvilinear transformation is used to map a structured grid in physical space to a Cartesian grid in computational space, and multiblock allows the construction of complex geometry. Solution adaptivity is achieved by applying a nested hierarchy of grids concentrated on physics of interest, such as large solution discontinuities, regions of high vorticity, or combustion flame fronts. Chord can model turbulence using both large eddy simulation (LES) and Reynolds-averaged Navier-Stokes (RANS) approaches. The LES model used in Chord is a state-of-the-art stretched-vortex model [73–75]. Chord is particularly powerful in solving high-speed flows with chemical reactions where detonation waves or shocks are present. A range of fuels and their chemical kinetics are available for combustion, including H2, CH4, C3H8, and NH3.

A system of semi-discrete ordinary differential equations (ODEs) is first obtained from finite-volume spatial discretization before applying a time marching method to advance the solution in time. The governing equations, Eqns. (2.1)–(2.4), are written in vector form and include components of the hyperbolic flux, elliptic flux, and reaction source

$$\frac{\partial(J\mathbf{U})}{\partial t} + \vec{\nabla}_\xi \cdot \left(\mathrm{N}^\mathrm{T}\big(\vec{\mathbf{F}} - \vec{\boldsymbol{\mathcal{G}}}\big)\right) = J\mathbf{S}\,. \tag{2.12}$$

with the hyperbolic flux dyad $\vec{\mathbf{F}}$, elliptic flux dyad $\vec{\boldsymbol{\mathcal{G}}}$, and reacting source vector $\mathbf{S}$ given by

$$\mathbf{U} = \begin{bmatrix} \rho \\ \rho\vec{u} \\ \rho e \\ \rho c_n \end{bmatrix}, \vec{\mathbf{F}} = \begin{bmatrix} \rho\vec{u} \\ \rho\vec{u}\vec{u} + p\vec{\vec{I}} \\ \rho\vec{u}\,(e + p/\rho) \\ \rho c_n\vec{u} \end{bmatrix}, \vec{\boldsymbol{\mathcal{G}}} = \begin{bmatrix} 0 \\ \vec{\vec{\mathcal{T}}} \\ \vec{\vec{\mathcal{T}}} \cdot \vec{u} - \vec{\mathcal{Q}} \\ \vec{\mathcal{J}}_n \end{bmatrix}, \vec{\mathbf{S}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \rho\dot{\omega}_n \end{bmatrix}. \tag{2.13}$$

For finite volume methods, the integral form of the governing equations is expressed first as

$$\frac{\partial}{\partial t}\int_{V_\xi} (J\mathbf{U}) + \int_{V_\xi} \vec{\nabla}_\xi \cdot \left(\mathrm{N}^{\mathrm{T}}(\vec{\mathbf{F}} - \vec{\boldsymbol{\mathcal{G}}})\right)\,\mathrm{d}V = \int_{V_\xi} J\mathbf{S}dV\,. \tag{2.14}$$

Then, the general divergence form is applied with the cell-averaged definition to arrive at the semi-discrete form of the Navier-Stokes equations

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\langle J\mathbf{U}\rangle_{\boldsymbol{i}} = -\frac{1}{h}\sum_{d=0}^{\mathrm{D}-1}\Bigg(&\left(\langle \mathrm{N}^{\mathrm{T}}{}_d\vec{\mathbf{F}}\rangle_{\boldsymbol{i}+\frac{1}{2}\boldsymbol{e}^d} - \langle \mathrm{N}^{\mathrm{T}}{}_d\vec{\mathbf{F}}\rangle_{\boldsymbol{i}-\frac{1}{2}\boldsymbol{e}^d}\right) \\ &- \left(\langle \mathrm{N}^{\mathrm{T}}{}_d\vec{\boldsymbol{\mathcal{G}}}\rangle_{\boldsymbol{i}+\frac{1}{2}\boldsymbol{e}^d} - \langle \mathrm{N}^{\mathrm{T}}{}_d\vec{\boldsymbol{\mathcal{G}}}\rangle_{\boldsymbol{i}-\frac{1}{2}\boldsymbol{e}^d}\right)\Bigg) + \langle J\mathbf{S}\rangle_{\boldsymbol{i}}\,,\end{aligned} \tag{2.15}$$

with $\langle\cdot\rangle$ indicating cell-averaged or face-averaged quantities. The computational space consists of a rectilinear-grid with cell centers marked by the points $(i_0, ..., i_{\mathrm{D}-1}) = \boldsymbol{i} \in \mathbb{Z}^{\mathrm{D}}$, and faces at the indices $\boldsymbol{i} \pm \frac{1}{2}\boldsymbol{e}^d$ where $\mathrm{e}^d$ is the unit vector in direction $d$, and $\mathrm{D}$ is the number of dimensions. A fourth-order center-differencing scheme is used to compute face-averaged quantities and gradients for flux evaluation. Face averaged quantities needed for the flux are interpolated from cell averaged quantities with

$$\begin{aligned}\langle\mathbf{W}\rangle_{\boldsymbol{j}+\frac{1}{2}\boldsymbol{e}^d} =& \frac{7}{12}\left(\langle\mathbf{W}\rangle_{\boldsymbol{j}} + \langle\mathbf{W}\rangle_{\boldsymbol{j}+\boldsymbol{e}^d}\right) \\ &- \frac{1}{12}\left(\langle\mathbf{W}\rangle_{\boldsymbol{j}-\boldsymbol{e}^d} + \langle\mathbf{W}\rangle_{\boldsymbol{j}+2\boldsymbol{e}^d}\right) + \mathcal{O}(\Delta x^4)\,,\end{aligned} \tag{2.16}$$

with $\langle \mathbf{W} \rangle_{\boldsymbol{j}+\frac{1}{2}\boldsymbol{e}^d}$ the average primitive value on face $\boldsymbol{j}+\frac{1}{2}\boldsymbol{e}^d$. The viscous flux additionally requires face averaged gradients normal and tangential to the face. Computing these gradients is a more extensive process than the face averaged values and the readers are referred to Gao et al. [16] for details on this, as well as other steps taken to preserve fourth-order accuracy on face value reconstruction and boundary conditions. When strong discontinuities or shock waves are present, the face-averaged quantities are limited via the Piece-wise Parabolic Method (PPM) limiter [15, 76, 77]. This creates a left and right state of the face-averaged quantities where an upwind scheme is applied by solving a Riemann problem at each face. A time integration method may then be used to evolve the semi-discrete ODEs in time (Eqn. (2.15)). While the standard fourth-order ERK4 method has been used in Chord for time integration, the present work is to enable ARK4 for efficient solution of stiff combustion simulations.

# Chapter 3

# Fourth-Order ImEx ARK Method

Kennedy and Carpenter [50, 78] provide a great deal of detail for the ARK4 family of time marching methods, and their work serves as an excellent reference. Herein, for completeness and convenience, the main solution procedure and features that are employed and adapted for the present work are described.

In the general ARK procedure, a semi-discrete form of the governing equations, such as Eqn. (2.15), has the right-hand side (RHS) split into $N$ terms

$$\frac{\mathrm{d}}{\mathrm{d}t}\langle J\mathbf{U}\rangle = \mathbf{L}(\langle J\mathbf{U}\rangle) = \sum_{\nu=1}^{N}\mathbf{L}(\langle J\mathbf{U}\rangle)^{[\nu]}\,, \tag{3.1}$$

where $\mathbf{L}(\langle J\mathbf{U}\rangle)^{[\nu]}$ represents one of the $N$ additive terms from which the RHS is constructed. The $N$ terms are integrated by an $m$-stage Runge-Kutta method where each stage $i$ is computed from

$$\langle J\mathbf{U}^{(i)}\rangle = \langle J\mathbf{U}^{(n)}\rangle + \Delta t^{(n)}\sum_{\nu=1}^{N}\sum_{j=1}^{m} a_{ij}^{[\nu]}\mathbf{L}^{[\nu],(j)}\,, \tag{3.2}$$

where $\langle J\mathbf{U}^{(n)}\rangle = \langle J\mathbf{U}(t^n)\rangle$ is the solution at time step $n$, $\langle J\mathbf{U}^{(i)}\rangle = \langle J\mathbf{U}(t^n + c_i\Delta t^{(n)})\rangle$ the solution at the $i$th stage, and $\mathbf{L}^{[\nu],(i)} = \mathbf{L}(\langle J\mathbf{U}^{(i)}\rangle)^{[\nu]}$ the additive term $\nu$. A nonlinear problem arises from Eqn. (3.2), for which the specifics of solving depend on the exact ARK method employed. At the end of the $m$th stage, the solution is updated with

$$\langle J\mathbf{U}^{(n+1)}\rangle = \langle J\mathbf{U}^{(n)}\rangle + \Delta t^{(n)}\sum_{\nu=1}^{N}\sum_{i=1}^{m} b_{i}^{[\nu]}\mathbf{L}^{[\nu],(i)}\,, \tag{3.3}$$

where $\langle J\mathbf{U}^{(n+1)}\rangle = \langle J\mathbf{U}(t^n + \Delta t^{(n)})\rangle$ is the solution at time step $n$+1. The present study uses step-size control for considerations of accuracy, iteration, and stability, and therefore the embedded scheme is included as

$$\langle \widehat{J\mathbf{U}}^{(n+1)} \rangle = \langle J\mathbf{U}^{(n)} \rangle + \Delta t^{(n)} \sum_{\nu=1}^{N} \sum_{i=1}^{m} \hat{b}_i^{[\nu]} \mathbf{L}^{[\nu],(i)} \,, \tag{3.4}$$

where $\hat{\cdot}$ indicates a quantity associated with the embedded scheme. The solution $\langle \widehat{J\mathbf{U}}^{(n+1)} \rangle$ is used in conjunction with the dense output for computing stage value predictors as initial guess for the nonlinear solver. The coefficients $a_{ij}^{[\nu]}$, $b_i^{[\nu]}$, $\hat{b}^{[\nu]}$, and $c_i^{[\nu]}$ are Butcher tableau coefficients and can be found in references [50, 78].

For the present study, the 2-ARK$_4$(3)6$L$[2]$SA$ scheme is used, whose format is uniquely identified as that there are 2 additive terms, the order of the main method is 4, the order of the embedded method is 3, there are 6 stages, it is L-stable, the 2nd-order accuracy of the stage-order of the implicit method, and the stiff term is integrated with the explicit singly diagonal implicit Runge-Kutta (ESDIRK) method. ESDIRK is a subclass of Runge-Kutta methods that, like ERK, utilize a lower-diagonal Butcher tableau, but are better suited for stiff problems. Explicit singly diagonal indicates that the first stage is computed explicitly and that the diagonal coefficients of the Butcher tableau are identical. Each stage after the first is solved implicitly, providing better performance compared to fully implicit methods [79].

Accordingly, Eqn. (2.15) is split into two additive terms, a non-stiff term solved explicitly and a stiff term solved implicitly. As the goal of this study is to increase the time-stepping size for advancing chemical reactions, the reacting source term is chosen to be solved implicitly while the inertial and viscous fluxes are chosen to be solved explicitly. This leads to the semi-discrete ARK form

$$\frac{\mathrm{d}}{\mathrm{d}t} \langle J\mathbf{U} \rangle = \mathbf{L}(\langle J\mathbf{U} \rangle) = \mathbf{L}(\langle J\mathbf{U} \rangle)^{[\mathrm{ns}]} + \mathbf{L}(\langle J\mathbf{U} \rangle)^{[\mathrm{s}]} \,, \tag{3.5}$$

with the two terms

$$\begin{aligned}\mathbf{L}(\langle J\mathbf{U}\rangle)^{[\text{ns}]} =& -\frac{1}{h}\sum_{d=0}^{D-1}\Bigg( \left(\langle \mathrm{N}^{\mathrm{T}}{}_{d}\vec{\mathbf{F}}\rangle_{\boldsymbol{i}+\frac{1}{2}\boldsymbol{e}^{d}} - \langle \mathrm{N}^{\mathrm{T}}{}_{d}\vec{\mathbf{F}}\rangle_{\boldsymbol{i}-\frac{1}{2}\boldsymbol{e}^{d}}\right) \\ & - \left(\langle \mathrm{N}^{\mathrm{T}}{}_{d}\vec{\boldsymbol{\mathcal{G}}}\rangle_{\boldsymbol{i}+\frac{1}{2}\boldsymbol{e}^{d}} - \langle \mathrm{N}^{\mathrm{T}}{}_{d}\vec{\boldsymbol{\mathcal{G}}}\rangle_{\boldsymbol{i}-\frac{1}{2}\boldsymbol{e}^{d}}\right)\Bigg)\,, \qquad (3.6)\\ \mathbf{L}(\langle J\mathbf{U}\rangle)^{[\text{s}]} =& \langle J\mathbf{S}\rangle_{\boldsymbol{i}}\,. \qquad (3.7)\end{aligned}$$

The superscript [ns] indicates the non-stiff term that is solved explicitly and the superscript [s] indicates the stiff term that is solved implicitly. Substitute these two terms into Eqn. (3.2) to get the specific stage values

$$\langle J\mathbf{U}^{(i)}\rangle = \langle J\mathbf{U}^{(n)}\rangle + \Delta t^{(n)}\sum_{j=1}^{m} a_{ij}^{[\text{ns}]}\mathbf{L}^{[\text{ns}],(j)} + a_{ij}^{[\text{s}]}\mathbf{L}^{[\text{s}],(j)}\,. \qquad (3.8)$$

The stage values must be found by solving the nonlinear problem

$$\langle J\mathbf{U}^{(i)}\rangle = \langle J\mathbf{U}^{(n)}\rangle + \mathbf{X}^{(i)} + \Delta t\gamma\mathbf{L}^{[\text{s}],(i)}\,, \quad i \geq 2\,, \qquad (3.9)$$

where previous stage values are used to compute $\mathbf{X}^{(i)}$ explicitly by

$$\mathbf{X}^{(i)} = \Delta t\sum_{j=1}^{i-1}\left(a_{ij}^{[\text{ns}]}\mathbf{L}^{[\text{ns}],(j)} + a_{ij}^{[\text{s}]}\mathbf{L}^{[\text{s}],(j)}\right)\,, \qquad (3.10)$$

with $\gamma = 1/4$ [50]. The coefficients $a_{ij}^{[\text{ns}]}$ correspond to matrix entries from the Butcher tableau used to integrate the non-stiff terms explicitly, while $a_{ij}^{[\text{s}]}$ corresponds to matrix entries from the Butcher tableau used to integrate the stiff terms implicitly. A modified Newton iteration method is employed to solve Eqn. (3.9) by linearizing the nonlinear term with respect to the reference time $t^n$ and the solution at $i$th stage and Newton iteration $k$: $\langle J\mathbf{U}^{(i)}\rangle_k$. Designate the Jacobian $\mathbf{J} = \partial\mathbf{L}^{[\text{s}],(j)}/\partial\mathbf{U}$, take the first two terms of the Taylor expansion, eliminate the explicit dependence on time, then expand about $\langle J\mathbf{U}^{(i)}\rangle_{k+1}$ to arrive at

$$\mathbf{L}_{k+1}^{[\mathrm{s}],(i)} = \mathbf{L}_{k}^{[\mathrm{s}],(i)} + \mathbf{J}_{k}^{(i)} \left( \langle J\mathbf{U}^{(i)}\rangle_{k+1} - \langle J\mathbf{U}^{(i)}\rangle_{k} \right) . \tag{3.11}$$

This expansion is substituted into Eqn. (3.9)

$$\langle J\mathbf{U}^{(i)}\rangle = \langle \mathbf{U}^{(n)}\rangle + \mathbf{X}^{(i)} + \Delta t\gamma \mathbf{L}_{k}^{[\mathrm{s}],(i)} + \Delta t\gamma \mathbf{J}_{k}^{(i)} \left( \langle J\mathbf{U}^{(i)}\rangle_{k+1} - \langle J\mathbf{U}^{(i)}\rangle_{k} \right) \ , \ i \geq 2 \, , \tag{3.12}$$

and rearranging it leads to the form of

$$(I - \Delta t\gamma \mathbf{J}_{k}^{(i)})\Delta\mathbf{U} = -(\langle J\mathbf{U}^{(i)}\rangle_{k} - \langle J\mathbf{U}^{(n)}\rangle) + \mathbf{X}^{(i)} + \Delta t\gamma \mathbf{L}_{k}^{[\mathrm{s}],(i)} \, , \tag{3.13}$$

where $k$ is the Newton iteration and $\Delta\mathbf{U} \equiv \left( \langle J\mathbf{U}^{(i)}\rangle_{k+1} - \langle J\mathbf{U}^{(i)}\rangle_{k} \right)$. A converged solution from Eqn. (3.13) provides the value $\langle \mathbf{U}_{k+1}^{(i)}\rangle$ which is the solution of stage $i$, that is $\langle \mathbf{U}^{(i)}\rangle$.

The step update to time $t^{n+1}$ is

$$\langle J\mathbf{U}^{(n+1)}\rangle = \langle J\mathbf{U}^{(n)}\rangle + \Delta t^{(n)} \left( \sum_{i=1}^{6} b_i \mathbf{L}^{[\mathrm{ns}],(i)} + \sum_{i=1}^{6} b_i \mathbf{L}^{[\mathrm{s}],(i)} \right) , \tag{3.14}$$

for the fourth-order method, and

$$\langle \widehat{J\mathbf{U}}^{(n+1)}\rangle = \langle J\mathbf{U}^{(n)}\rangle + \Delta t^{(n)} \left( \sum_{i=1}^{6} \hat{b}_i \mathbf{L}^{[\mathrm{ns}],(i)} + \sum_{i=1}^{6} \hat{b}_i \mathbf{L}^{[\mathrm{s}],(i)} \right) , \tag{3.15}$$

for the third-order embedded method.

## 3.1 Time Step Size Control

For better control of accuracy, iteration, and stability, the PID-controller as described in Kennedy and Carpenter [50] is considered, using

$$\Delta t_{\text{PID}} = \kappa \Delta t^{(n)} \left[ \frac{\epsilon_{\text{PID}}}{||\delta^{(n+1)}||_\infty} \right]^\alpha \left[ \frac{||\delta^{(n)}||_\infty}{\epsilon_{\text{PID}}} \right]^\beta \left[ \frac{\epsilon_{\text{PID}}}{||\delta^{(n-1)}||_\infty} \right]^\gamma . \tag{3.16}$$

In this formula, $\delta$ is the difference between the solution state associated with the fourth-order method (Eqn. (3.14)) and the solution state associated with the third-order embedded method (Eqn. (3.15)): $\delta^{(n+1)} = \langle J\mathbf{U}^{(n+1)} \rangle - \langle \widehat{J\mathbf{U}}^{(n+1)} \rangle$. The max norm is evaluated over all solution components. In the present study, it was found through numerical experimentation that $\epsilon_{\text{PID}} = 5.0 \times 10^{-10}$ provides a stable time integration method without limiting the step size significantly. Other parameters are specified as $\kappa = 0.9$ and the exponents by

$$\alpha = \left[ k_I + k_P + \left( \frac{2\omega_n}{1+\omega_n} \right) k_D \right] /p \,, \quad \beta = \left[ k_P + 2\omega k_D \right] /p \,, \quad \gamma = \left( \frac{2\omega_n^2}{1+\omega_n} \right) /p \,, \tag{3.17}$$

$$\tag{3.18}$$

with $p = 3$ the order of the embedded method, $k_I = 0.25$, $k_P = 0.14$, $k_D = 0.1$, and $\omega_n = \Delta t^{(n)} / \Delta t^{(n-1)}$.

## 3.2 Stage-Value Predictors

To potentially provide a better initial guess for the nonlinear solve of Eqn. (3.9), the dense output format is used to extrapolate stage-value guesses for stage $i$. Concerning the stability, a second order dense output method [80] is adopted and the specific form is given by

$$\langle J\mathbf{U}^{(i)} \rangle \left( t^{(n)} + \theta_i \Delta t^n \right) = \langle J\mathbf{U}^{(n)} \rangle + \Delta t^n \sum_{i=1}^{m} b_i^* \left( \theta_i \right) \left( \mathbf{L}^{[\text{ns}],(i)} + \mathbf{L}^{[\text{s}],(i)} \right) , \tag{3.19}$$

with the extrapolation coefficient $\theta_i = 1 + rc_i$, where $r = \Delta t^{(n)}/\Delta t^{(n-1)}$. The coefficients $b_i^*$ and $c_i$ are in the Butcher tableau [50, 78].

## 3.3 Stability of 2-ARK$_4$(3)6L[2]SA

A stability analysis is performed using a scalar ODE with both stiff and non-stiff terms,

$$\frac{d\phi}{dt} = \lambda^{[\text{ns}]}\phi + \lambda^{[\text{s}]}\phi\,, \tag{3.20}$$

with non-stiff eigenvalues $\lambda^{[\text{ns}]}$ from the $\mathbf{L}^{[\text{ns}]}$ term that is solved explicitly and stiff eigenvalues $\lambda^{[\text{ns}]}$ from the $\mathbf{L}^{[\text{s}]}$ term that is solved implicitly.

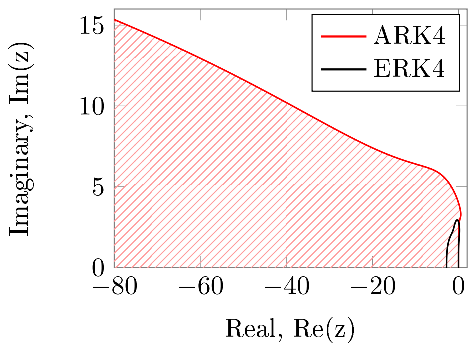
The stability function is [50, 81]

$$\mathcal{R}(\lambda^{[\text{ns}]}\Delta t, \lambda^{[\text{s}]}\Delta t) = 1 + \left(\lambda^{[\text{ns}]}\Delta t + \lambda^{[\text{s}]}\Delta t\right) \boldsymbol{b} \left(\mathrm{I} - \lambda^{[\text{ns}]}\Delta t \mathrm{A}^{[\text{ns}]} - \lambda^{[\text{s}]}\Delta t \mathrm{A}^{[\text{s}]}\right)^{-1} \boldsymbol{e}\,, \tag{3.21}$$

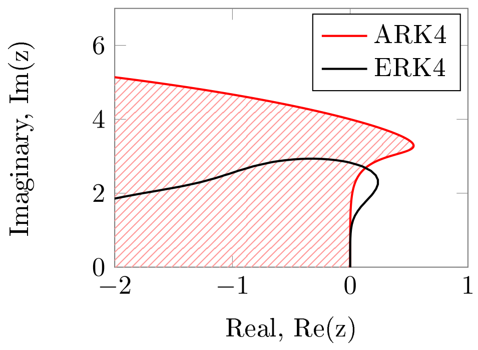
where $\boldsymbol{e} = \{1, 1, \ldots, 1\}$ and the equation satisfies [8]

$$\mathcal{R}(\lambda^{[\text{ns}]}\Delta t, \lambda^{[\text{s}]}\Delta t) = \frac{\det\left(\mathbf{I} - \lambda^{[\text{ns}]}\Delta t \mathrm{A}^{[\text{ns}]} - \lambda^{[\text{s}]}\Delta t \mathrm{A}^{[\text{s}]} + \left(\lambda^{[\text{ns}]}\Delta t + \lambda^{[\text{s}]}\Delta t\right) \boldsymbol{e}\cdot\boldsymbol{b}\right)}{\det\left(\mathbf{I} - \lambda^{[\text{s}]}\Delta t \mathrm{A}^{[\text{s}]}\right)}\,. \tag{3.22}$$

Substituting in the Butcher Table coefficients $\boldsymbol{b}$, $\mathrm{A}^{[\text{ns}]}$, and $\mathrm{A}^{[\text{s}]}$ and plotting the stable region $|\mathcal{R}(\lambda^{[\text{ns}]}\Delta t, \lambda^{[\text{s}]}\Delta t)| < 1$ yields Fig. 3.1. Clearly, ARK4 indeed provides a much larger stability region than ERK4.



**(a)** Overview of the stability region.



**(b)** Stability region near the origin.

**Figure 3.1:** Stability region for ARK4 when treating the reaction term implicitly and the advection physics explicitly.

# Chapter 4

# Implementation of ARK4 in a Single-Level Algorithm

First, the ARK4 scheme is described for a single-level grid before introducing AMR. Alg. 1 presents the pseudo code for the single-level algorithm for ARK4 as a reference. In Alg. 1, the linear system, Eqn. (4.2), can be solved directly or iteratively. If an iterative approach is employed, then the linear solver has a separate convergence criterion from the nonlinear solver. The present study does a direct solve using LAPACK's LU decomposition with partial pivoting and row interchanges [82]. As the implicit method is of stage-order two, the chemical Jacobian, $\partial\mathbf{L}^{[\mathrm{s}]}/\partial\mathbf{U}$, is computed using cell-averaged quantities that are approximated by cell-centered values. This leads to a block-diagonal matrix for the Jacobian. In Eqn. (4.2), the matrix A is a block diagonal matrix

$$\mathrm{A} = \mathrm{diag}\left(\left[\mathbf{I} - \Delta t\gamma\frac{\partial\mathbf{L}^{[\mathrm{s}]}}{\partial\mathbf{U}}\bigg|_{k}\right]_{1,1}, \cdots, \left[\mathbf{I} - \Delta t\gamma\frac{\partial\mathbf{L}^{[\mathrm{s}]}}{\partial\mathbf{U}}\bigg|_{k}\right]_{N,N}\right). \tag{4.1}$$

Matrix A being block diagonal results in an efficient solution process for the data locality that allows each cell to be solved independently. Further, this allows each level of the AMR hierarchy to be advanced independently as in standard ERK4. For the outer nonlinear solver, the step length is computed as described in Section 4.1, though other line search methods such as the Goldstein-Armijo method described by Dennis and Schnabel [83] could be considered. After a number of numerical experiments, the tolerance value of $\epsilon_{\mathrm{NLS}} = 1.0 \times 10^{-8}$ is found to work well for the problems considered herein. Note that the convergence tests scale the tolerance value by the current mapped density value, $\langle J\rho^{(i)}\rangle_{k+1}$, which in practice leads to a convergence tolerance on the order of $1.0 \times 10^{-14}$.

**Algorithm 1** Algorithm to Solve the Nonlinear Problem

```
    **Objective:** Solve nonlinear problem $\langle J\mathbf{U}^{(i)}\rangle = \langle J\mathbf{U}^{(n)}\rangle + \mathbf{X}^{(i)} + \Delta t\gamma\mathbf{L}^{[s],(i)}$
 1: $\mathbf{X}^i \leftarrow \Delta t \sum_{j=1}^{i-1} a_{ij}^{[ns]}\mathbf{L}^{[ns],(j)} + a_{ij}^{[s]}\mathbf{L}^{[s],(j)}$
 2: $\langle J\mathbf{U}^{(i)}\rangle_{k=0} \leftarrow \langle J\mathbf{U}^{(i-1)}\rangle$        ▷ Initial guess of $\langle J\mathbf{U}^{(i)}\rangle$ at iteration $k = 0$
 3: **while** not converged **do**
 4:     Linearize the problem about $\langle J\mathbf{U}^{(i)}\rangle_{k+1}$:
```

$$\left(\mathbf{I} - \Delta t\gamma \left.\frac{\partial \mathbf{L}^{[s]}}{\partial \mathbf{U}}\right|_k\right)\left(\langle J\mathbf{U}^{(i)}\rangle_{k+1} - \langle J\mathbf{U}^{(i)}\rangle_k\right) = -\left(\langle J\mathbf{U}^{(i)}\rangle_k - \langle J\mathbf{U}^{(n)}\rangle\right) + \mathbf{X}^i + \Delta t\gamma\mathbf{L}_k^{[s],(i)}\,. \tag{4.2}$$

```
 5:     Solve this linear $\mathrm{A}_k^{(i)}\vec{x}^{(i)} = \vec{b}_k^{(i)}$ problem for $\langle J\mathbf{U}^{(i)}\rangle_{k+1}$ where:
```

$$\mathrm{A}_k^{(i)} = \left(\mathbf{I} - \Delta t\gamma \left.\frac{\partial \mathbf{L}^{[s]}}{\partial \mathbf{U}}\right|_k\right)\,, \tag{4.3}$$

$$\vec{x}_k^{(i)} = \langle J\mathbf{U}^{(i)}\rangle_{k+1} - \langle J\mathbf{U}^{(i)}\rangle_k\,, \tag{4.4}$$

$$\vec{b}_k^{(i)} = -\left(\langle J\mathbf{U}^{(i)}\rangle_k - \langle J\mathbf{U}^{(n)}\rangle\right) + \mathbf{X}^i + \Delta t\gamma\mathbf{L}_k^{[s],(i)}\,. \tag{4.5}$$

```
 6:     **if** $\|\vec{r}_k^{(i)}\| \le \langle J\rho^{(i)}\rangle_{k+1}\,\epsilon_{\mathrm{NLS}}$ or $\|\vec{x}_k^{(i)}\| \le \langle J\rho^{(i)}\rangle_{k+1}\,\epsilon_{\mathrm{NLS}}$ **then**
 7:         Converged
 8:     **else**
 9:         $\langle J\mathbf{U}^{(i)}\rangle_{k+1} \leftarrow \langle J\mathbf{U}^{(i)}\rangle_k + \eta\mathrm{A}^{-1}\vec{r}_k^i$        ▷ $\eta$ is step length
10:     **end if**
11:     With residual $\vec{r}_k^{(i)} = \mathrm{A}\vec{x}_k^{(i)}$.
12: **end while**
```

## 4.1 Nonlinear Solver's Step Length Calculation

Large step lengths in the nonlinear solver may cause negative species mass fractions. If species mass fractions in a cell are allowed to become significantly negative, then the thermodynamic state in that cell becomes inconsistent and a physically valid temperature and pressure cannot be determined. Typical methods of preventing negative species mass fractions, such as using an inert species to absorb error or re-normalizing the species to sum to unity [34, 84, 85], were found to lead the species into inconsistent thermodynamic states in this study.

Therefore, two different methods are implemented to limit the step length. First, the step length is reduced based on the change in species mass fractions at each iteration of the nonlinear solver.

Each nonlinear iteration starts with step length of unity, $\eta = 1$. For iteration $k$, species $n$ is evaluated for a change in sign. If the sign has changed in this iteration, that is if

$$\langle J\mathbf{U}^{(i)}\rangle_{k,n} \geq 0, \text{ and } \langle J\mathbf{U}^{(i)}\rangle_{k+1,n} < 0\,, \tag{4.6}$$

then the step length is reduced by

$$\eta = \min\left(\eta,\, \frac{-\langle J\mathbf{U}^{(i)}\rangle_{k,n}}{x_{k,n}^{(i)}}\right)\,. \tag{4.7}$$

If species $n$ at iteration $k$ is negative and becomes more negative, that is if $\langle J\mathbf{U}^{(i)}\rangle_{k,n} < 0$ and $x_{k,n}^{(i)} < 0$, then the species is prevented from becoming highly negative by directly setting the update value to

$$x_{k,n}^{(i)} = -\langle J\rho^{(i)}\rangle_{k=0}\, \epsilon_{NLS}\,, \tag{4.8}$$

where $\epsilon_{NLS}$ is the previously defined nonlinear solver convergence tolerance.

Even with the reduction in step length from the first method, the iteration counts remain high in some situations. In cases where the iteration count exceeds 10 iterations, the solution is typically oscillating between two states. Reducing the step length allows for converging on a single state. Accordingly, a reduction in step length is also implemented after a fixed number of nonlinear iterations, as shown in Eqn. (4.9). Both methods are necessary for robust convergence.

$$\eta = \begin{cases} \min(\eta,\, 0.5) & \text{if } k > 10\,, \\ \min(\eta,\, 0.25) & \text{if } k > 20\,, \\ \min(\eta,\, 0.1) & \text{if } k > 30\,. \end{cases} \tag{4.9}$$

## 4.2 Time Step Size Evaluation

The time step is calculated based on the maximum wave-speed for inviscid flux, the von Neumann number for diffusive flux, and a species destruction rate for the reacting terms. Since uniform grid spacing and refinement is used in computational space, $\Delta\xi = \Delta\eta = \Delta\zeta$, and only $\Delta\xi$ is used in the following notation. The maximum wave-speed calculation is similar to the CFL number for convection terms [16, 86]

$$\Delta t_{\text{inertial}} = \alpha \frac{\Delta\xi}{\left(|\vec{u}| + a\right)_{\max}} \,, \tag{4.10}$$

where $\Delta\xi$ is the grid spacing, the stability constraint $\alpha = 1.3925$ is derived by Colella et al. [7], $|\vec{u}|$ is the magnitude of velocity, and $a$ is the speed of sound (so that $\left(|\vec{u}| + a\right)_{\max}$ is the maximum wave speed in the domain). The von Neumann number is used to calculate the stable $\Delta t$ for the second-order diffusive terms [16, 72]

$$\Delta t_{\text{viscous}} = 2.5 \frac{\Delta\xi^2 \rho}{|\lambda_d|_{\max} \mu D} \,, \tag{4.11}$$

where $|\lambda_d|_{\max}$ is a stability constraint for the mapped grids [72], $\mu$ is the dynamic viscosity, and $D$ the number of spatial dimensions. The chemical time step is determined by [3]

$$\tau_n = \min\left(\frac{[X_n]}{\dot{\Phi}_n}\right) \,, \tag{4.12}$$

with $[X_n]$ the molar concentration and $\dot{\Phi}_n$ the destruction rate for the $n^{th}$ species defined by [87]

$$\dot{\Phi}_n = \sum_{r=1}^{N_r} \left( \nu^{'}_{n,r} k_{\text{f},r} \prod_{i=1}^{N_s} [X_i]^{\nu^{'}_{i,r}} + \nu^{''}_{n,r} k_{\text{b},r} \prod_{i=1}^{N_s} [X_i]^{\nu^{''}_{i,r}} \right) \,, \tag{4.13}$$

where all the terms and notation for the chemical time step are the same as for the source term (Eqn. (2.9)). The time step based on the destruction rate is given by $\Delta t_{\text{chemical}} = \min\left(\tau_1, \tau_2, \ldots, \tau_{N_s}\right)$

for $N_s$ number of species. The overall time step size is calculated using [3]

$$\Delta t = \left[ \left( \frac{1}{\text{CFL}\Delta t_{\text{inertial}}} + \frac{1}{\Delta t_{\text{viscous}}} + \frac{1}{\Delta t_{\text{chemical}}} \right)^{-1} \right] , \tag{4.14}$$

with typically CFL $\leq 1$. In this work, the chemical source term is treated implicitly and therefore the chemical step size is removed from the overall time-step constraint. Indeed, this is the purpose of employing ARK4, because the chemical time step is usually expected to be the smallest step size among them. Correspondingly, the overall time step size is determined by

$$\Delta t_{\text{physics}} = \left[ \left( \frac{1}{\text{CFL}\Delta t_{\text{inertial}}} + \frac{1}{\Delta t_{\text{viscous}}} \right)^{-1} \right] . \tag{4.15}$$

For pathologically stiff reaction mechanisms, instability still occurs. The last resort is to solve a nonlinear optimization problem as proposed and described in Section 4.4. Nevertheless, a solution of this optimization process may yield drastically different temperature or species mass fractions from the previous time step or from its neighboring cells. Although not ideal, the nonlinear optimization is used as a final attempt to find a consistent thermodynamic state which will help the dynamical system to recover gradually over the following time steps.

A more optimal solution is to avoid the instability issues, which can largely be accomplished by limiting the time step size with the PID-controller. However, in some cases the PID-controller predicts a step size smaller than the chemical time step size, particularly when a solution involves shock waves. It is unclear exactly why this happens, but it should be avoided for the sake of computational efficiency. Since the chemical time step size is the limiting factor for stability in this study, the following is proposed and works reasonably well as a criterion for determining the ARK4 time-step size

$$\Delta t = \min\{\max\{\Delta t_{\text{PID}},\, \Delta t_{\text{chemical}}\},\, \Delta t_{\text{physics}}\} . \tag{4.16}$$

Note that the PID-controller has a self-starting issue because it requires information from two previous time steps. Therefore, for the first two time steps, Eqn. (4.15) is used. For a case with extremely stiff chemical kinetics that begin at the initial conditions, the more conservative Eqn. (4.14) is used.

## 4.3 Chemical Source Jacobian

The chemical source term in the present ARK4 scheme is treated as a stiff term and thus is integrated implicitly in time. Therefore, the source Jacobian, or specifically, the chemical source Jacobian must be determined when solving Eqn. (3.13) for the next iteration. Since the reacting source term is local, the implicit solve, and therefore the Jacobian, is computed at every cell independently from the other cells. The chemical source Jacobian is determined analytically based on the finite rate Arrhenius formula [88]. A numerical Jacobian was also approximated by finite difference. However, the analytical Jacobian is preferred in the present study for two reasons. First, the formulation is more accurate while still applicable to arbitrary fuels and reaction mechanisms. Secondly, the analytical form takes into account third-body reactions precisely. This consideration of non-linearity helps the numerical stability. Through the study, the reaction Jacobian is found to have a major impact on the solution accuracy and stability. For convenience, the chemical source Jacobian is provided in Appendix A.

## 4.4 Optimization Method for Inconsistent Thermodynamic States

An important issue that deserves attention is how to handle a physical quantity that becomes unphysical during the numerical solution process. Often in numerical combustion, due to discretization error and round-off error, a species concentration may become negative, or the summation of all species mass fractions may be greater or less than unity, or both scenarios happen. Because of this, more than often, the temperature of the mixture could be out of the physical range. All is deemed unphysical. Previously, a simple renormalization correction method was used [34], and it worked well for the reactions used in that reference. For convenience and comparison, the simple renormalization method is duplicated here. This process is applied to both cell-averaged values and cell-point values, so the cell-averaged indicators, $\langle\cdot\rangle$, are omitted for the remainder of this section. For a species $n$, the mass fractions are truncated to the range of physically possible values if necessary $\tilde{\rho c_n} = \max(0, \min(\rho, \rho c_n))$. Then, the species are normalized by the sum of species mass fractions $\rho c_n = (\tilde{\rho c_n}\rho) \,/\, \left(\sum_{j=1}^{\mathrm{N_s}} \tilde{\rho c_j}\right)$.

Nevertheless, the renormalization method works less ideally for the hydrocarbon reactions, such as $CH_4$-air and $C_3H_8$-air considered in the present study. The reaction mechanisms for $CH_4$-air and $C_3H_8$-air are 13-species 38-reactions [89] and 25-species 66-reactions [90], respectively. Both reactions are stiff. Repeatedly, unphysical phenomena were observed during the solution process for the $CH_4$-air and $C_3H_8$-air combustion. In addition to the different fuel combustion kinetics, the present study introduces an additional nonlinear solver arising from the ARK4. Usually, the symptom would be demonstrated by the nonlinear solver which solves the temperature from the total energy by finding the root of the specific total energy of the mixture $f(T) = \sum_{n=1}^{N_s} c_n h_n(T) - RT - e + |\vec{u}^2|/2$, where the kinetic energy and total energy can be found directly from the conservative state. On top of this temperature nonlinear solve, ARK4 adds the nonlinear solve as Alg. 1. All may contribute to the unphysical issue.

With extensive and thorough numerical experiments by devising various correction methods, an optimal method which is based on BFGS [91] is achieved. The motivation behind the use of BFGS is built on the concept that the numerical correction of species in the solution process should be consistent in the mathematical and numerical sense. Due to discretization and round-off errors, the numerical system produces unphysical quantities and must then adjust itself through an optimization process to become physical again. The unphysical situation has been predominately associated with the species mass fractions. The species distribution should be adjusted by an optimization process while satisfying the total energy of the system, which is conserved. The optimization method is L-BFGS-B, a limited-memory quasi-Newton code for bound-constrained optimization. Its main concept is the use of limited-memory BFGS matrices to approximate the Hessian of the objective function. The method is especially useful when the Hessian matrix is not practical to compute. L-BFGS-B is widely used and well documented. Briefly, for this study, the objective functions and the constraints are

$$f(T, \tilde{c}_n) = \sum_{n=1}^{N_s} \tilde{c}_n h_n(T) - \sum_{n=1}^{N_s} \tilde{c}_n R_n T - e + \left|\vec{u}^2\right| / 2 \,, \tag{4.17}$$

$$\min_{T, c_n} \quad \mathcal{F}(T, c_n) = \left| f(T, \tilde{c}_n) \right| \,, \tag{4.18}$$

$$\text{s.t.} \quad 0 \leq c_n \leq 1,\ 290 \leq T \leq 5590, \text{ and } \tilde{c}_n = c_n / \sum_{j=1}^{N_s} c_j \,. \tag{4.19}$$

For the present combustion problems, the method has demonstrated a reasonable balance between the species correction and the overall numerical stability in comparison to the simple renormalization method. The computational cost is negligible.

# Chapter 5

# Integration of ARK4 with AMR

Now, the ARK4 scheme is implemented with AMR. Implementing the ARK4 method in the context of AMR presents numerical challenges. Furthermore, coupling high-order ImEx time integrators with high-order finite-volume methods introduces robustness issues. Operators to address the challenges are discussed. Subcycling and the temporal interpolation scheme required to enable ARK4 with AMR are briefly described.

## 5.1 Subcycling

Subcycling allows for adaptive refinement in time. The time step size of the fine grid, $\delta t$, is scaled from the time step size of the coarse mesh, $\Delta\tau$, by the refinement ratio $n = \Delta\tau/\delta t$. A fixed ratio $n$ for all levels maintains the same CFL condition for all spatial resolutions.

This process is shown in Fig. 5.1 for a two-level grid. The coarse level, $\Omega^0$, is first integrated from time point $\tau^0$ to $\tau^1$ using $\Delta\tau$. Next, the fine level, $\Omega^1$, is integrated in smaller steps from $t^0$ to $t^4$ using $\delta t$, which is one-fourth of $\Delta\tau$. Prior to each step on the fine grid, interpolations in space and time fill the invalid ghost cells surrounding the fine grid as indicated by the upwards pointing arrows. Also refer to Fig. 5.2 for the definition of ghost cells. After the fine grid has been integrated to the end of the subcycling interval ($t^3$), the fine solution is averaged down to the overlaying region on the coarse grid and flux corrections occur in adjacent coarse cells to preserve single-valued fluxes along the coarse-fine interface.

## 5.2 Interpolation in Time

In AMR with subcycling, invalid ghost cells need to be filled at interfaces via interpolation from the next coarser grid. Since subcycling leads to the finer grid being solved at some time intermediate time relative to the time points of the coarser grid, a time interpolation is required in addition to a spatial interpolation. Dense output as described by Kennedy-Carpenter [50] is

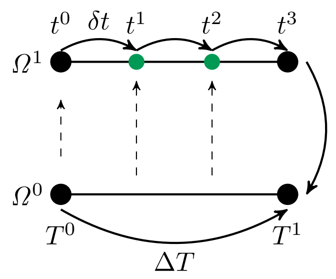


**Figure 5.1:** Subcycling allows coarse spatial meshes to take larger time steps than the nested fine spatial meshes. This allows each level to take time integrations with step sizes near the stability limit of the level.

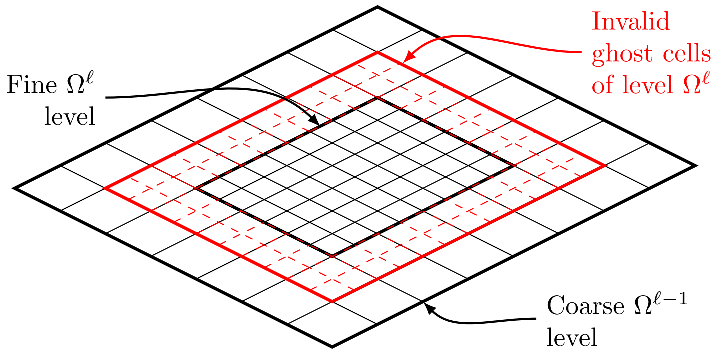

Invalid ghost cells of level $\Omega^\ell$
Fine $\Omega^\ell$ level
Coarse $\Omega^{\ell-1}$ level

**Figure 5.2:** An AMR hierarchy with a coarse level $\Omega^{\ell-1}$ and a fine level $\Omega^\ell$. Invalid ghost cells form a halo around $\Omega^\ell$ and allow centered stencil operations to be used on the fine level at the AMR interfaces.

used for the interpolation in time, and a fourth-order least squares approximation as described by McCorquodale-Colella [15] is used for the interpolation in space. As the spatial interpolator requires no changes to work with ARK4, only the time interpolation technique is described.

First, the invalid ghost cells that need to be filled by interpolation are described. Figure 5.2 shows two grids in the AMR hierarchy, a coarse level $\Omega^{\ell-1}$ and a fine level $\Omega^\ell$. The base level, denoted by $\Omega^0$, contains no invalid ghost cells because any ghost cells outside the domain on $\Omega^0$ are either physical boundary ghost cells that are filled by the physical boundary conditions or are periodic ghost cells. Therefore, only the AMR levels above the base grid are of concern for this interpolation process (that is, for levels $\Omega^\ell,\ \ell > 0$). Due to the proper nesting requirements [62], all invalid ghost cells on $\Omega^\ell$, $\ell > 0$ have a sufficient number of cells on grid $\Omega^{\ell-1}$ to perform the interpolation procedure using only the valid cells of $\Omega^{\ell-1}$.

During coarse grid integration from time $t^{(\ell-1)}$ to time $t^{(\ell-1)} + \Delta t^{(\ell-1)}$, the coarse stage values (Eqn. (3.8)) are stored for use in temporal interpolation. Then, the fine grid is integrated in time over that same interval with multiple smaller steps of size $\Delta t^{(\ell)}$. At each stage of each step on the fine level, the coarse solution is interpolated in time and space to fill the invalid ghost cells of level $\Omega^\ell$.

Dense output provides high-order interpolation of the coarse grid solution to any point between time $t^{(\ell-1)}$ and $t^{(\ell-1)} + \Delta t^{(\ell-1)}$. Given a fine level at time $t^{\ell} = t^{(\ell-1)} + \theta \Delta t^{(\ell-1)}$ such that $t^{(\ell-1)} \leq t^{(\ell)} \leq t^{(\ell-1)} + \Delta t^{(\ell-1)}$, where $\theta = \left(t^{\ell} - t^{(\ell-1)}\right) / \Delta t^{(\ell-1)}$, interpolation is performed with

$$\langle J\mathbf{U} \rangle^{(\ell-1)} \left(t^{(n)} + \theta \Delta t\right) = \langle J\mathbf{U}^{(n)} \rangle + (\Delta t)^{(n)} \sum_{i=1}^{s} b_i^* (\theta) \left(\mathbf{L}^{[\text{ns}],(i)} + \mathbf{L}^{[\text{s}],(i)}\right) , \tag{5.1}$$

where $b_i^* (\theta)$ is the dense output coefficient $b_i^* (\theta) = \sum\limits_{j=1}^{p^*} b_{ij}^* \theta^j$ and the values of $b^*$ are given in the Butcher tableau. An assumption is made that $b_{i,\text{ns}}^* = b_{i,\text{s}}^* = b_i^*$ which is valid for the ARK4 scheme used in the present study per Kennedy-Carpenter [50]. After the interpolated solution of the coarse grid is evaluated, the spatial interpolator is used to fill the invalid ghost cells of the fine spatial grid.

Stage-value prediction via extrapolation, as described in Section 3.2, can be performed with AMR with one caveat. Extrapolation requires the stage values from one previous time step, so stage-value prediction cannot be performed on the first time step. Likewise, when new finer spatial levels are created, previous step stage values are not available. Therefore, extrapolation is not done on the first step of a newly created fine level. Similarly, when an AMR level re-grids, the previous step stage values are not defined on newly refined regions. After re-gridding, extrapolation is not used for the first step.

PID step size control also requires information from two previous step solution values to be calculated. More precisely, it requires the max norm of the difference between the 4th-order solution and the solution associated with the 3rd-order embedded method for the previous two time steps. When a new level is created, Eqn. (4.15) is used for computing the step size for the first two time steps. When re-gridding happens, that is when the grids on an existing level adjust to updated solution conditions, the norms from the previous time steps may be used and the PID method can be used for step size control. Shown in Alg. 2 is the solution process for updating the solution on a level $\Omega^{\ell}$ with AMR.

**Algorithm 2** Recursive time advancement with AMR and ARK4

```
function ADVANCE($\ell$)
    Advance $\langle J\mathbf{U}\rangle^l$ from time $t^l$ to time $t^l + \Delta t^l$ per Section 4:
      1. Solve for stage $i$ values as described in Alg. 1
      2. Store stage values for interpolation on finer levels
      3. Accumulate flux values at faces on boundaries between $\Omega^\ell$ and $\Omega^{\ell+1}$
      4. Update solution per Eqn. (3.3)
    while $t^{\ell+1} < t^\ell$ do
        Call Advance($\ell + 1$)
    end while
    Synchronize the solution on level $\ell$ with the solution on level $\ell + 1$:
      1. Average solution down from overlying fine regions
      2. Perform flux corrections at boundaries between $\Omega^\ell$ and $\Omega^{\ell+1}$
      3. Update time $t^\ell \leftarrow t^\ell + \Delta t^\ell$
    Adjust the grid to the solution, if necessary
end function
```

# Chapter 6

# Verification and Validation of ARK4 with AMR

Prior to applying ARK4 to solve practical combusting flows, the algorithm is verified and validated. Verification is the process through which the programming implementation is confirmed to match the conceptual model. Through validation, the computational simulation is compared against literature data.

## 6.1 Verification

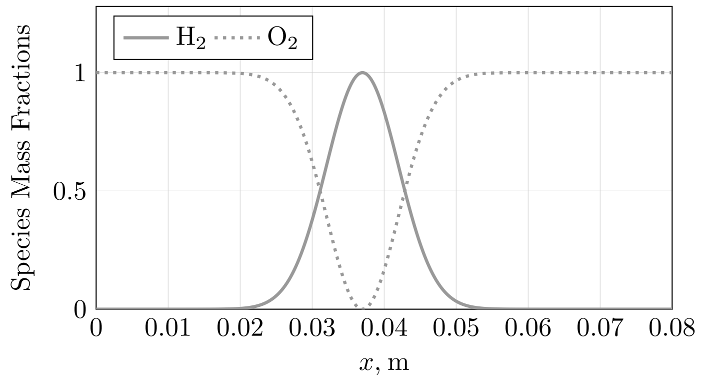


**Figure 6.1:** The initial hydrogen and oxygen mass fractions for the convection-diffusion-reaction test case.

To verify the ARK4+AMR algorithm, a grid convergence study is performed on a pseudo one-dimensional convection-diffusion-reaction test case with no gradients in the $y$-direction. A rectangular domain is used with periodic boundaries on all sides with a length of $8\,\mathrm{cm}$ and a height of $0.25\,\mathrm{cm}$. A set of four meshes were used from the coarse grid with $256 \times 8$ cells up to a fine grid of $4096 \times 128$ cells, with a refinement ratio of 2 between two consecutive grids. The $H_2$–$O_2$ combustion is modeled with a chemical mechanism of 8 species and 18 reactions without the inert $N_2$ [92]. A Gaussian distribution of the fuel, $H_2$, is defined by $c_{\mathrm{H}_2} = \exp\left(-(x - x_0)^2/(2\sigma^2)\right)$, where $\sigma = 0.005$ is the width of the distribution and $x_0 = 3.7\,\mathrm{cm}$ is the center of distribution. The oxidizer, $O_2$, is set to $c_{\mathrm{O}_2} = 1 - c_{\mathrm{H}_2}$ and all other species are initialized to zero. A profile

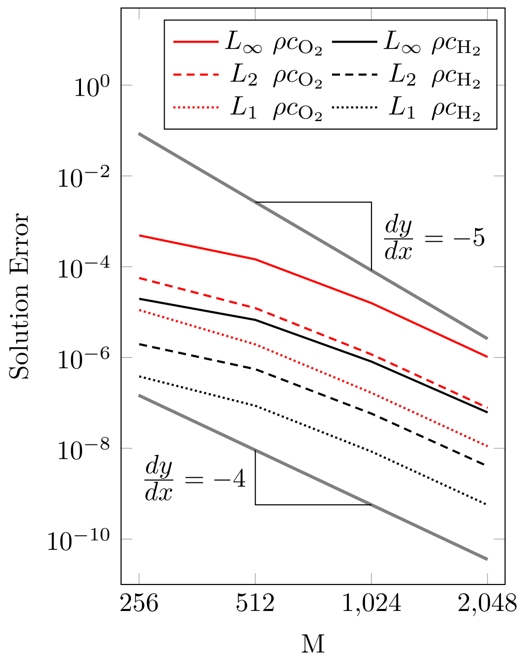


**Figure 6.2:** Error reduction rates of $\rho c_{O_2}$ and $\rho c_{H_2}$ converge to four as the mesh is refined.

of the initial conditions for the hydrogen and oxygen mass fractions is shown in Fig. 6.1. The temperature of the fuel is $T_{H_2} = 1000\,\mathrm{K}$ and the temperature of the oxidizer is $T_{O_2} = 2000\,\mathrm{K}$, the density of the mixture is initialized to $\rho_{\text{mix}} = p_{\text{atm}} \left(c_{H_2} R_{H_2} T_{H_2} + c_{O_2} R_{O_2} T_{O_2}\right)^{-1}$ where $R_{H_2}$ and $R_{O_2}$ are the gas constants of the fuel and oxidizer, respectively, and $p_{\text{atm}}$ is the standard atmospheric pressure. The entire domain is initialized with standard atmospheric pressure and a constant flow of $U_0 = 20\,\mathrm{m\,s^{-1}}$ in the positive $x$-direction.

Richardson extrapolation is used to verify the 4th-order error convergence rates using the $4096 \times 128$ case as the reference solution. Chord has previously been verified as achieving 4th-order error convergence for non-reacting multi-species flow [63], so this study focuses on verifying the convergence rate for reacting flows with ARK4 time integration. The reference case is run for 1600 time steps with a fixed $\Delta t = 2.5 \times 10^{-9}\,\mathrm{s}$. For the subsequent cases, the time step size and number of time steps is scaled in accordance with the CFL number. Shown in Fig. 6.2 are the errors of the conservative quantities $\rho c_{H_2}$ and $\rho c_{O_2}$ as the mesh is refined from 256 cells to 2048 cells in the $x$-direction, along with two guidelines showing a 4th order and 5th order slope. This shows that the solution errors for $\rho c_{O_2}$ and $\rho c_{H_2}$ are converging with 4th order accuracy as the grid is refined, confirming that the 4th order accuracy of the algorithm is maintained with ARK4 time integration. For completeness and reference, the solution error and convergence rates for all conserved solution quantities are tabulated in Table 6.1.

**Table 6.1:** Solution errors measured with the $L_\infty$-, $L_1$-, and $L_2$-norms at $4.0 \times 10^{-6}\,\mathrm{s}$ and convergence rates between consecutive grid resolutions for the convection-diffusion-reaction case.

| **Var** | $L_\#$**-norm** | **256×8** | **Rate** | **512×16** | **Rate** | **1024×32** | **Rate** | **2048×64** |
|---|---|---|---|---|---|---|---|---|
| $\rho$ | $L_\infty$ | $7.074 \times 10^{-4}$ | 1.856 | $1.954 \times 10^{-4}$ | 3.335 | $1.937 \times 10^{-5}$ | 3.887 | $1.309 \times 10^{-6}$ |
| | $L_1$ | $1.716 \times 10^{-5}$ | 2.763 | $2.527 \times 10^{-6}$ | 3.713 | $1.927 \times 10^{-7}$ | 3.887 | $1.303 \times 10^{-8}$ |
| | $L_2$ | $8.796 \times 10^{-5}$ | 2.410 | $1.655 \times 10^{-5}$ | 3.511 | $1.451 \times 10^{-6}$ | 3.908 | $9.669 \times 10^{-8}$ |
| $\rho u$ | $L_\infty$ | $5.941 \times 10^{-1}$ | 1.638 | $1.909 \times 10^{-1}$ | 3.181 | $2.106 \times 10^{-2}$ | 3.905 | $1.405 \times 10^{-3}$ |
| | $L_1$ | $1.576 \times 10^{-2}$ | 2.642 | $2.525 \times 10^{-3}$ | 3.660 | $1.997 \times 10^{-4}$ | 3.898 | $1.339 \times 10^{-5}$ |
| | $L_2$ | $8.029 \times 10^{-2}$ | 2.283 | $1.650 \times 10^{-2}$ | 3.459 | $1.500 \times 10^{-3}$ | 3.894 | $1.009 \times 10^{-4}$ |
| $\rho e$ | $L_\infty$ | $1.657 \times 10^{3}$ | 1.878 | $4.509 \times 10^{2}$ | 3.285 | $4.625 \times 10^{1}$ | 3.901 | $3.097 \times 10^{0}$ |
| | $L_1$ | $3.924 \times 10^{1}$ | 2.734 | $5.890 \times 10^{0}$ | 3.696 | $4.543 \times 10^{-1}$ | 3.894 | $3.056 \times 10^{-2}$ |
| | $L_2$ | $2.032 \times 10^{2}$ | 2.395 | $3.865 \times 10^{1}$ | 3.500 | $3.416 \times 10^{0}$ | 3.906 | $2.279 \times 10^{-1}$ |
| $\rho c_{\mathrm{H_2}}$ | $L_\infty$ | $1.965 \times 10^{-5}$ | 1.544 | $6.738 \times 10^{-6}$ | 3.052 | $8.121 \times 10^{-7}$ | 3.719 | $6.166 \times 10^{-8}$ |
| | $L_1$ | $3.837 \times 10^{-7}$ | 2.151 | $8.639 \times 10^{-8}$ | 3.350 | $8.473 \times 10^{-9}$ | 3.910 | $5.637 \times 10^{-10}$ |
| | $L_2$ | $1.967 \times 10^{-6}$ | 1.830 | $5.531 \times 10^{-7}$ | 3.253 | $5.803 \times 10^{-8}$ | 3.848 | $4.029 \times 10^{-9}$ |
| $\rho c_{\mathrm{O_2}}$ | $L_\infty$ | $4.895 \times 10^{-4}$ | 1.751 | $1.454 \times 10^{-4}$ | 3.205 | $1.577 \times 10^{-5}$ | 3.940 | $1.027 \times 10^{-6}$ |
| | $L_1$ | $1.111 \times 10^{-5}$ | 2.516 | $1.943 \times 10^{-6}$ | 3.546 | $1.664 \times 10^{-7}$ | 3.913 | $1.105 \times 10^{-8}$ |
| | $L_2$ | $5.638 \times 10^{-5}$ | 2.203 | $1.224 \times 10^{-5}$ | 3.386 | $1.171 \times 10^{-6}$ | 3.909 | $7.792 \times 10^{-8}$ |
| $\rho c_{\mathrm{H}}$ | $L_\infty$ | $1.327 \times 10^{-5}$ | 2.600 | $2.189 \times 10^{-6}$ | 3.900 | $1.466 \times 10^{-7}$ | 3.746 | $1.093 \times 10^{-8}$ |
| | $L_1$ | $1.814 \times 10^{-7}$ | 2.933 | $2.374 \times 10^{-8}$ | 3.848 | $1.649 \times 10^{-9}$ | 3.900 | $1.104 \times 10^{-10}$ |
| | $L_2$ | $1.161 \times 10^{-6}$ | 2.791 | $1.678 \times 10^{-7}$ | 3.850 | $1.164 \times 10^{-8}$ | 3.923 | $7.674 \times 10^{-10}$ |
| $\rho c_{\mathrm{O}}$ | $L_\infty$ | $1.569 \times 10^{-4}$ | 3.164 | $1.751 \times 10^{-5}$ | 4.183 | $9.641 \times 10^{-7}$ | 3.675 | $7.549 \times 10^{-8}$ |
| | $L_1$ | $1.987 \times 10^{-6}$ | 3.766 | $1.460 \times 10^{-7}$ | 3.992 | $9.177 \times 10^{-9}$ | 3.559 | $7.789 \times 10^{-10}$ |
| | $L_2$ | $1.377 \times 10^{-5}$ | 3.515 | $1.204 \times 10^{-6}$ | 4.178 | $6.654 \times 10^{-8}$ | 3.547 | $5.691 \times 10^{-9}$ |
| $\rho c_{\mathrm{OH}}$ | $L_\infty$ | $9.613 \times 10^{-5}$ | 2.550 | $1.642 \times 10^{-5}$ | 3.822 | $1.161 \times 10^{-6}$ | 3.621 | $9.434 \times 10^{-8}$ |
| | $L_1$ | $1.495 \times 10^{-6}$ | 3.250 | $1.572 \times 10^{-7}$ | 4.206 | $8.517 \times 10^{-9}$ | 3.631 | $6.873 \times 10^{-10}$ |
| | $L_2$ | $8.836 \times 10^{-6}$ | 2.877 | $1.203 \times 10^{-6}$ | 4.015 | $7.437 \times 10^{-8}$ | 3.664 | $5.868 \times 10^{-9}$ |
| $\rho c_{\mathrm{HO_2}}$ | $L_\infty$ | $2.457 \times 10^{-7}$ | 2.209 | $5.314 \times 10^{-8}$ | 3.448 | $4.868 \times 10^{-9}$ | 3.671 | $3.823 \times 10^{-10}$ |
| | $L_1$ | $8.672 \times 10^{-9}$ | 3.033 | $1.059 \times 10^{-9}$ | 3.622 | $8.604 \times 10^{-11}$ | 3.752 | $6.388 \times 10^{-12}$ |
| | $L_2$ | $3.553 \times 10^{-8}$ | 2.770 | $5.210 \times 10^{-9}$ | 3.604 | $4.285 \times 10^{-10}$ | 3.915 | $2.841 \times 10^{-11}$ |
| $\rho c_{\mathrm{H_2O_2}}$ | $L_\infty$ | $3.829 \times 10^{-7}$ | 3.372 | $3.697 \times 10^{-8}$ | 4.534 | $1.596 \times 10^{-9}$ | 3.915 | $1.058 \times 10^{-10}$ |
| | $L_1$ | $5.394 \times 10^{-9}$ | 4.087 | $3.173 \times 10^{-10}$ | 4.315 | $1.594 \times 10^{-11}$ | 3.893 | $1.073 \times 10^{-12}$ |
| | $L_2$ | $3.666 \times 10^{-8}$ | 3.811 | $2.613 \times 10^{-9}$ | 4.531 | $1.130 \times 10^{-10}$ | 4.024 | $6.948 \times 10^{-12}$ |
| $\rho c_{\mathrm{H_2O}}$ | $L_\infty$ | $2.507 \times 10^{-4}$ | 2.202 | $5.449 \times 10^{-5}$ | 3.841 | $3.804 \times 10^{-6}$ | 4.044 | $2.306 \times 10^{-7}$ |
| | $L_1$ | $4.560 \times 10^{-6}$ | 2.970 | $5.819 \times 10^{-7}$ | 4.102 | $3.388 \times 10^{-8}$ | 3.921 | $2.236 \times 10^{-9}$ |
| | $L_2$ | $2.633 \times 10^{-5}$ | 2.647 | $4.203 \times 10^{-6}$ | 3.977 | $2.670 \times 10^{-7}$ | 3.999 | $1.670 \times 10^{-8}$ |

## 6.2 Validation

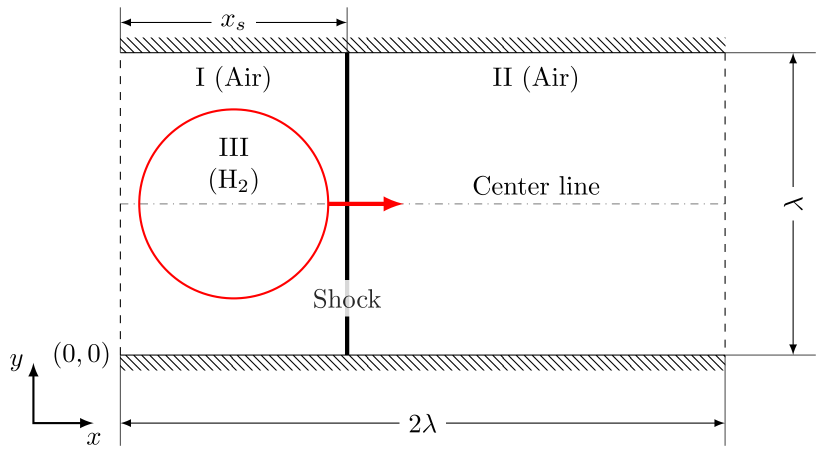


**Figure 6.3:** The shock bubble case setup: an $H_2$-bubble in air is advected through a standing shock of $\mathrm{Ma} = 2$.

To further validate Chord's ARK4 time stepping with reacting flows and shock waves, a two-dimensional $H_2$ bubble is convected through a shock. The case geometry and initial conditions are shown in Fig. 6.3, which replicates the geometry and initial conditions of Owen et al. [3], where the standard ERK4 method was used to validate Chord against Billet et al. [93] and Attal et al. [94]. As such, the new ARK4 time integration method is compared against the pressure profile published by Owen et al. as well as the results from Chord's existing standard ERK4 time integration. Slip walls bound the upper and lower boundaries of the $y$-direction, while extrapolated boundary conditions are used on the left and right $x$-direction boundaries.

A Mach 2 steady planar shock is located $0.75\,\mathrm{cm}$ to the right of the origin and parallel to the $y$-axis, and the hydrogen bubble is located just upstream of the shock. The initial velocities $U_{\mathrm{I}} = U_{\mathrm{III}} = 1.24 \times 10^5\,\mathrm{cm\,s^{-1}}$ and $U_{\mathrm{II}} = 4.34 \times 10^4\,\mathrm{cm\,s^{-1}}$ are the upstream and downstream velocities of the shock. For the reaction mechanism, the same $H_2$–$O_2$ mechanism in Section 6.1 is used for all shock bubble cases. The hydrogen mass fraction is initialized to

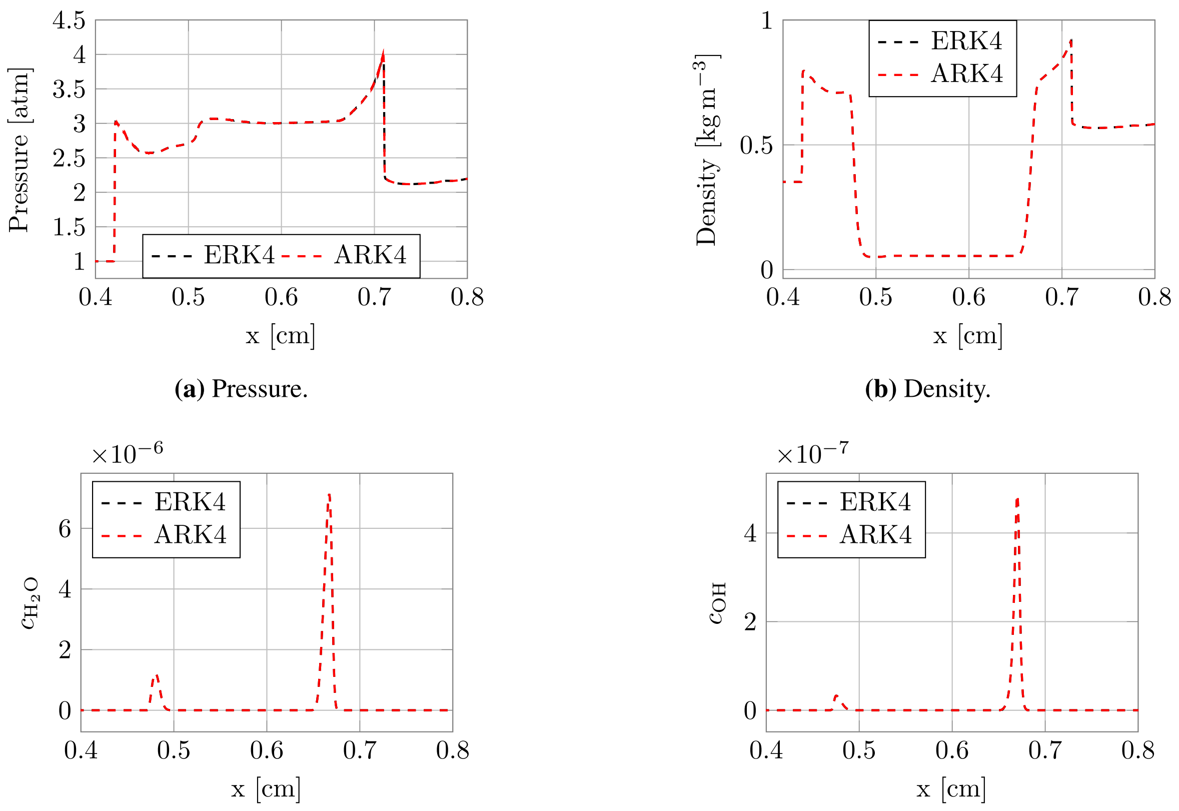
$$c_{\mathrm{H_2}} = \frac{1}{2}\left[1 + \tanh\left(\frac{r_c - r}{C_2}\right)\right], \quad r = \sqrt{((x - x_0)^2 + (y - y_0)^2)}, \tag{6.1}$$

with $r_c$ the radius of the bubble and the coordinate $(x_0, y_0)$ the center of the bubble. The coefficient $C_2$ determines the sharpness of the interface between the $\mathrm{H_2}$ bubble and the surrounding air. The case parameters are $C_2 = 3 \times 10^{-3}\,\mathrm{cm}^{-1}$, $r_c = 0.28\,\mathrm{cm}$, and $(x_0, y_0) = (0.4, 0.75)\mathrm{cm}$. The mass fractions for the surrounding air are set to $c_{\mathrm{N_2}} = 0.767$ and $c_{\mathrm{O_2}} = 0.233$ both upstream and downstream of the shock.

This case uses the same base resolution and AMR levels as in Owen et al. [3], that is, a base resolution of $1024 \times 512$ with three levels of AMR and a refinement ratio of 2 for each level. With AMR, the inter-level ARK4 operations are validated. The grids are refined based on gradients of density and pressure. The case is run to a solution time of $t = 10\,\mathrm{\mu s}$ and solution profiles are taken along the center line marked in Fig. 6.3.

The pressure profile by ARK4, shown in Fig. 6.4a, tracks nearly exactly to both Owen et al. [3] and the ERK4 case, with only a slight over-prediction of pressure at the right reflected shock. Density in Fig. 6.4b is also in nearly identical agreement, with a reciprocating slight under-prediction in the right reflected shock. Shown in Fig. 6.4c are the traces of $\mathrm{H_2O}$ mass fraction found from ARK4 and ERK4. There is nearly identical agreement between the two methods. Likewise, the mass fraction of OH found from ARK4 matches well with the ERK4 case, as seen in Fig. 6.4d. This validates the accurate predictions by ARK4.

The evolution of the solution over time for three different cases is shown in Fig. 6.5. In addition to the reacting ARK4 and ERK4 cases, a non-reacting ERK4 case is run in order to observe the effects of reactions on the acoustic waves. The figures show pressure contour lines ranging from $1\,\mathrm{atm}$ to $7.37\,\mathrm{atm}$. Overlaid on the pressure contours are shadings of $\mathrm{H_2}$ mass fractions. In the first row, Figs. 6.5a–6.5c show the solution at time $t = 1.5\,\mathrm{\mu s}$. The initial $\mathrm{H_2}$ mass fraction bubble has started to impinge on the standing shock and is being compressed, while reflected, refracted, and transmitted waves are being produced. In the second row, solutions at time $t = 3.5\,\mathrm{\mu s}$ are shown in Figs. 6.5d–6.5f. A right reflected shock forms downstream of the $\mathrm{H_2}$ bubble and a left reflected wave forms inside the $\mathrm{H_2}$ bubble. Also seen in all three cases is the secondary transmitted wave forming to the left of the $\mathrm{H_2}$ bubble. Lastly, in the bottom row, Figs. 6.5g–6.5i show the



**(a)** Pressure.

**(b)** Density.

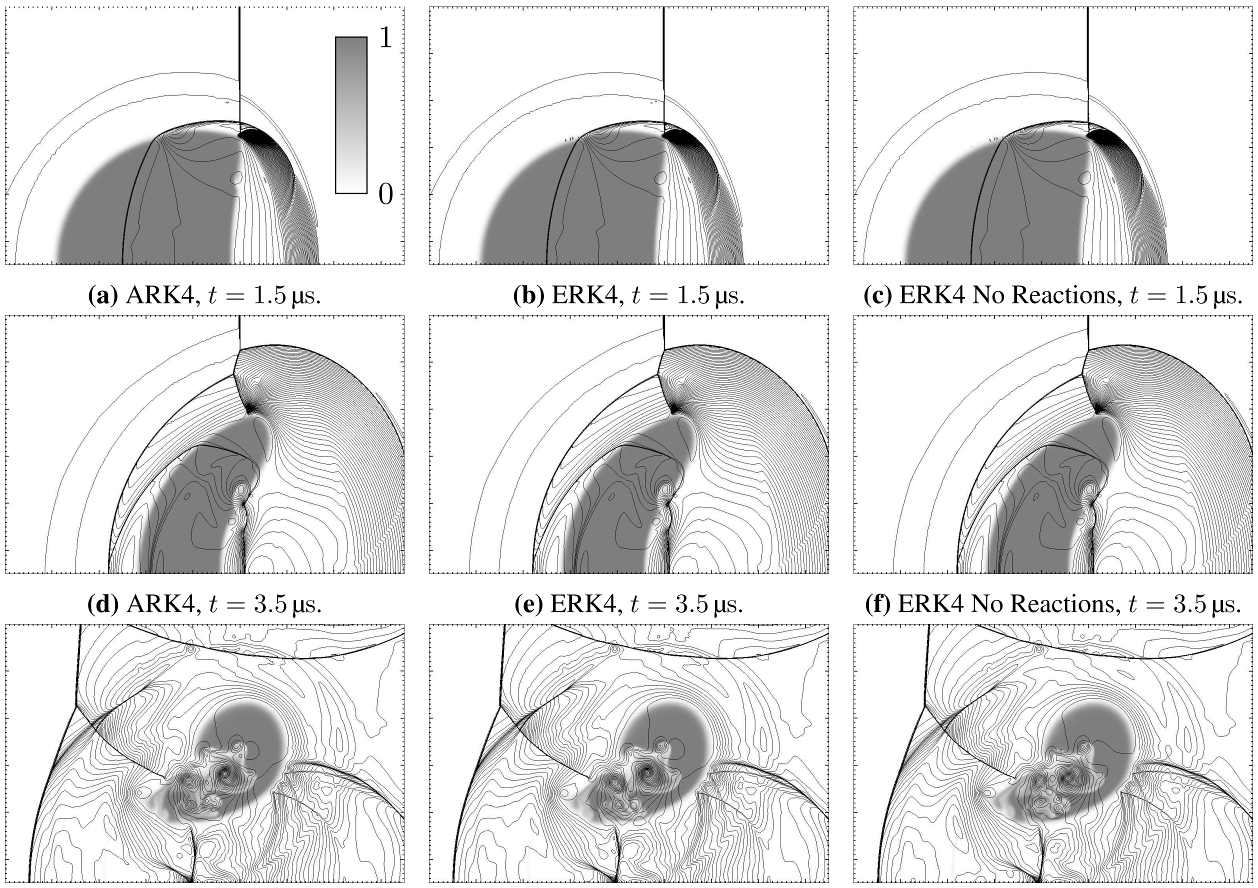
**(c)** $H_2O$ mass fraction.

**(d)** OH mass fraction.

**Figure 6.4:** Pressure, density, $H_2O$ mass fraction, and OH mass fraction profiles along the center line at $t = 3.5\,\mu s$ for the shock bubble case with three AMR levels.

solutions at time $t = 10\,\mu s$. A wave is reflected from the top boundary, and two counter-rotating vortices have developed in the $H_2$ bubble. At all three solution times, the ARK4 and ERK4 pressure waves are nearly identically located, while the no-reaction case pressure waves slightly lag behind the pressure waves of the reaction cases. This indicates that the reactions are driving the pressure waves forward, as expected, and that ARK4 does correctly model the reactions driving the pressure waves.

The comparison of time step sizes for the inertial, viscous, chemical, and total time step size is shown in Fig. 6.6. In these figures, level 0 corresponds to the coarsest mesh resolution and level 2 corresponds to the finest mesh resolution. Due to the subcycling algorithm, the finer AMR levels require more steps than coarser level, in this case at a ratio of 2 per level. The $x$-axes of levels 1 and 2 have been scaled to the same length as the level 0 $x$-axis in order to correctly align each



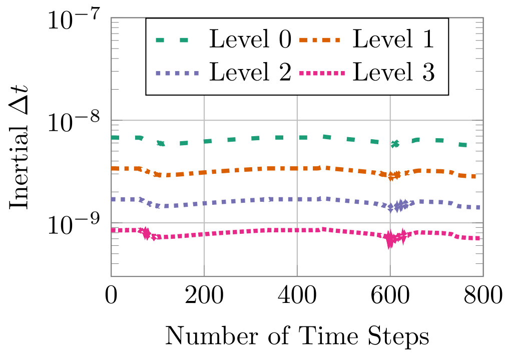
**(a)** ARK4, $t = 1.5\,\mu s$. **(b)** ERK4, $t = 1.5\,\mu s$. **(c)** ERK4 No Reactions, $t = 1.5\,\mu s$.

**(d)** ARK4, $t = 3.5\,\mu s$. **(e)** ERK4, $t = 3.5\,\mu s$. **(f)** ERK4 No Reactions, $t = 3.5\,\mu s$.

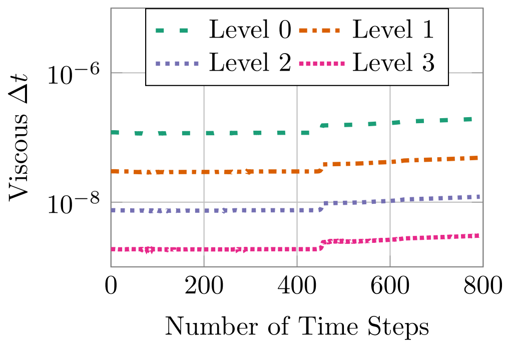
**(g)** ARK4, $t = 10\,\mu s$. **(h)** ERK4, $t = 10\,\mu s$. **(i)** ERK4 No Reactions, $t = 10\,\mu s$.

**Figure 6.5:** Pressure contour lines ($1\,\mathrm{atm}$ to $7.37\,\mathrm{atm}$) superimposed on $c_{\mathrm{H}_2}$ (grayscale) obtained by ARK4 and ERK4 with reactions, respectively, in addition to ERK4 with no reactions. These shock bubble cases are run with three levels of AMR.

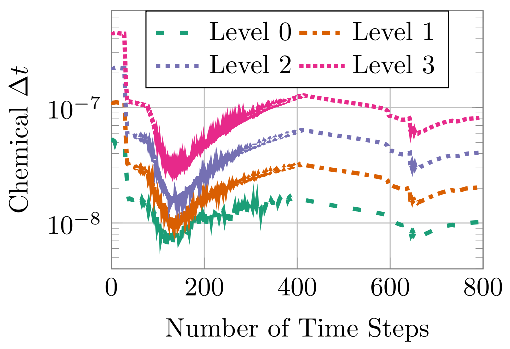
level's time step number with the solution time. On the finer levels, the inertial step size decreases proportional to the mesh spacing, as required by the CFL condition. Likewise, the von Neumann condition can be seen with the viscous step size decreasing quadratically to the mesh spacing. The chemical time step size increases as the mesh is refined, and varies over time as the reactions begin and then stabilize over time. The ARK4 time step size represents the time step size computed from Eqn. (4.16). For this high-resolution shock bubble case, the inertial time step size is smaller than the chemical time step size and so ARK4 cannot provide a speedup over ERK4 as expected. Again, the purpose of this case is to demonstrate that ARK4 properly resolves the strong shock waves, flame fronts, and chemical reactions.



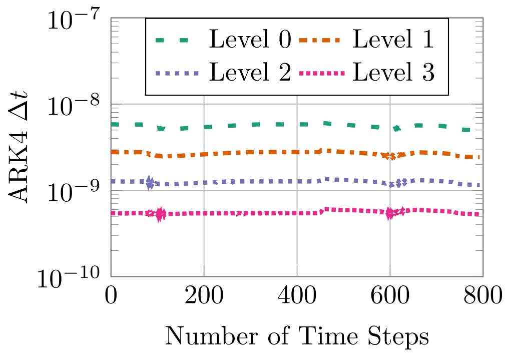
**(a)** Inertial $\Delta t$.



**(b)** Viscous $\Delta t$.



**(c)** Chemical $\Delta t$.



**(d)** ARK4 $\Delta t$.

**Figure 6.6:** The step size found from inertial, viscous, and chemical time scales on the base level and three AMR levels of the 2D shock bubble case, along with the step size taken by ARK4. Due to subcycling, the finer levels take more steps than the coarser level at a ratio of 2 per level.

# Chapter 7

# Results and Discussion of ARK4 with AMR

The ARK4+AMR algorithm is now applied to solve reacting flows in a bluff-body combustor, a case that is near-intractable for explicit time stepping. Research into aerospace combustors is accelerating due to a demand for increased efficiency and fewer emissions, but the complexity of fluid and combustion interactions in practical applications provides a significant challenge to numerical combustion. The efficient numerical techniques are tested by using the bluff-body combustor as a representative for practical combustors. Common physical processes are sufficiently represented, such as shear layers, a region of recirculation behind the flame holder with geometric complexity, volumetric expansion in the wake, and complex thermoacoustic instabilities [90, 95].

## 7.1 Combustion in 2D Bluff-Body Combustor

The bluff body, as shown in Fig. 7.1 by the equilateral triangle shaded in gray, sits in a straight channel with three sections of interest: the inlet, the combustor, and the outlet. At the inlet, a gaseous mixture consisting of $4.01\%$ $C_3H_8$, $22.36\%$ $O_2$, and $73.62\%$ $N_2$ by mass fraction flows into the domain at a velocity of $15.7\,\mathrm{m\,s^{-1}}$. The mixture has a temperature of $310\,\mathrm{K}$ and a pressure of $101\,325\,\mathrm{Pa}$. The initial conditions in the domain are set to the same values as the inlet, except for a small region around the bluff body. Near the bluff body, the gas mixture is initialized to

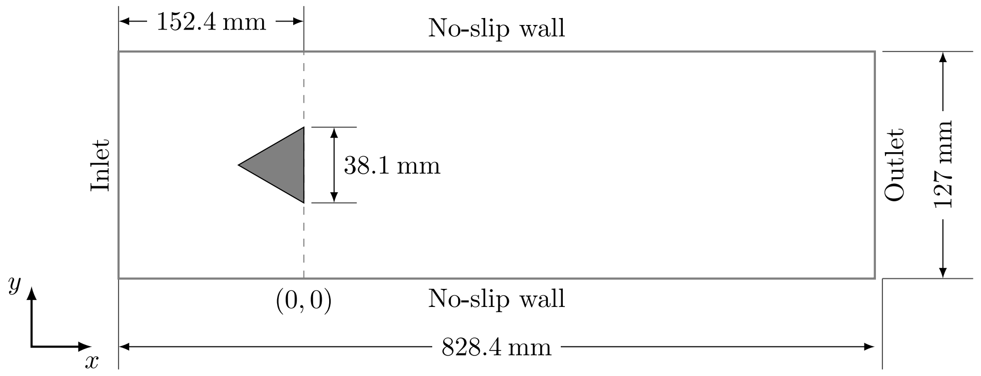
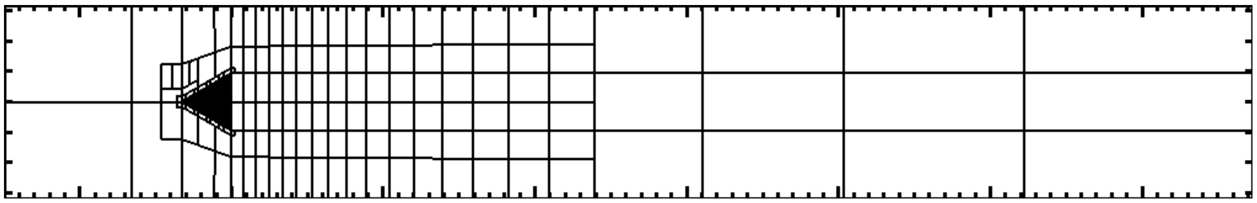
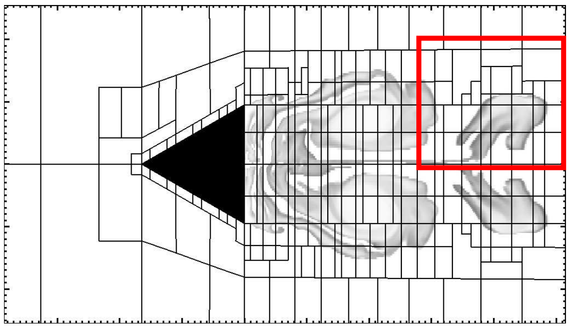


**Figure 7.1:** A diagram of the bluff-body combustor geometry.

$12\%$ $CO_2$, $6.54\%$ $H_2O$, $5.14\%$ $O_2$, and $73.62\%$ $N_2$, with a temperature of $1300\,\mathrm{K}$ and a pressure of $106\,661\,\mathrm{Pa}$. This hot spot acts as the ignition to initiate reactions. The velocity near the bluff body is the same as the initial conditions.


**(a)** The initial grid boxes for the bluff-body combustor.


**(b)** Grid boxes showing dynamic adaptation to the solution at $t = 5.2\,\mathrm{ms}$.

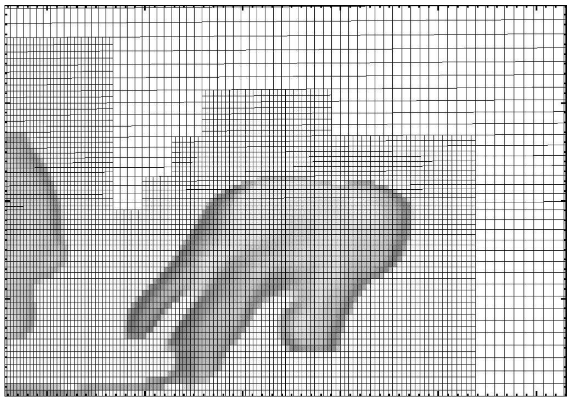

**(c)** At $t = 5.2\,\mathrm{ms}$ the cells are dynamically refined for chemically reacting regions. This is a close up view of the cells in the region outlined in red in Fig. 7.2b.

**Figure 7.2:** In the physical domain: grids at the initial conditions and after some solution time.

A base mesh of $11\,008$ cells is used, with the following cases using various AMR levels. The Cartesian computational domain is transformed from the physical domain using the MMB technique, enabling Chord to handle relatively complex geometries, such as the bluff body, while efficiently using the finite-volume algorithm on a Cartesian grid with AMR. Cells are grouped into grid boxes, and grid boxes are distributed across processors to achieve spatial parallelization. The initial grid boxes are shown in Fig. 7.2a, with the domain comprised of seven mapped blocks. During time integration, the AMR algorithm tags regions in the domain based on the values of OH and $CH_2O$ mass fractions, subject to $c_{\mathrm{OH}} \times c_{\mathrm{CH_2O}} > 2.0 \times 10^{-9}$. A representative set of grid boxes

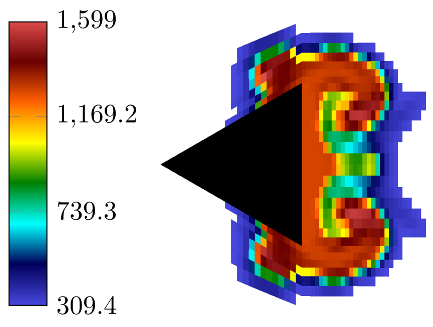
**Table 7.1:** A comparison between ARK4 and ERK4 for the average step size and average wall-clock time per step (in seconds) for the 2D bluff-body combustor.

| Case | Average $\Delta t$ | Average Wall-Clock/Step | Speedup |
|---|---|---|---|
| ERK4 | $2.5 \times 10^{-9}$ | 0.091 | – |
| ARK4 | $1.0 \times 10^{-6}$ | 0.521 | 70 |

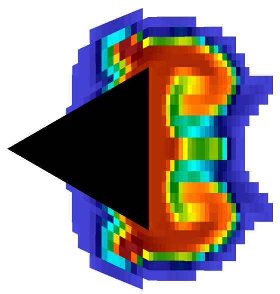
at $t = 5.2\,\mathrm{ms}$ is shown in Fig. 7.2b, and the individual cells may be seen at this solution time in Fig. 7.2c in a close-up view.

For all cases considered here, neither the PID step-size controller nor the extrapolation for initial stage value guesses in the ARK4 scheme is invoked. The PID step-size controller is found unnecessary to control the stability, while in some cases, extrapolating initial stage value guesses appears to prevent the nonlinear solver from converging. Regarding the nonlinear convergence tolerance, it is set to $\epsilon_{\mathrm{NLS}} = 1.0 \times 10^{-8}$, but is scaled by $J\rho$ which leads to a typical value of $\epsilon_{\mathrm{nls}} \approx 1.0 \times 10^{-14}$. A very tight convergence tolerance is required to ensure a consistent thermodynamic state in a cell.

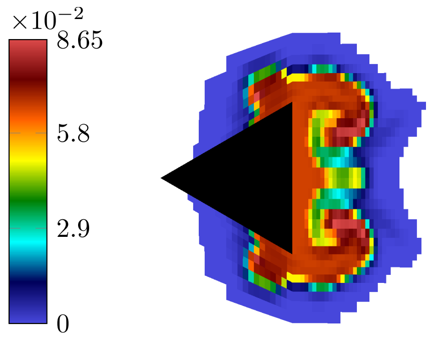
First, two cases, one with ARK4 and the other with ERK4, are run to a solution time of $t = 1\,\mathrm{ms}$. For a fair comparison, both cases use the same base grid. Results are presented to compare the solution accuracy and efficiency between the ERK4 and ARK4 time integration methods. Statistical data on the average time step size and average wall-clock time per step is collated in Table 7.1. This demonstrates that the ARK4 time integration is able to take time step sizes of approximately three orders of magnitude larger than the ERK4 time integration, with a wall clock time of approximately half an order of magnitude longer per time step. This leads to an average speedup of $70\times$ by using ARK4 for the bluff body problem while providing an acceptable solution accuracy. The ERK4 and ARK4 time integrators require $310\,000$ steps and $1200$ steps, respectively, to reach the same solution time. Figure 7.3 compares the contours of temperature, $c_{\mathrm{H_2O}}$, and $c_{\mathrm{OH}}$ in a region immediately behind the bluff body where the flow and flame dynamics is important. The difference in the temperature contours between the two time integrators is negligible for both the structure and the magnitude. The $c_{\mathrm{H_2O}}$ contours are also nearly indistinguishable. Although the difference between the $c_{\mathrm{OH}}$ contours is visible in a few spatial locations, the structures remain identical. The visible difference is not a concern because $\mathrm{OH}$ variation over time is much more dynamic than other species and its impact on the flame dynamics overall seems to be short-lived. The difference in a mean distribution over a characteristic timescale is significantly less than that shown in an instantaneous snapshot. The ARK4 time integrator achieves a $70\times$ times speed-up for this test.



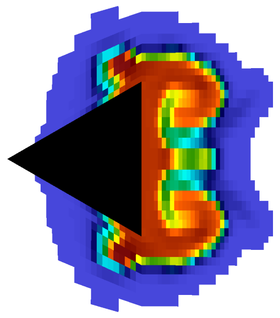
**(a)** Temperature for ARK4 after 1,200 steps.



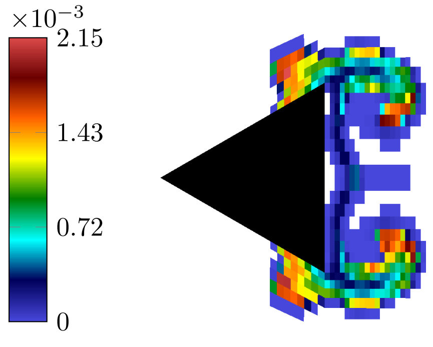
**(b)** Temperature for ERK4 after 310,000 steps.



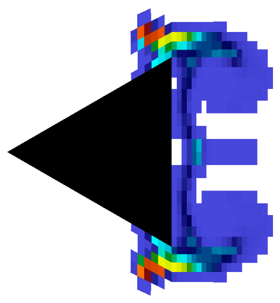
**(c)** $H_2O$ mass fraction for ARK4 after 1,200 steps.



**(d)** $H_2O$ mass fraction for ERK4 after 310,000 steps.



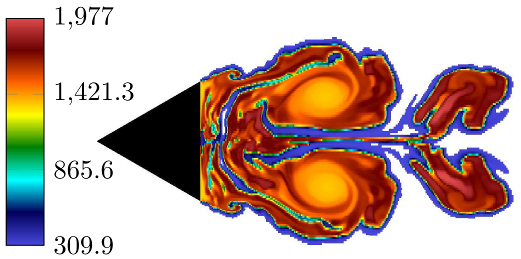
**(e)** OH mass fraction for ARK4 after 1,200 steps.



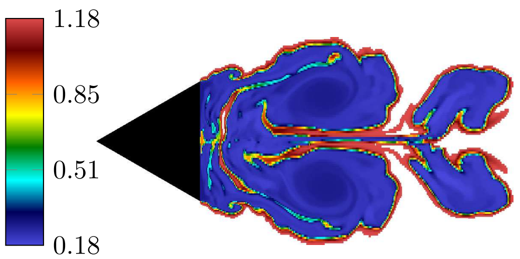
**(f)** OH mass fraction for ERK4 after 310,000 steps.

**Figure 7.3:** Distributions of temperature, $H_2O$ mass fraction, and OH mass fraction behind the bluff body, comparing ARK4 and ERK4 at $t = 1\,\mathrm{ms}$ for the $C_3H_8$-air chemistry.

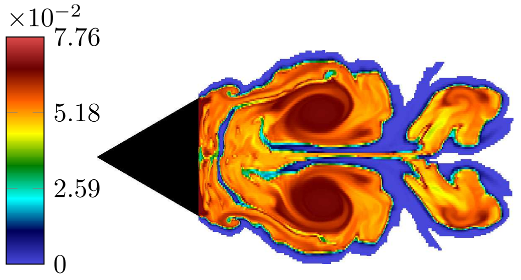
The case with the ARK4 time integrator was further advanced to a solution time of $t = 5.2\,\mathrm{ms}$, while two additional levels of AMR was employed with a refinement ratio of 2 for each level. Figure 7.4 shows the temperature, density, $c_{\mathrm{H_2O}}$, and $c_{\mathrm{OH}}$ contours at the same physical region as that in Fig. 7.3. As seen in Shanbhogue et al. [96], the recirculation and the beginning of the generation of flame wrinkling are clearly observed. Using AMR, the flow and flame details are efficiently resolved while greatly reducing the computational cost for this ARK4 case. Without AMR, the base grid did not resolve the fine structure. Without AMR, it is simply not affordable for a uniformly refined grid that has the same finest resolution as it in this AMR case,



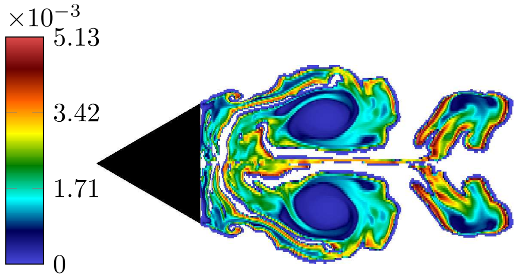
**(a)** Temperature, ARK4.



**(b)** Density, ARK4.



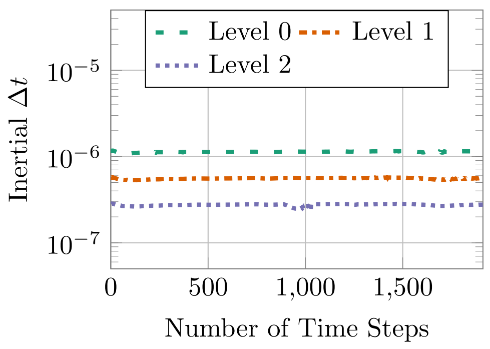
**(c)** $H_2O$ mass fraction, ARK4.



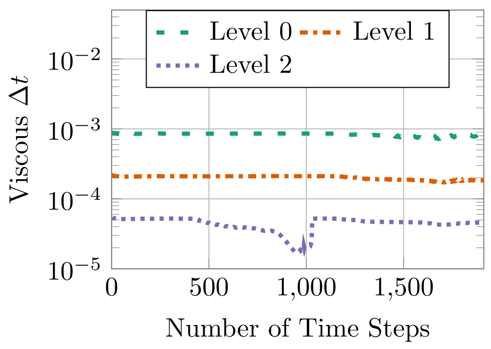
**(d)** OH mass fraction, ARK4.

**Figure 7.4:** Distributions of temperature, density, $H_2O$ mass fraction, and OH mass fraction behind the bluff body at $t = 5.2\,\mathrm{ms}$ for the $C_3H_8$-air combustion with two levels of AMR.

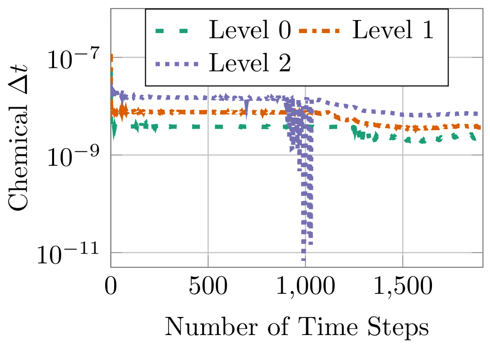
More interestingly, the time step sizes for the convective, diffusive, and reactive physics on each AMR level are shown in Fig. 7.5. As shown in Fig. 7.5a, the inertial time step size decreases proportionally to the mesh resolution as expected from the CFL condition (Eqn. (4.10)). Figure 7.5b shows the viscous time step size decreases quadratically to the mesh resolution as expected from the von Neumann condition (Eqn. (4.11)). As the mesh is refined, the average step



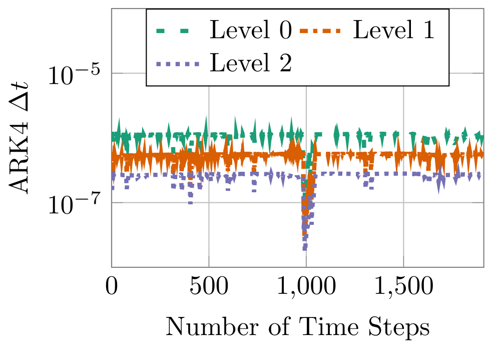
**(a)** Inertial $\Delta t$.



**(b)** Viscous $\Delta t$.



**(c)** Chemical $\Delta t$.



**(d)** ARK4 $\Delta t$.

**Figure 7.5:** The step size found from inertial, viscous, chemical, and ARK4 time scales on the base grid and two AMR levels of the $C_3H_8$-air chemistry in the 2D bluff-body combustor. Due to subcycling, the finer levels take more steps than the coarser level at a ratio of 2 per level.

size for chemical reactions increases as shown by Fig. 7.5c. Nevertheless, the chemical time step is still the limiting one on each AMR level. Lastly, the combined ARK4 time step size, as computed by Eqn. (4.15), is shown in Fig. 7.5d.

Inspired by the observation through numerical experiments that the geometry has demonstrated an impact on chemistry stiffness, the impact on stiffness of the $H_2$-air combustion in the bluff body configuration is compared to the shock bubble configuration where there is no geometric complexity. Figure 7.6 shows the evolution of the inertial (convective), viscous, chemical, and ARK4 time-step sizes over the solution time and the AMR level. Similarly to the $C_3H_8$-air combustion, the inertial and viscous time step sizes for the $H_2$-air flame decrease proportionally to the mesh spacing and mesh spacing squared, respectively. The same is also observed that the chemical step size increases as the spatial resolution increases. However, the evolution of the chemical time step

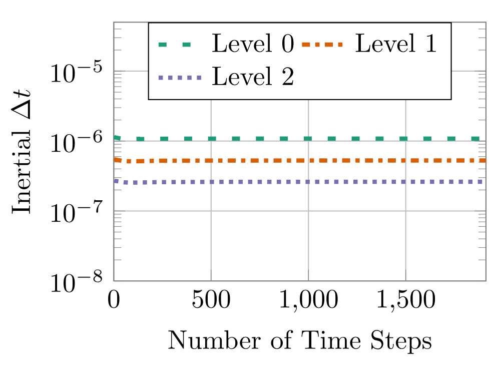
**Table 7.2:** The average $\Delta t$ (in seconds) for the $H_2$-air and $C_3H_8$-air chemistry for the shock bubble and bluff body cases.

| Case | Level | ARK4 $\Delta t$ | Inertial $\Delta t$ | Viscous $\Delta t$ | Chemical $\Delta t$ |
|---|---|---|---|---|---|
| Shock bubble ($H_2$-air) | 0 | $5.6 \times 10^{-9}$ | $6.4 \times 10^{-9}$ | $1.4 \times 10^{-7}$ | $1.4 \times 10^{-8}$ |
| | 1 | $2.7 \times 10^{-9}$ | $3.2 \times 10^{-9}$ | $3.5 \times 10^{-8}$ | $2.6 \times 10^{-8}$ |
| | 2 | $1.3 \times 10^{-9}$ | $1.6 \times 10^{-9}$ | $8.9 \times 10^{-9}$ | $5.0 \times 10^{-8}$ |
| | 3 | $5.5 \times 10^{-10}$ | $8.0 \times 10^{-10}$ | $2.2 \times 10^{-9}$ | $9.8 \times 10^{-8}$ |
| Bluff body ($H_2$-air) | 0 | $1.8 \times 10^{-7}$ | $1.1 \times 10^{-6}$ | $7.9 \times 10^{-4}$ | $8.8 \times 10^{-12}$ |
| | 1 | $1.0 \times 10^{-7}$ | $5.3 \times 10^{-7}$ | $1.9 \times 10^{-4}$ | $9.6 \times 10^{-9}$ |
| | 2 | $5.8 \times 10^{-8}$ | $2.6 \times 10^{-7}$ | $4.8 \times 10^{-5}$ | $2.1 \times 10^{-8}$ |
| Bluff body ($C_3H_8$-air) | 0 | $1.1 \times 10^{-6}$ | $1.1 \times 10^{-6}$ | $8.3 \times 10^{-4}$ | $3.3 \times 10^{-9}$ |
| | 1 | $5.3 \times 10^{-7}$ | $5.6 \times 10^{-7}$ | $2.0 \times 10^{-4}$ | $6.2 \times 10^{-9}$ |
| | 2 | $2.6 \times 10^{-7}$ | $2.75 \times 10^{-7}$ | $4.4 \times 10^{-5}$ | $1.1 \times 10^{-8}$ |

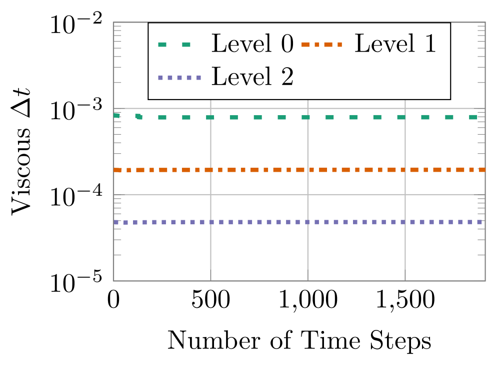
for the $C_3H_8$-air is much less oscillatory than that for the $H_2$-air flame. The overall time step size determined by ARK4 for the former is about two orders of magnitude larger than the latter. Furthermore, the time step size on the base grid (level 0) for the $H_2$-air flame is significantly smaller than that on the finer levels, indicating a strong need for implicit time marching for the chemical source term. The average time step values are summarized in Table 7.2 for the 2D shock bubble with the $H_2$-air mechanism as well as 2D bluff body cases with $H_2$-air and $C_3H_8$-air mechanisms. This demonstrates that the $H_2$-air combustion is more stiff in the bluff-body combustor than in the shock bubble configuration. However, the difference in the stiffness can also be a consequence of the operating conditions. Additionally, $H_2$-air combustion is more stiff than $C_3H_8$-air combustion in the bluff-body combustor.

## 7.2 Combustion in 3D Bluff-Body Combustor

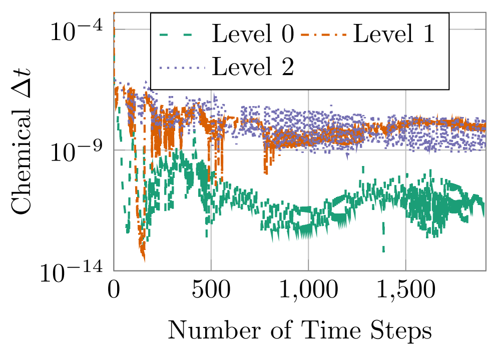
For the 3D bluff body case, the domain size and boundary conditions remain the same as the 2D case, except for the addition of the span-wise direction with a depth of $127\,\mathrm{mm}$. The combustor is extruded to the entire depth. Periodic conditions are used on the boundaries normal to the span-wise direction. The base mesh in the 3D case contains approximately $330\,000$ cells, and one level of refinement is added at a refinement ratio of 2. As with the 2D case, the flow-direction mesh



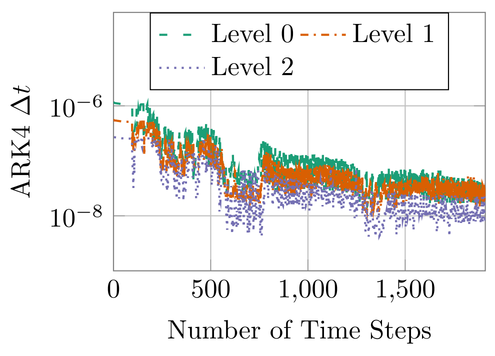
**(a)** Inertial $\Delta t$.



**(b)** Viscous $\Delta t$.



**(c)** Chemical $\Delta t$.

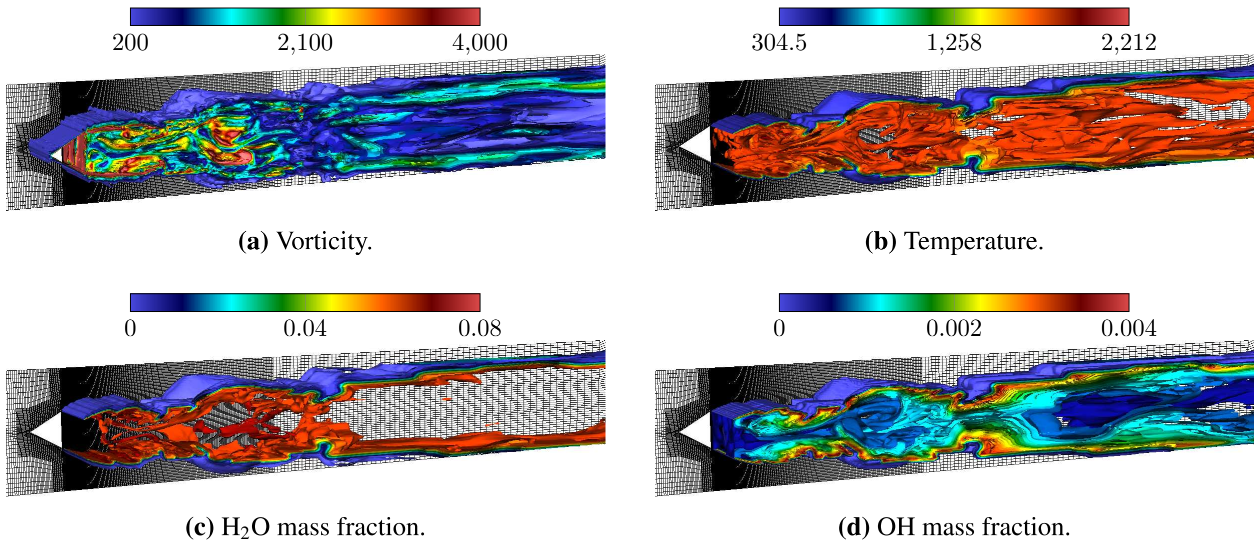


**(d)** ARK4 $\Delta t$.

**Figure 7.6:** The step size found from inertial, viscous, chemical, and ARK4 time scales on the base level and two AMR levels of the $H_2$-air combustion in the 2D bluff-body combustor. Due to subcycling, the finer levels take more steps than the coarser level at a ratio of 2 per level.

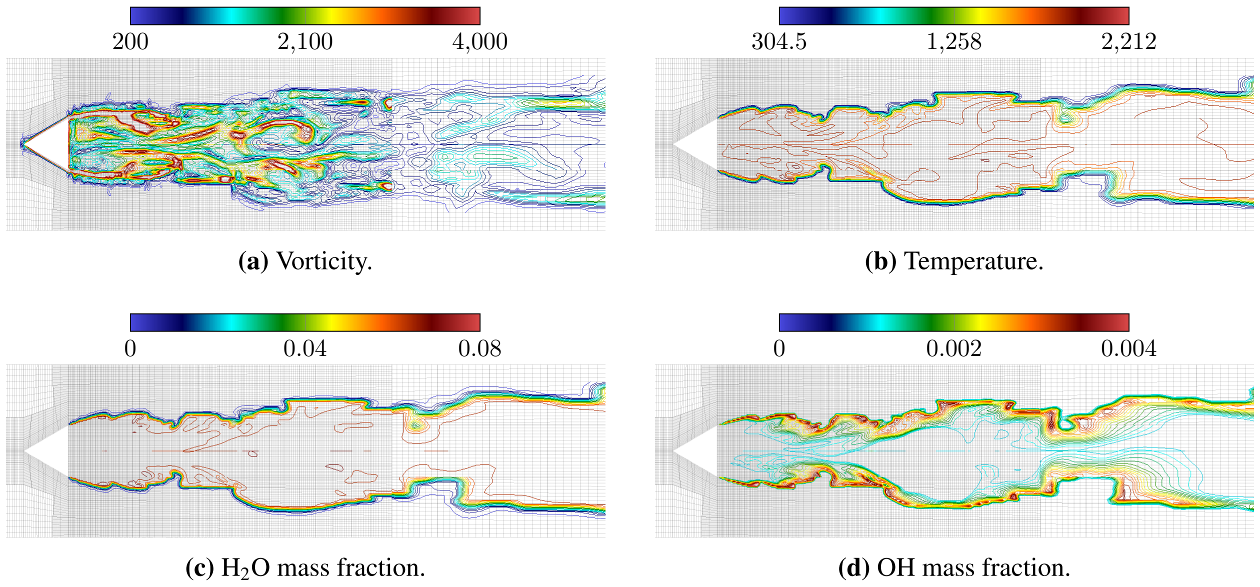
resolution stretched logarithmically from $1\,\mathrm{mm}$ immediately behind the bluff body to $11\,\mathrm{mm}$ at the outlet. A level of mesh refinement is applied to resolve the complex dynamics immediately behind the bluff body. Only the ARK4 time integration is employed for these 3D cases since the computational time of using ERK4 is infeasible. First, the $C_3H_8$-air mechanism is demonstrated, then the $CH_4$-air mechanism is also considered.

Figure 7.7 shows the instantaneous isosurfaces of the vorticity, temperature, and mass fractions of $H_2O$ and OH, respectively, for the 3D $C_3H_8$-air flame at a solution time of $t = 246\,\mathrm{ms}$. The flame is much more developed at $4.9$ flow-through times. Note that a flow-through time is $50\,\mathrm{ms}$. The vorticity contours demonstrate a significant recirculation zone immediately behind the combustor, as well as vortex shedding in the wake. The temperature, $c_{\mathrm{OH}}$, and $c_{\mathrm{H_2O}}$ plots show resolution of the flame front, hot products in the wake, and flame wrinkling.



**(a)** Vorticity. **(b)** Temperature. **(c)** $H_2O$ mass fraction. **(d)** OH mass fraction.

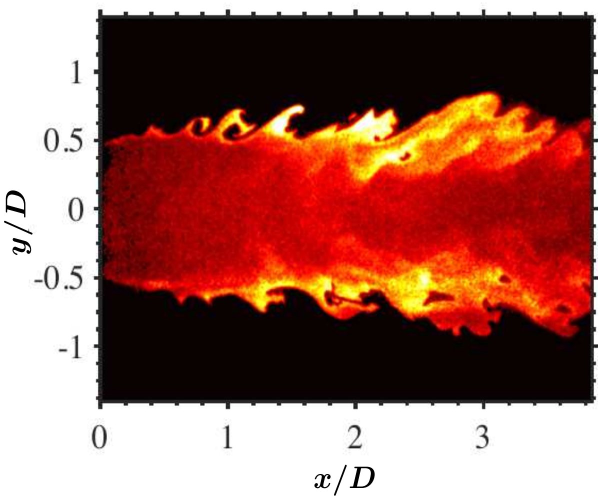
**Figure 7.7:** Vorticity, temperature, $H_2O$ mass fraction, and OH mass fraction of the $C_3H_8$-air flame in the 3D bluff-body combustor at solution time $t = 246\,\mathrm{ms}$.



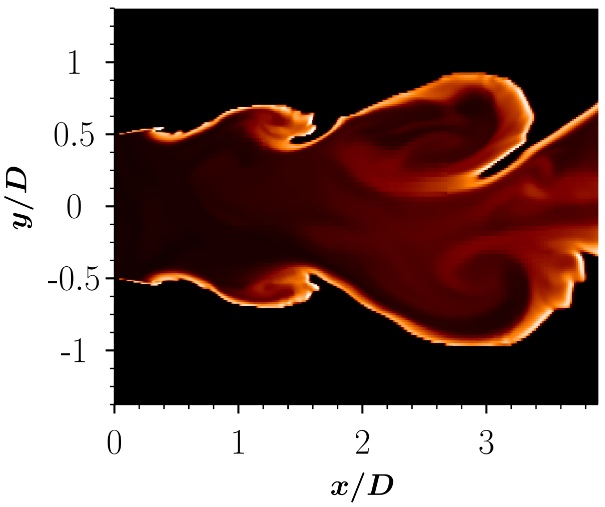
**(a)** Vorticity. **(b)** Temperature. **(c)** $H_2O$ mass fraction. **(d)** OH mass fraction.

**Figure 7.8:** A cross section of the contours of vorticity, temperature, $H_2O$ mass fraction, and OH mass fraction of the $C_3H_8$-air flame in the 3D bluff-body combustor at solution time $t = 246\,\mathrm{ms}$.

While 3D images are more interesting to view than 2D images, the latter is usually easier for visualizing the details using contour lines. To show the contour lines, a mid-$z$-plane is taken from Fig. 7.7, as shown in Fig. 7.8. Several important physical structures can be identified in the 2D contours. Vortex shedding is seen in the vorticity plot, along with the recirculation zone behind



**(a)** Experimental OH-PLIF images reproduced from Fuggar et al. [97].



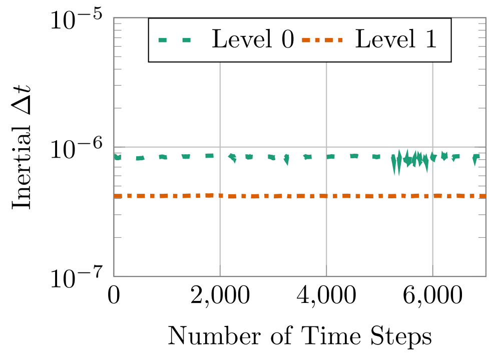
**(b)** Computational results from Chord using ARK4 showing $\rho c_{\mathrm{OH}}$.

**Figure 7.9:** Comparison of instantaneous OH flame structure between experimental results and computational results.

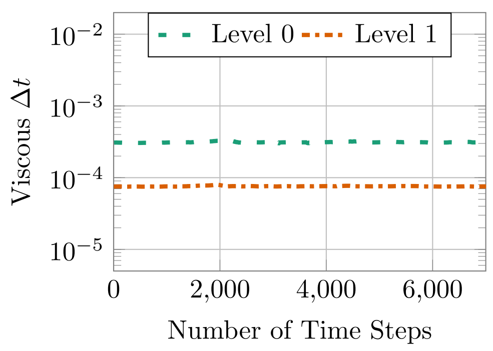
the combustor and its interaction with the shear layer. The temperature plot shows flame wrinkling and the hot wake generated by the combustion.

For a quantitative comparison, consider the comparison between the experimental and the computational results in Fig. 7.9, showing instantaneous OH flame structures. In these figures, the trailing edge of the bluff body is aligned at $x/D = 0$. The flame thickness in the experimental measurement is approximately the same as that predicted by Chord. In addition, the wrinkling feature in the computational flame behaves closely to the experimental flame in the shear layer. Immediately behind the flame holder is the recirculation zone, whose size is on the same order between the experimental and computational flames. Nevertheless, time-averaging profiles would be more insightful for comparison than instantaneous ones. A follow-up study will further explore the time-averaged solutions.

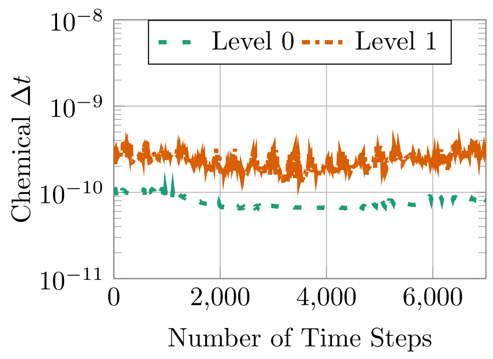
The time step sizes for an AMR bluff-body case with the $C_3H_8$-air mechanism are shown in Fig. 7.10. The inertial time step size decreases as the mesh resolution increases, and the diffusive time step size decreases quadratically to the mesh resolution. As with previous cases, the chemical time step size increases as the mesh resolution increases. For all three physical processes, the time step sizes started decreasing as the solution time advanced.



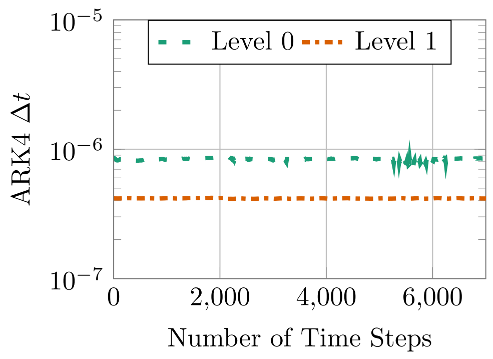
**(a)** Inertial $\Delta t$.



**(b)** Viscous $\Delta t$.



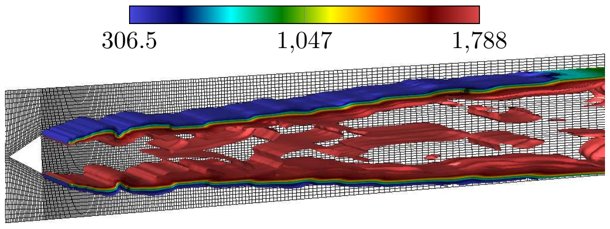
**(c)** Chemical $\Delta t$.



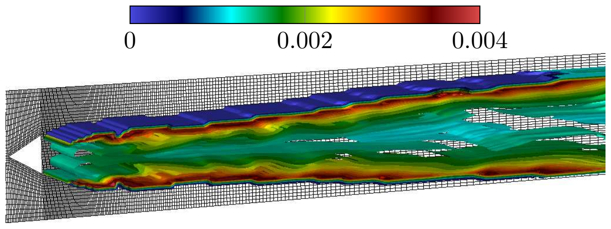
**(d)** ARK4 $\Delta t$.

**Figure 7.10:** The step size found from inertial, viscous, and chemical time scales on the base level and two AMR levels of the 3D bluff body case with $C_3H_8$-air chemistry. Due to subcycling, the finer levels take more steps than the coarser level at a ratio of 2 per level.

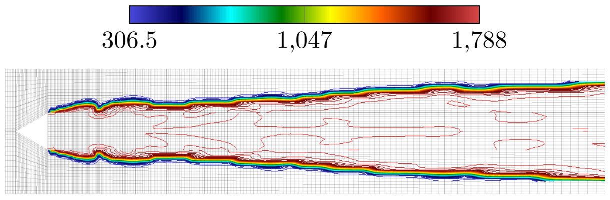
A 13-species, 38-reaction $CH_4$-air mechanism is used to evaluate ARK4 time integration with methane as a fuel [98]. The instantaneous isosurfaces of temperature and OH mass fraction are shown in Fig. 7.11 at a solution time of $t = 834\,\text{ms}$, or 16.7 flow-through times. Interestingly, the chemical time scales for the methane reaction mechanism is of the same order of magnitude as the inertial time scales. Not surprisingly, $CH_4$-air combustion is much less stiff than $C_3H_8$-air combustion. The average time step sizes for the $CH_4$-air mechanism on the 3D bluff body geometry is shown in Table 7.3. On the first level of refinement, the chemical time step size is larger than the inertial time step size, rendering ARK4 ineffective at obtaining speedups. On the coarse level (level 0), the $C_3H_8$-air mechanism takes an average step size of $8.4 \times 10^{-7}\,\text{s}$ while the chemical step size would limit an ERK4 time integration method to an average step size of $7.9 \times 10^{-11}\,\text{s}$.



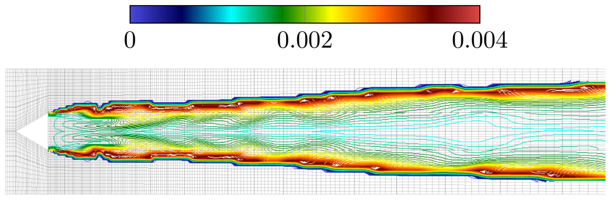
**(a)** Temperature, 3D contour.



**(b)** OH mass fraction, 3D contour.



**(c)** Temperature, 2D cross section.



**(d)** OH mass fraction, 2D cross section.

**Figure 7.11:** Temperature and OH mass fraction of the 3D bluff body $CH_4$-air case at solution time $t = 834\,\mathrm{ms}$.

This corresponds to taking a step size $10\,000\times$ larger in ARK4 than in ERK4. On the fine level, the average step size of $4.2 \times 10^{-7}\,\mathrm{s}$ is $2000\times$ larger than the chemical step size of $2.4 \times 10^{-10}\,\mathrm{s}$.

**Table 7.3:** The average $\Delta t$ (in seconds) for the $C_3H_8$-air and $CH_4$-air combustion in the 3D bluff-body combustor.

| **Case** | **Level** | **ARK4 $\Delta t$** | **Inertial $\Delta t$** | **Viscous $\Delta t$** | **Chemical $\Delta t$** |
|---|---|---|---|---|---|
| $C_3H_8$-air | 0 | $8.4 \times 10^{-7}$ | $8.4 \times 10^{-7}$ | $3.1 \times 10^{-4}$ | $7.9 \times 10^{-11}$ |
| | 1 | $4.2 \times 10^{-7}$ | $4.2 \times 10^{-7}$ | $7.6 \times 10^{-5}$ | $2.4 \times 10^{-10}$ |
| $CH_4$-air | 0 | $8.6 \times 10^{-7}$ | $8.6 \times 10^{-7}$ | $3.4 \times 10^{-4}$ | $1.5 \times 10^{-7}$ |
| | 1 | $4.2 \times 10^{-7}$ | $4.2 \times 10^{-7}$ | $8.2 \times 10^{-5}$ | $3.8 \times 10^{-6}$ |

# Chapter 8

# Adaptive Space-Time Parallel Algorithm

While ARK4 has significantly reduced the time-to-solution for turbulent reacting flows by increasing time step sizes, the number of time steps required to eliminate transients from the solution remains high. To this point, those time steps are solved sequentially by ARK4. This part of the dissertation will discuss the parallelization of the temporal domain. The goal is to explore further reduction in the time-to-solution. Two objectives are defined to achieve this goal. First, AMR in time is implemented in MGRIT to create an efficient space-time adaptive time-parallelized algorithm. This will require establishing multigrid operators compatible with AMR hierarchies and creating a method to generate adaptive space-time meshes. The second objective is to apply MGRIT with AMR to solve turbulent flows. This will require modifying some standard multigrid operators to effectively work with flows dominated strongly by hyperbolic characteristics. The new design of the operators will need be non-intrusive. Details of the new techniques will be presented following the brief background overview on the MGRIT method.

## 8.1 MGRIT

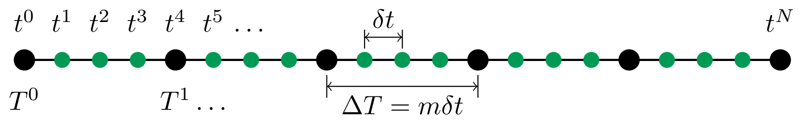


**Figure 8.1:** A two-level time grid composed of F-points in green and C-points in black. The composition of C-points and F-points form the fine time grid while, the C-points form the coarse time grid.

As mentioned in Chapter 1, this research adopts the MGRIT method for parallel-in-time. XBraid [99], a non-intrusive open-source implementation of MGRIT, is used to enable the time parallelization for the present study. MGRIT defines a temporal mesh of time points as

$$t^i = i\delta t\,, \qquad i = 0, \dots, N_t\,, \tag{8.1}$$

with $\delta t = t/N_t$, and the solution state $\mathbf{U}_i \approx \mathbf{U}(t^i)$. While a uniform temporal mesh is presented for simplicity, non-uniform temporal meshes are supported and used in this study. Given a one-step time integration method, $\Phi_i$, such as RK4, the time discretization method can be represented as

$$\mathbf{U}_0 = g_0, \quad \mathbf{U}_i = \Phi_i(\mathbf{U}_{i-1}) + \boldsymbol{g}_i, \quad i = 1, 2, \dots, N_t\,, \tag{8.2}$$

where $\boldsymbol{g}_i$ are solution-independent terms. This can be represented, for simplicity in the linear case, with the system

$$\mathrm{A}\mathbf{U} \equiv \begin{pmatrix} \mathrm{I} & & & \\ -\Phi_1 & \mathrm{I} & & \\ & \ddots & \ddots & \\ & & -\Phi_{N_t} & \mathrm{I} \end{pmatrix} \begin{pmatrix} \mathbf{U}_0 \\ \mathbf{U}_1 \\ \vdots \\ \mathbf{U}_{N_t} \end{pmatrix} = \begin{pmatrix} \boldsymbol{g}_0 \\ \boldsymbol{g}_1 \\ \vdots \\ \boldsymbol{g}_{N_t} \end{pmatrix} \equiv \boldsymbol{g}\,. \tag{8.3}$$

A forward block solve of this system corresponds to sequential time integration. While a forward solve only requires $O(N_t)$ operations, it is not directly parallelizable. Applying a suitable multigrid method such as MGRIT allows a concurrent solve that is also $O(N_t)$ per iteration, but usually with a larger computational constant (in the case of implicit methods, the computational constant can be close to one [100, 101]).

With MGRIT, the temporal mesh specified in Eqn. (8.1) is coarsened into a hierarchy of temporal grids using a coarsening factor $m$. Each grid is partitioned into F-points, which exist only on the current fine time grid, and C-points which exist on both the fine time grid and the next coarser time grid. A two-level hierarchy is shown in Fig. 8.1 where the fine grid is the composite of F-points (in green) and C-points (in black), and the coarse grid is just the C-points. A two-level

MGRIT method is described below for relaxation, prolongation, and restriction operators, but the process may be applied recursively for additional levels.

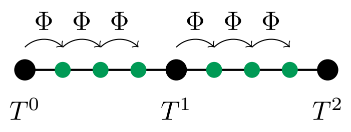


**Figure 8.2:** F-relaxation is the parallelized application of $\Phi$, the time integration operator, to each of the F-points.

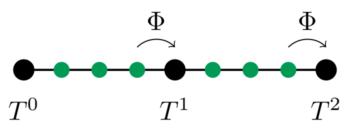


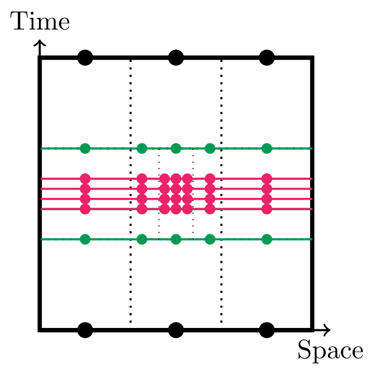
**Figure 8.3:** C-relaxation is the parallelized application of $\Phi$, the time integration operator, to each of the C-points.

Relaxation is performed by application of the time integration operator, $\Phi$, to the time points in two stages. The first stage is F-relaxation which applies the time integration operator to each block of F-points in parallel. F-relaxation is shown in Fig. 8.2 which performs the application of $\Phi$ to the block of three F-points following $T^0$ concurrently with the application of $\Phi$ to the block of three F-points following $T^1$. The second stage is C-relaxation, which propagates the solution to the C-points in parallel. This is shown in Fig. 8.3 with applications of $\Phi$ to points $T^1$ and $T^2$ completed in parallel. Successive applications of F-relaxation and C-relaxation, called FCF-relaxation, make up the relaxation strategy for MGRIT on a given temporal level.

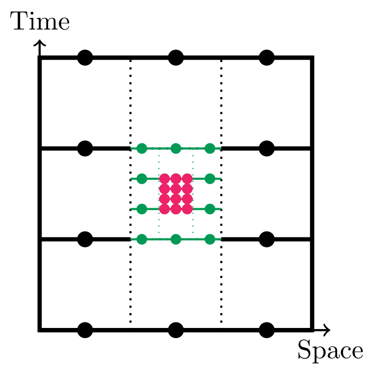
For this study, a full multigrid (FMG) initialization is used. The coarse-grid system of equations, defined at mesh C-points $i = jm$, $j = 0, 1, \ldots, N_t/m$, is

$$\mathrm{A}_\Delta \boldsymbol{e}_\Delta \equiv \begin{pmatrix} \mathrm{I} & & & \\ -\Phi_{\Delta,1} & \mathrm{I} & & \\ & \ddots & \ddots & \\ & & -\Phi_{\Delta,N_t/m} & \mathrm{I} \end{pmatrix} \begin{pmatrix} \mathbf{e}_{\Delta,0} \\ \mathbf{e}_{\Delta,1} \\ \vdots \\ \mathbf{e}_{\Delta,N_t/m} \end{pmatrix} \equiv \boldsymbol{r}_\Delta, \tag{8.4}$$

with the coarse-grid error approximation $\mathbf{e}_{\Delta,i}$, the residual $\boldsymbol{r}_\Delta = \mathbf{R_T}\left(\boldsymbol{g} - \mathrm{A}\left(\mathbf{U}\right)\right)$, and $\mathbf{R_T}$ the temporal restriction operator. Temporal restriction is performed by injection at the C-points and, in this study, is accompanied by spatial coarsening, with $\mathbf{R_S}$ the spatial coarsening operator. Spatial



**(a)** Adaptive mesh refinement without subcycling.



**(b)** Adaptive mesh refinement with subcycling.

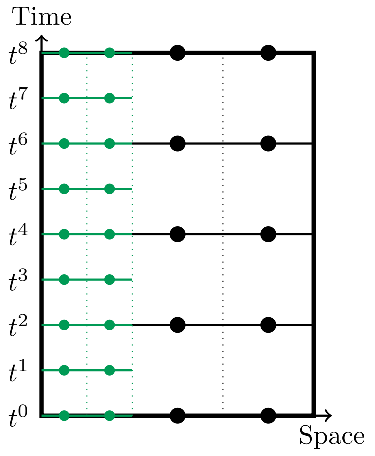
**Figure 8.4:** Subcycling presents a reduction in work for adaptive meshes through less time integration of coarse spatial grids.

coarsening is performed with the exact averaging operator [62]. After restriction, the coarse grid residual equation $\mathrm{A}_{\triangle}\mathbf{e}_{\triangle} = \boldsymbol{r}_{\triangle}$ is solved on the coarse grid sequentially using the coarse grid time propagator $\Phi_{\triangle,j}$.

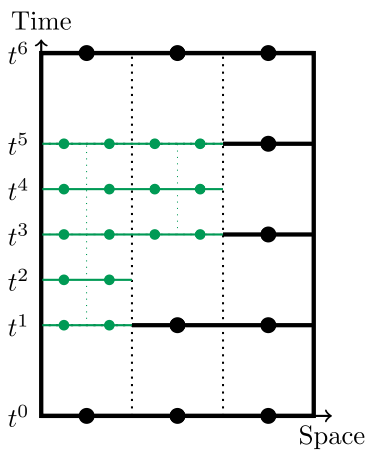
Complementary to restriction, the coarse grid error approximation is first spatially interpolated to the fine spatial resolution, denoted with $\hat{e}_{\triangle} = \mathbf{P_S}(e_{\triangle})$. Spatial interpolation is performed with a fourth-order accurate least-squares method [15]. Next, interpolation in time, $\mathbf{P_T}$, injects the coarse grid error correction to the C-points on the fine grid, and the solution is updated with $\mathbf{U} = \mathbf{U} + \mathbf{P_T}(\hat{e}_{\triangle})$. F-relaxation then updates the F-points. This completes a two-grid MGRIT cycle of relaxation, coarse grid solve for $e_{\triangle}$, followed by the error correction on the fine grid. This can be applied recursively for additional levels. Full approximation scheme (FAS) [102] cycling extends this to nonlinear problems.

## 8.2 Adaptive MGRIT

This research adds the subcycling capability to XBraid. That is the first objective to enable the adaptive space-time parallel algorithm for the purpose of further reduction in work performed on coarser grids. Without subcycling, the coarser spatial resolutions must be integrated to every time point, as seen in Fig. 8.4a. Since the time step sizes are limited to the CFL condition of the fine spatial resolution, the coarse spatial resolution is taking time step sizes much smaller than



**(a)** A static mesh with a subset of the spatial domain refined.



**(b)** An adaptive mesh with a subset of the space-time domain refined.

**Figure 8.5:** A comparison of a static mesh and an adaptive mesh (both with subcycling).

necessary. Introducing subcycling enables the use of the same CFL condition across various spatial resolutions, as shown in Fig. 8.4b, and eliminates integrating the coarse spatial grids at the fine temporal resolutions. Therefore, a further reduction in computational cost is expected. Combining subcycling with MGRIT is one of the novelties of this work.

For clarity, the construction process of the adaptive space-time algorithm will be described in two phases. In the first phase, a static mesh is used to facilitate the explanation of the new methodology. A static space-time mesh is one for which the spatial mesh does not change in time, as illustrated in Fig. 8.5a. In the second phase, an adaptive mesh is introduced and the complication it presents to the methodology is discussed. An adaptive space-time mesh is one for which only subsets of the space-time domain are refined, such as in Fig. 8.5b.

The description of the new algorithm largely follows the procedure of MGRIT [46] and uses the standard multigrid terminologies such as restriction, prolongation, and relaxation [103]. Communication between temporal processors and between spatial processors follows the standard domain decompositions as used by MGRIT [46] and Chombo [62], respectively. However, a few concepts, such as subcycling, vectors, and support structures, are further detailed in this section to assist describing the implementation process of the new algorithm presented in the following two sections.

### 8.2.1 Subcycling

As discussed in Section 5.1, subcycling allows for adaptive refinement in time. The time step size of the fine grid, $\delta t$, is scaled from the time step size of the coarse mesh, $\Delta T$, by the refinement ratio $n = \Delta T/\delta t$. A fixed ratio $n$ for all levels maintains the same CFL condition for all spatial resolutions. Subcycling is heavily leveraged in this PinT algorithm to advance the solution in time on the space-time adaptive meshes.

### 8.2.2 Composite Mesh

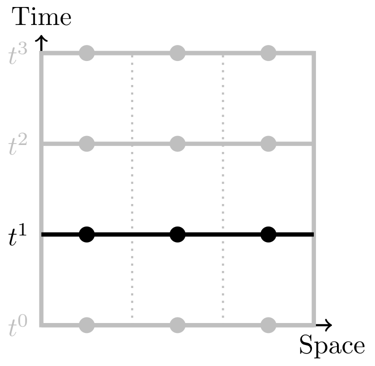
A composite mesh is the hierarchy of spatial grids that excludes coarse points which are overlaid by a finer spatial mesh. Figure 8.5b, for example, shows composite meshes at time points $t^1$, $t^3$, and $t^5$. The overlaid points are excluded from the composite mesh since the finer grid is assumed to be of higher accuracy and replaces the solution in the overlaying regions. Thus, the composite mesh defines a single solution value for each location in the spatial domain.

### 8.2.3 Composite Vector

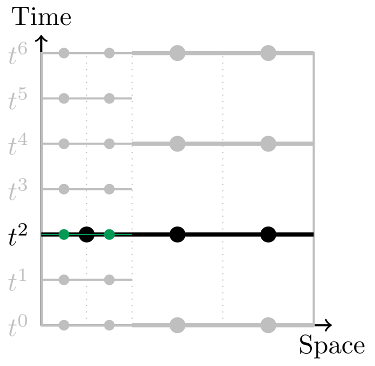
A composite vector consists of the solution values on the composite mesh. Solution values associated with overlaid coarse mesh points are not part of the composite vector. A further requirement of the composite vector is that the solution at all spatial resolutions is at the same solution time. This requirement is met when both the coarse and fine levels are at the same time point during subcycling, such as $T^1$ in Fig. 5.1. With Chord, the composite vector is the solution data that is output for post-processing and analysis.

### 8.2.4 XBraid Vector

An XBraid vector (commonly shortened to "a vector" in this manuscript) is the basic unit of state data that all XBraid operations are performed on. The entire AMR hierarchy at a single time point is considered a vector. While this appears to be resource intensive, it provides the data locality required for subcycling.



**(a)** A vector on the coarse level containing just the base spatial grid. In black is the space-time slice encapsulated by the vector, in gray is the remaining space-time domain not stored by this vector.



**(b)** A vector on the fine level containing both the coarse and fine spatial grids. In black and green is the space-time slice encapsulated by the vector, in gray is the remaining space-time domain not stored by this vector.

**Figure 8.6:** Representation of a vector encapsulating the AMR hierarchy at a single time point.

Therefore, a vector on the coarse multigrid level represented by Fig. 8.6a contains just the base AMR level at one time point. A vector on the fine multigrid level represented by Fig. 8.6b encapsulates a two-level AMR hierarchy containing state information on both the base AMR level and the one level of refinement. This extends in the obvious way to three or more levels. Placing the AMR hierarchy into an XBraid vector is a new concept introduced by this dissertation and is further expanded upon in the following subsections.

### 8.2.5 Support Structure

The support structure includes the backward support structure and forward support structure. In order to perform subcycling, coarse data is required at the beginning and end points of each subcycling interval, that is the coarse time step interval from $T^0$ to $T^1$ of Fig. 5.1. Therefore, support structures are the data structures providing coarse data that is required by subcycling. To facilitate the discussion, the following definitions are used:

- the beginning time point of a subcycling interval (the time point that the coarse level is integrated from) is called the *backward support* structure;

- the ending time point of a subcycling interval (the time point that the coarse level is integrated to) is called the *forward support* structure.

The representation in the XBraid Vector of the support structures is illustrated in Fig. 8.7a. The forward support structure for the coarsest level is at timeline 3, while the intermediate level's forward support structure is at timeline 2. The backward support structure for both coarse and intermediate levels are at timeline 0.

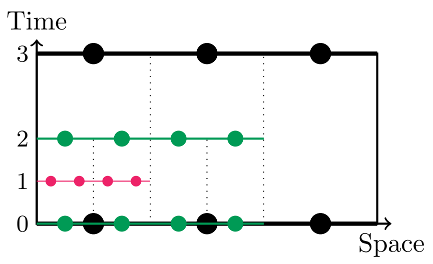
### 8.2.6 Working Time Point

The *working time point* of a vector is the current time point of a vector. The finest spatial level in a time point is always at the working time point, whereas the forward and backward support structures may be forward or backward in time relative to the working time point. This is illustrated in Fig. 8.7a with the working time point in red at timeline 1 and the support structures at the time points forward and backward from the working time point. When operations are applied to a vector, they primarily operate on the working time point. For example, relaxation integrates the working time point forward in time. The current working time point must always be nested between the forward and backward support structure time points, or coincident with one of the support structures.

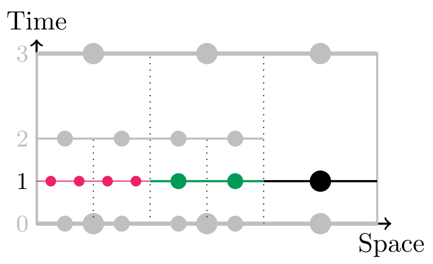
The complete vector data type is represented in Alg. 3. A vector contains a working time point, $\omega$, an array of forward support structures, $\alpha$, an array of backward support structures, $\beta$, $t$ the time of the working time point, and $q$ the number of levels (inclusive of the working time point level).

**Algorithm 3** An XBraid vector containing an AMR hierarchy

```
struct XBraid_vector contains
    integer q                                    ▷ Number of spatial levels
    float t                                      ▷ Current time

    AMRLevel $\omega$                             ▷ Working time point

    AMRLevel[q-1] $\alpha$                        ▷ Array of forward supports
    AMRLevel[q-1] $\beta$                         ▷ Array of backward supports
end
```



**(a)** An XBraid vector representing time point 1. The vector contains three levels of AMR with the working time point (red) and the coarse (black) and intermediate (green) level support structures.



**(b)** The composite XBraid vector of Fig. 8.7a, where coarse (black) and intermediate (green) solutions at timeline 1 are filled by interpolation.

**Figure 8.7:** A representation of how XBraid vectors are stored compared to the composite representation. Note all the space-time points that are not part of the vector have been made gray.

### 8.2.7 Composite XBraid Vector

A composite XBraid vector is an XBraid vector that has been transformed so that the working time point and all supports align in time. This is a new application of the composite vector concept to XBraid vectors, with the distinction that composite XBraid vectors can be constructed at any time point whereas, in AMR, composite vectors are only realized at the end of subcycling intervals. Shown in Fig. 8.7a is when the fine vector is in the middle of a subcycling interval. The forward and backward support structures are shown in black (coarse level) and green (intermediate level), while the working time point is shown in red. The corresponding composite XBraid vector is shown in Fig. 8.7b, where the coarse support structures are interpolated in time to timeline 1. Certain operations, such as computing a vector norm, requires a composite XBraid vector.

## 8.3 Integrate MGRIT with AMR

Implementing AMR with MGRIT is intricate. For clarity, the coupling process is first demonstrated on a static mesh, showing the mesh hierarchy. Then, adaptive mesh refinement is introduced to MGRIT, enabling space-time adaptivity.

### 8.3.1 Demonstration on a Static Mesh

Although static, the mesh refinement possesses the hierarchy of spatial grids from AMR on which subcycling is performed. The primary simplification that occurs with static refinement meshes, versus the adaptive refinement meshes discussed in Section 8.3.2, is that the MGRIT C-points can be made to align with the coarse time interval of subcycling with an appropriate choice of $m$. That is, points $T^0$ and $T^1$ of Fig. 5.1 coincide with C-points.

The coarse temporal grid is $\Omega_T = [0, N_T \Delta T]$, with $N_T$ the number of time points on the coarse temporal grid. This temporal grid is the same as defined in Section 8.1 and corresponds to the C-points in Fig. 8.1. A coarse Cartesian spatial grid is denoted with $\Omega^{(0)}$. This region consists of disjoint patches where an individual patch is

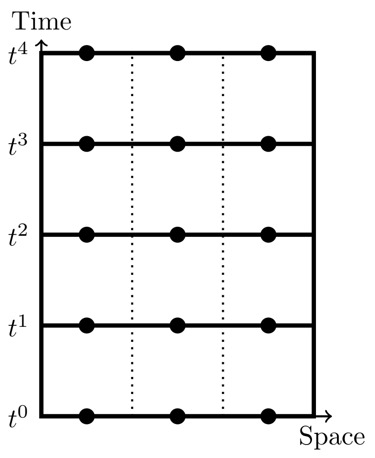
$$\Omega_p^{(0)} = [\boldsymbol{x}_p, \boldsymbol{x}_p + \boldsymbol{k}\Delta x] \ , \tag{8.5}$$

with $\boldsymbol{k} = (k_0, \ldots, k_{\mathrm{D}-1})$ for $\boldsymbol{k} \in \mathbb{Z}^{\mathrm{D}}$, D the spatial dimensionality, and $\Delta x$ the spatial resolution. Therefore $\Omega_p^{(0)}$ expresses a rectangular region of space extending from a lower coordinate $\boldsymbol{x}_p$ to an upper coordinate $\boldsymbol{x}_p + \boldsymbol{k}\Delta x$. The coarse spatial grid is the collection of these patches

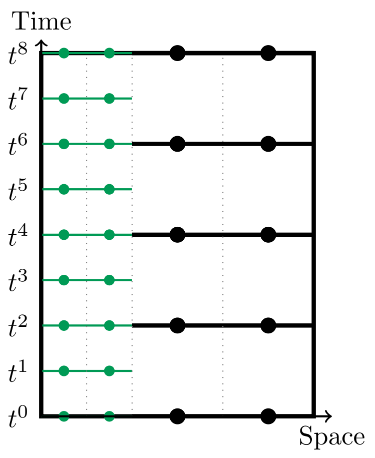
$$\Omega^{(0)} = \bigcup_{p=1}^{N_p^{(0)}} \Omega_p^{(0)} \, , \tag{8.6}$$

with $N_p^{(0)}$ being the number of patches in the coarse spatial grid. The disjoint requirement specifies that each patch satisfies $\Omega_p^{(0)} \cap \Omega_{p'}^{(0)} = \emptyset$ if $p \neq p'$. The collection of time and space points constitutes the coarse space-time mesh, $\Gamma^{(0)} \equiv \Omega_T \times \Omega^{(0)}$. A coarse space-time mesh is shown in Fig. 8.8a.

The fine space-time mesh is a refinement of the temporal domain by a factor of $m$ and a partial refinement of the spatial domain by a factor of $n$. The fine temporal grid is $\Omega_t = [0, N_t \delta t]$, with $N_t = m N_T$. The fine spatial domain is a subset of the coarse spatial domain such that $\Omega^{(1)} \subseteq \Omega^{(0)}$ where $\Delta x^{(1)} = \Delta x^{(0)}/n$, as shown in Fig. 5.2. Similar to the coarse spatial grid, the fine spatial



**(a)** The initial mesh and coarse MGRIT level.



**(b)** The fine mesh and fine MGRIT level.

**Figure 8.8:** The static space-time mesh is constructed by refining a subset of the spatial domain at every time point.

grid is composed from the set of patches on the fine grid

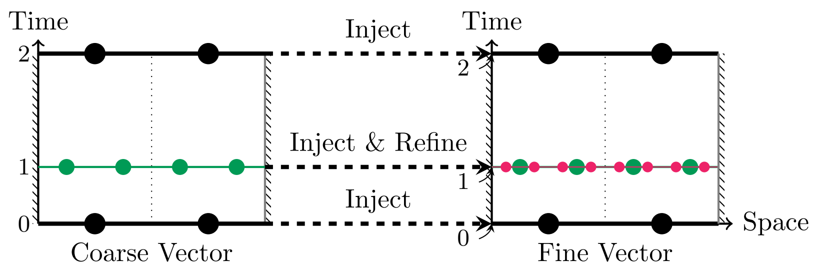
$$\Omega^{(1)} = \bigcup_{p=1}^{N_p^{(1)}} \Omega_p^{(1)} \,, \tag{8.7}$$

with $N_p^{(1)}$ being the number of fine spatial patches. The disjoint requirement applies to the fine level similarly as the coarse level. As shown in Fig. 8.8b the fine space-time mesh contains fine spatial points as well as coarse spatial points that are not overlaid by the fine spatial points. With a spatial coarsening operator Cr, the fine space-time mesh is

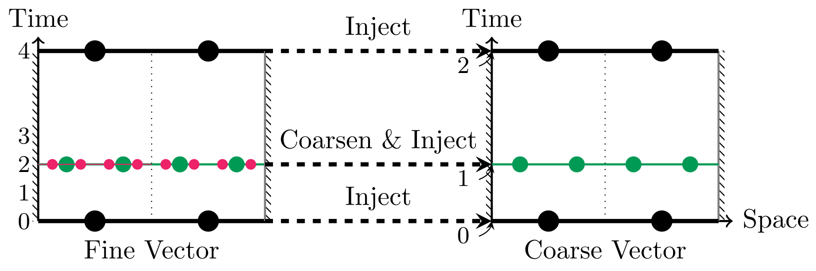
$$\Gamma^{(1)} \equiv \Omega_t \times \Omega^{(1)} \cup \Omega_T \times \left(\Omega^{(0)} - \mathrm{Cr}\left(\Omega^{(1)}\right)\right) \,. \tag{8.8}$$

This refinement is illustrated in Fig. 8.8b for a temporal and spatial refinement factor of 2 in the left part of the domain. Note this is applicable to any refinement factor.

Between levels, data must be transferred through spatial prolongation and restriction. Prolongation and restriction are depicted in Fig. 8.9 between the second and third levels in a 3-level MGRIT method. Prolongation, illustrated in Fig. 8.9a, uses injection to transfer the current fine level error approximation at timeline 1 (in green) and coarser support structures at timelines 0 and



**(a)** Prolongation: Spatial refinement and injection to the finer multigrid level.



**(b)** Restriction: Spatial coarsening and injection to the coarser multigrid level.

**Figure 8.9:** Injection to coarser or finer levels also injects the forward and backward support levels at their current time value.

2 (in black) onto the fine vector. The new fine spatial grid (in red) is created and the solution filled by spatial interpolation ($\mathbf{P_S}$) using the fourth-order least-squares method [15].

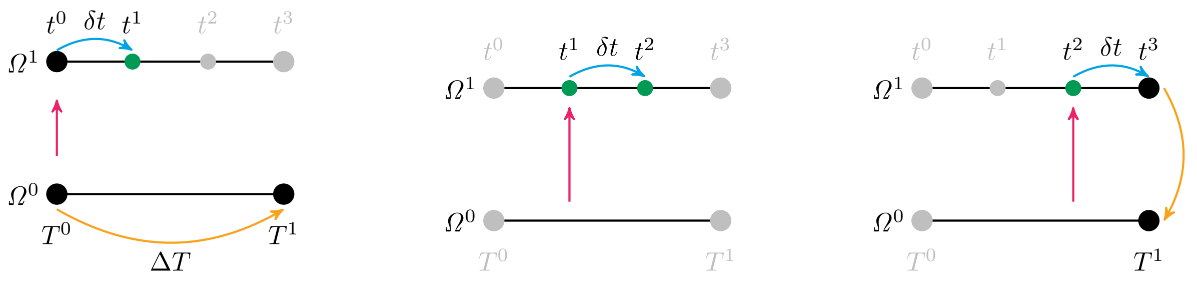
Restriction performs the complementary action of injecting and coarsening values to the coarse grid. Restriction is shown in Fig. 8.9b where all timelines in the fine vector are injected to the coarse vector. The solution is averaged ($\mathbf{R_S}$) from the fine spatial grid onto the coarse spatial grid. The prolongation and restriction process is represented in Alg. 4.

Next, the application of $\Phi$ for use in relaxation is defined. On the coarsest level, there is only the base coarse grid in the AMR hierarchy and so time integration may be applied directly. Application of the time integration operator, $\Phi$, is demonstrated on the fine level in Fig. 8.10. Since the C-points align with the coarse time points in subcycling, this leads to two different time integration scenarios. The first is when time integration begins at a C-point, the second when time integration begins at an F-point. Time integration is outlined with pseudocode in Alg. 5.

In the first scenario, time integration begins at a C-point and the support structures align with the working time point. This scenario corresponds to the first time point of F-relaxation and is

**Algorithm 4** Procedure for prolongation and restriction

```
                                              ▷ $v_\mathrm{c}$: Coarse vector
                                              ▷ $v_\mathrm{f}$: Fine vector

function PROLONGATION($v_\mathrm{c}$, $v_\mathrm{f}$)
    $q$, $\omega$, $\alpha$, and $\beta$ from $v_\mathrm{f}$
    $q_\Delta$, $\omega_\Delta$, $\alpha_\Delta$, and $\beta_\Delta$ from $v_\mathrm{c}$
    for $i = 0, \dots, q_\Delta$ do
        Inject $\alpha_i \leftarrow \alpha_{\Delta,i}$
        Inject $\beta_i \leftarrow \beta_{\Delta,i}$
    end for
    Interpolate working time point $\omega \leftarrow \mathbf{P_S}(\omega_\Delta)$
end function

procedure RESTRICTION($v_\mathrm{c}$, $v_\mathrm{f}$)
    $q$, $\omega$, $\alpha$, and $\beta$ from $v_\mathrm{f}$
    $q_\Delta$, $\omega_\Delta$, $\alpha_\Delta$, and $\beta_\Delta$ from $v_\mathrm{c}$
    for $i = 0, \dots, q_\Delta$ do
        Inject $\alpha_{\Delta,i} \leftarrow \alpha_i$
        Inject $\beta_{\Delta,i} \leftarrow \beta_i$
    end for
    Average working time point $\omega_\Delta \leftarrow \mathbf{R_S}(\omega)$
end procedure
```



**(a)** Time integration that begins at a C-point.

**(b)** Subsequent step of fine-level integration of subcycling.

**(c)** Completion of subcycling interval.

**Figure 8.10:** Application of $\Phi$, demonstrating the two time integration scenarios on the fine multigrid level.

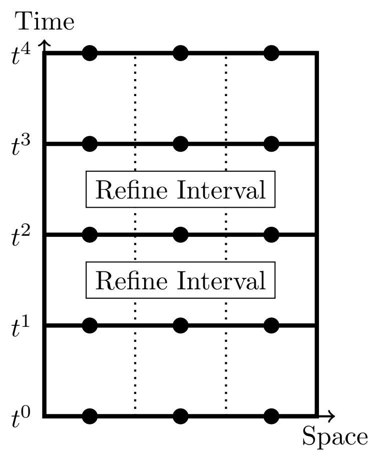
illustrated in Fig. 8.10a. Subcycling must perform the coarse time step first (in orange), followed by interpolation to invalid ghost cells at $t^0$ (in red), and lastly perform a fine time step on the fine grid (in blue).

**Algorithm 5** Procedure for time integration with two AMR levels

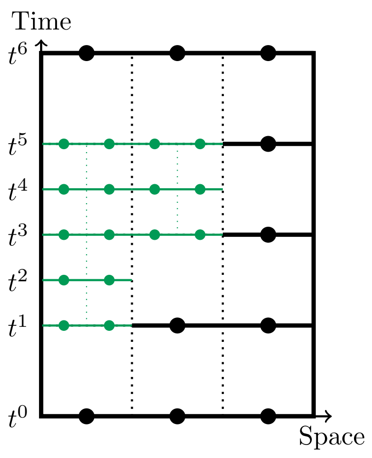
```
                                                        ▷ $v$: Vector to integrate
                                                        ▷ $t^i$: Starting time point
                                                        ▷ $t^{i+1}$: Ending time point

function RELAXATION($v$,$t^i$,$t^{i+1}$)
    $\omega$, $\alpha$, and $\beta$ from $v$
    if $t^i$ is a C-point then
        $\beta \leftarrow \alpha$                              ▷ $\alpha$ starts at same time as $\beta$
        $\alpha \leftarrow \Phi_{\Delta,i}(\beta)$
    end if
    $\omega = \mathrm{I}_\Delta(\beta, \alpha)$                ▷ Interpolate to invalid ghost cells
    $\omega = \Phi_i(\omega)$
    if $t^{i+1}$ is a C-point then
        $\alpha = \mathbf{R_S}(\alpha)$                        ▷ Average solution
    end if
end function
```

In the second scenario, integration begins at an F-point. This scenario corresponds to F-points subsequent to the first scenario as well as C-relaxation. The forward and backward support structures have been previously filled by the application of the time integration operator at the initial F-point. Since the support structures are already filled, all that needs to occur is the interpolation to invalid ghost cells (in red) followed by time integration on the fine grid (in blue), as shown in Fig. 8.10b.

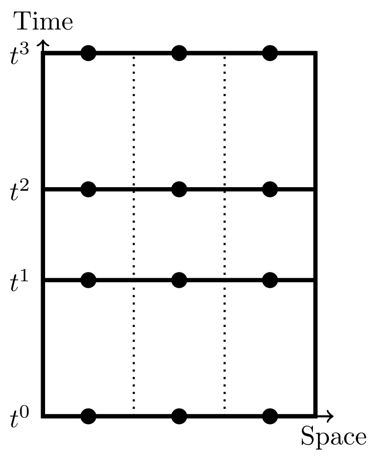
In the case of C-relaxation, time integration of the fine grid reaches the end time point of a subcycling interval, $t^3$ in Fig. 8.10. After the time integration, the solution is averaged to the coarse spatial grid to maintain coupling between the coarse and fine spatial grids. The flux is subsequently corrected to ensure single-valued flux along the coarse-fine interface. C-relaxation is illustrated in Fig. 8.10c with the interpolation in red, time integration in blue, and the averaging and flux correction in orange.



**(a)** The initial mesh points.



**(b)** The mesh after one refinement.



**(c)** The mesh used in MGRIT after time-coarsening.

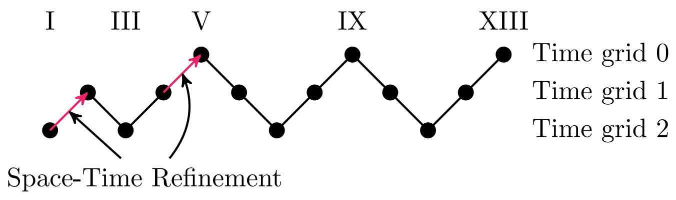
**Figure 8.11:** The refined space-time mesh in Fig. 8.11b is constructed by tagging the initial space-time mesh in Fig. 8.11a. Subcycling intervals on the refined mesh are established from the initial coarse mesh's time points. MGRIT's temporal coarsening creates coarser versions of the space-time mesh in Fig. 8.11c.

### 8.3.2 Coupling AMR with MGRIT

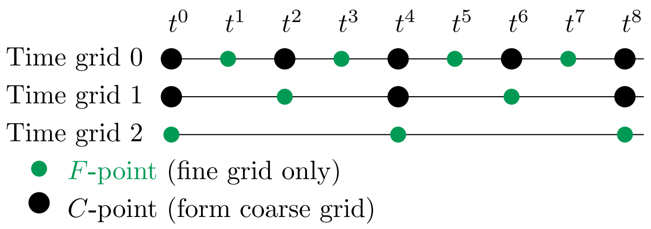
Subcycling is now added to the static refinement algorithm to enable space-time adaptivity. The main complication addressed is the fact that uniform coarsening in XBraid can lead to C-points that are not aligned with natural subcycling interval boundaries. For example, coarsening the grid in Fig. 8.11b by a factor of 2 leads to Fig. 8.11c. Maintaining conservation of mass, momentum, and energy with this grid configuration is problematic, and one solution is detailed below. The basic idea is to augment the standard space-time AMR grid with additional subcycling synchronization time points (support structure points) that line up with C-points. This creates a new space-time mesh over which subcycling can be performed and conservation ensured as usual. On this new space-time mesh, however, the components of the MGRIT algorithm are just as in the static case, with only a few extra precautions needed to maintain stability.

This section is organized as follows. First, the adaptive space-time mesh construction process is discussed, followed by a description of how points are added to align C-points with subcycling intervals. Lastly, the adaptations to time integration are discussed.

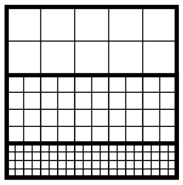
To begin, an adaptive space-time mesh is constructed by generating an initial coarse uniform spatial mesh, as illustrated in Fig. 8.11a, which corresponds to point “I” in the multigrid cycle



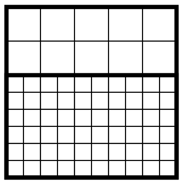
**(a)** Finer multigrid time grids are created by refinement in an FMG cycle.


$t^0$ $t^1$ $t^2$ $t^3$ $t^4$ $t^5$ $t^6$ $t^7$ $t^8$
Time grid 0
Time grid 1
Time grid 2
$F$-point (fine grid only)
$C$-point (form coarse grid)

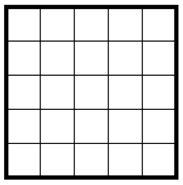
**(b)** After each temporal refinement, the coarser time grids are created by coarsening from the finest time grid with a fixed coarsening factor.



**(c)** Spatial mesh at time grid 0 and $t^4$.



**(d)** Spatial mesh at time grid 1 and $t^4$.



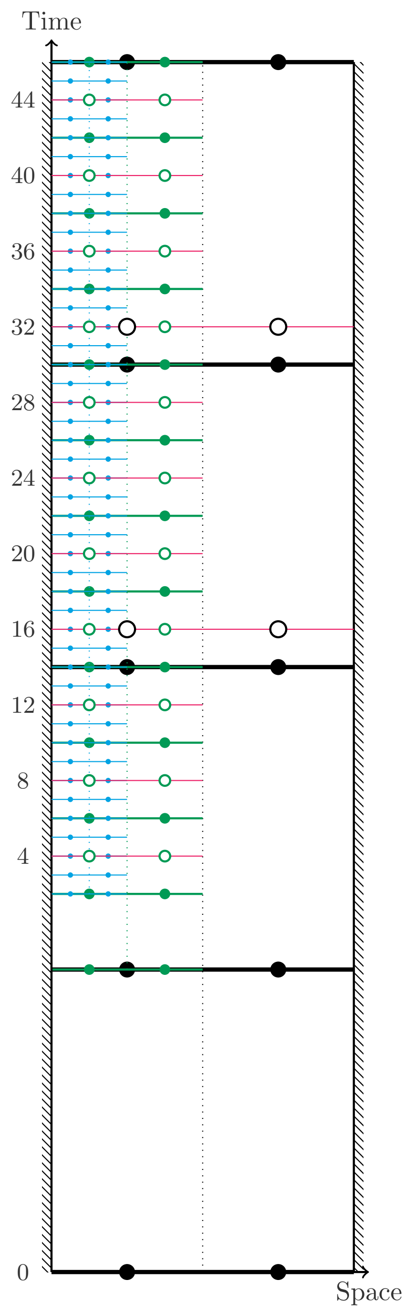
**(e)** Spatial mesh at time grid 2 and $t^4$.

**Figure 8.12:** More refined time grids are created in an FMG cycle. Each time point has a corresponding hierarchy of AMR meshes. An example spatial mesh is shown for time point $t^4$ at each level in the time grid.

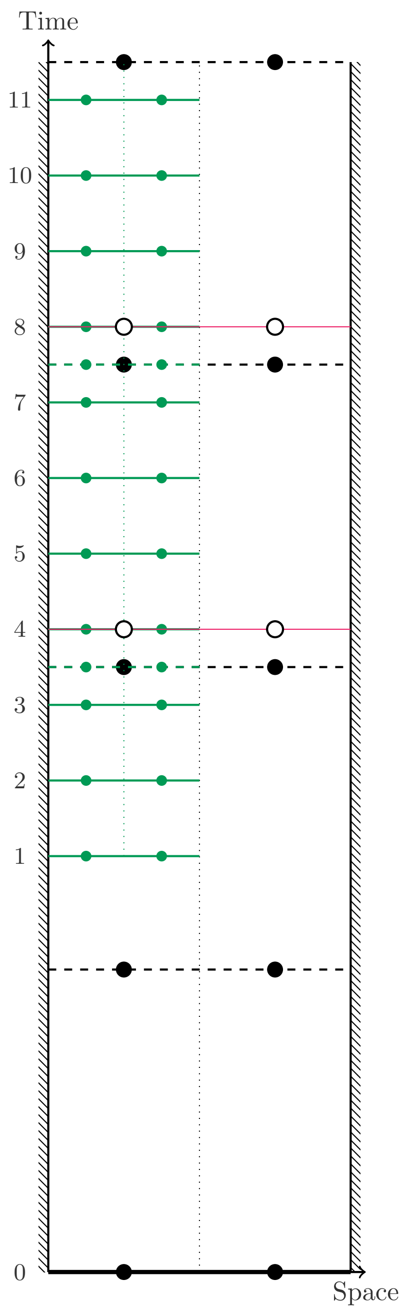
shown in Fig. 8.12a. Subsequently, on this coarse mesh, a preliminary solution is computed to provide guidance for refinement based on solution gradients. For example, two regions in Fig. 8.11a are marked for refinement based on velocity normal gradients. A new finer space-time mesh is created as shown in Fig. 8.11b. Coarse intervals for subcycling are set from the initial coarse space-time mesh, Fig. 8.11a, and the number of fine intervals is established by the temporal refinement factor, $m$. MGRIT coarsening of the mesh is carried out by removing every other time line from the temporal grid, as shown in Fig. 8.11c with $m = 2$ as an example. The coarse spatial grid points at timelines $t^1$ and $t^2$ are found from the coarse spatial grids of the nearest timeline backward on the fine mesh, which in this case would be those at $t^1$ and $t^3$ from Fig. 8.11b. The residual values are transferred to the coarse grid with restriction as described in the previous section. This process leads to a two-level cycle for MGRIT, and may be repeated iteratively to create more levels until the desired space-time resolution is met. To mention, a new finer time mesh defines a new multigrid level, such as point "V" in Fig. 8.12a. An adaptive space-time mesh is illustrated by Fig. 8.12b and Figs. 8.12c–8.12e together.

Flux correction must be made to ensure that there is only one flux value for each solution variable on each face of a cell. At the end of a subcycling interval, a single-valued flux must be maintained at the coarse-fine interface. However, in the adaptive MGRIT algorithm, coarse grid corrections are performed at C-points which do not necessarily time-synchronize with the end of a subcycling interval. This leads to a double-valued flux on the faces of the coarse-fine interface.

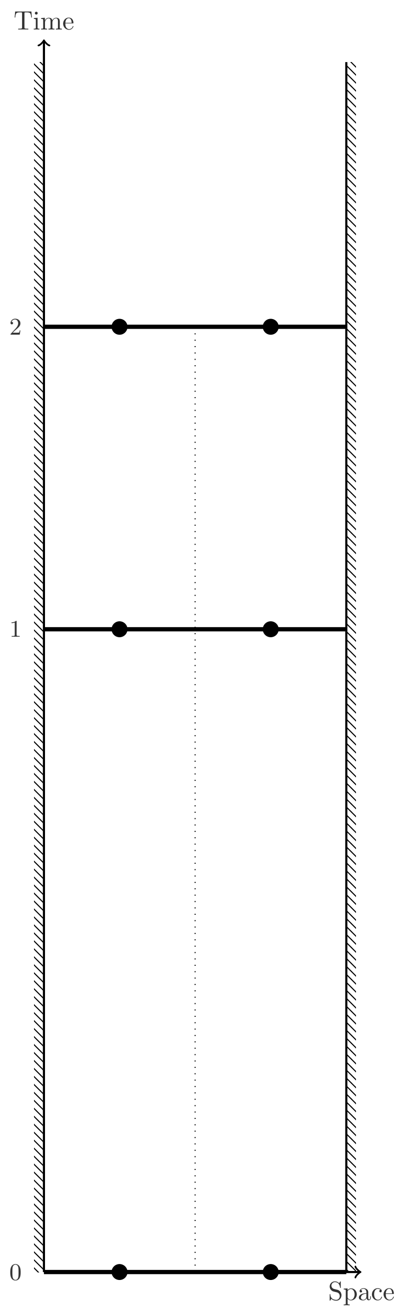
To resolve this discrepancy, additional points are added to the space-time mesh to force the synchronization of subcycling intervals and MGRIT C-points. This is illustrated in Fig. 8.13 where the new points are shown with open circles. With a coarsening factor of $m = 4$, the C-points occur every fourth time point as shown by the red lines. At $t = 4$ in Fig. 8.13a, the AMR timeline is extended to the next coarser AMR level as shown by the red line and open circle. In doing so, subcycling will halt at $t = 4$ for time-synchronizing with the C-point. At $t = 16$, the AMR timeline is extended to all coarser AMR levels because $t = 16$ is a C-point on both the finest mesh and the next coarser mesh, as shown in Fig. 8.13b. For MGRIT coarsening factors larger than 2,



**(a)** Finest mesh



**(b)** 1st Coarsening



**(c)** 2nd Coarsening

**Figure 8.13:** A space-time mesh with extended C-points shown in red, with the open circles indicating the extra space-time points added for synchronization. Dashed horizontal lines and their corresponding points indicate the support timelines that are not part of the MGRIT temporal domain in Fig. 8.13b.

this effectively halves the number of fine level time integrations per subcycling interval. Since the coarser level time integrations are relatively cheap and the time steps are parallelized, this should have minimal effect on the wall-clock time. With larger temporal refinement factors, the effect is negligible.

Prolongation and restriction occurs at MGRIT C-points, and subcycling intervals coincide with C-points due to the new space-time points added. This allows the use of the same prolongation and restriction processes discussed in Section 8.3.1.

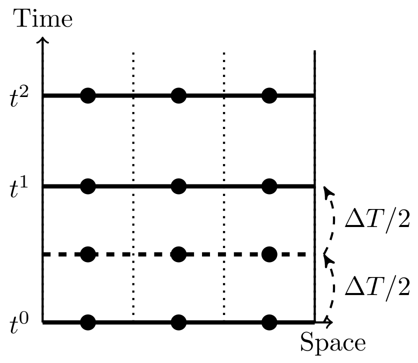


**Figure 8.14:** When MGRIT intervals exceed the maximum stable time step size, smaller time step sizes, here $\Delta T/2$, are taken by the CFD application code to maintain stability.

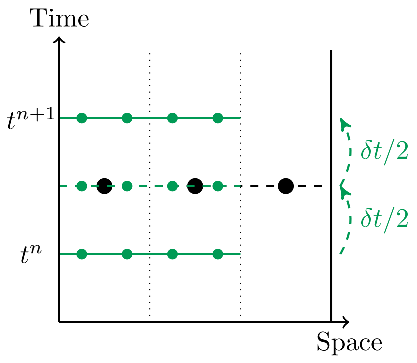


**Figure 8.15:** A subcycling interval (in dashed black) that has become misaligned with MGRIT time points. The fine level must stop at the subcycling interval as indicated with the dashed green line.

The interval between time point $t^0$ and $t^1$ has grown larger as seen in Fig. 8.13c. This larger interval may exceed the maximum stable time step for the explicit time marching method. In order to maintain a stable time step size, the CFD application code will instead take multiple smaller steps as shown in Fig. 8.14. These smaller time steps have no effect on the MGRIT algorithm.

When the subcycling intervals become misaligned with the C-points, they may also be misaligned with any MGRIT time point as shown in Fig. 8.15. Although the time step size from $t^n$ to $t^{n+1}$ may be within the stability limit of the time integration method, two separate steps are taken. The first step integrates the fine level to the end point of the subcycling interval. The second step begins the next subcycling interval by integrating the coarse grid before integrating the fine grid to time $t^{n+1}$.

# Chapter 9

# Verification and Validation of MGRIT with AMR

The coupled algorithm is verified and validated on two diffusion-driven flows. Diffusion-driven flows are tested here because the elliptic nature of the flows are known to converge well with MGRIT, allowing the coupled algorithm to be analyzed without introducing complexities needed for hyperbolic systems of equations. The verification and validation ensures that the coupled algorithm (i) preserves the fourth-order and conservation properties of Chord, (ii) obtains the correct solution profile, and (iii) achieves MGRIT convergence and a performance enhancement. The test problems are first introduced in Section 9.1 and the results are presented in Section 9.2.

## 9.1 Summary of Test Problems

The MGRIT+AMR algorithm is demonstrated with two test problems: a transient Couette flow and a periodic-in-time Stokes second problem. These problems are solved on cartesian grids with no source term, and the density and pressure fields are constant.

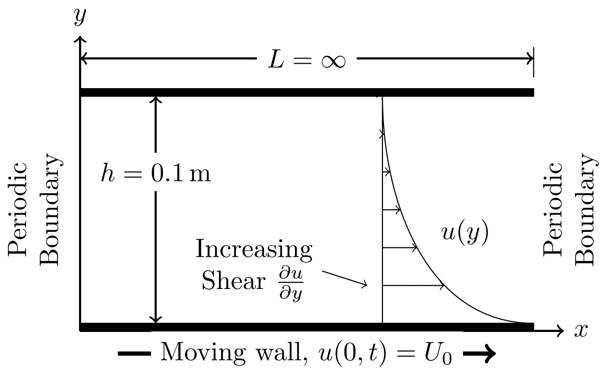


**Figure 9.1:** Diagram of the transient Couette flow case showing boundary conditions and velocity profile at some point before steady state.

### 9.1.1 Transient Couette Flow

An unsteady Couette flow is chosen because it is entirely driven by shear stress, where the underlying system of PDEs is parabolic. This makes Couette flow an excellent test for the convergence of the new adaptive MGRIT framework because MGRIT is known to perform well on parabolic problems. Shown in Fig. 9.1 is a diagram of Couette flow. The working fluid is bounded by an infinite, stationary wall on the top and a moving wall at the bottom of the domain while the left and right side are periodic. The initial flow field is quiescent and, as the solution time progresses, an axial velocity field develops until a steady state is achieved. At the steady state, the axial velocity decreases linearly from the moving wall to the stationary wall.

The moving wall speed $U_0$ is determined from a specified Reynolds number, $\mathrm{Re} = 200$, so that $U_0 = \mathrm{Re}\,\nu/h$ with the kinematic viscosity $\nu = 1.46 \times 10^{-5}\,\mathrm{m^2\,s^{-1}}$ and the distance between the walls $h = 0.1\,\mathrm{m}$. A base space-time mesh with spacing $\Delta x$ and uniform temporal spacing $\Delta t$ is specified. The initial coarse $\Delta t$ must be chosen so that the von Neumann [16] condition is satisfied. Refinements in both space and time are then performed. Refinement tagging for this problem is based on vorticity, which in a pseudo one-dimensional flow is equivalent to $\omega = \partial u/\partial y$ [104]. A typical space-time mesh for this Couette flow is as shown in Fig. 8.13 where the mesh builds up near the wall as the fluid accelerates. Not shown in Fig. 8.13, due to size constraints, is the space-time mesh as the solution time approaches steady state and the mesh recedes.

Transient Couette flow has an analytic solution that describes the velocity profile as a function of time [105]. Given the moving wall velocity of $U_0$ and domain height $h$, the velocity profile is

$$\begin{aligned} u(y,t) = {} & U_0\left(1 - \frac{y}{h}\right) \\ & - \frac{2U_0}{\pi}\sum_{n=1}^{\infty}\frac{1}{n}\exp\left(-n^2\pi^2\frac{\nu t}{h^2}\right)\sin\left(\frac{n\pi y}{h}\right) , \end{aligned} \tag{9.1}$$

where $\nu$ is the kinematic viscosity. Numerical error is measured using this analytic solution by $L_1$, $L_2$, and $L_\infty$ norms.

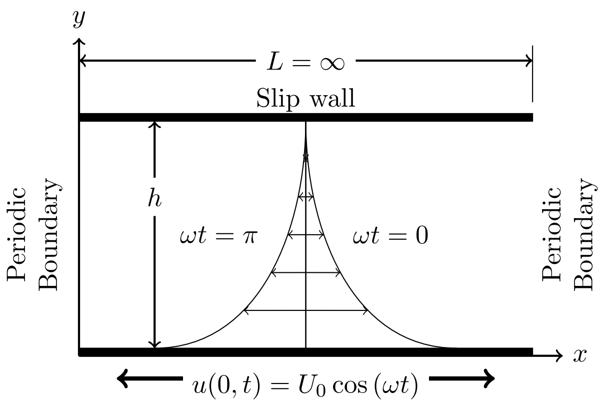


**Figure 9.2:** A diagram of Stokes second problem.

### 9.1.2 Periodic in Time Problem

There are many time-periodic problems of interest in engineering applications, such as turbomachinery, piston engines, and wind turbines. Such problems may see significant benefits from being solved by parallel-in-time algorithms. For instance, the MGRIT algorithm may drive out the transient components over only a single periodic time period [106].

Stokes second problem is periodic in time. As shown in Fig. 9.2, an infinite no-slip plate at $y = 0$ sets a fluid at rest into motion with a harmonic oscillation proscribed by

$$u(0,t) = U_0 \cos(\omega t) \ , \tag{9.2}$$

with amplitude $U_0 = \mathrm{Re}\,\nu/h$, Reynolds number $\mathrm{Re} = 20000$, kinematic viscosity $\nu = 1.46 \times 10^{-5}$ $\mathrm{m^2\,s^{-1}}$, and frequency $\omega = 2\pi/7$. The analytic solution of the velocity distribution is [105]

$$u(y,t) = U_0 \exp(-\eta y) \cos(\omega t - \eta y) \ , \tag{9.3}$$

with $\eta = \sqrt{\omega/(2\nu)}$. All cases are run on a solution time interval of $t_0 = 0\,\mathrm{s}$ to $t_f = 7\,\mathrm{s}$ in order to complete one period.

## 9.2 Results

These test problems have analytic solutions; therefore they serve both verification and validation of the final algorithm. Verification assessment determines if the conceptual model is correctly implemented in the new space-time parallel algorithm. Validation assessment ensures that the computational results agree with physical reality. To perform this, (i) the order of accuracy of the adaptive parallel-in-time algorithm is evaluated, (ii) the conservation properties of the underlying finite-volume method is measured, (iii) the convergence property of the MGRIT algorithm is assessed, and (iv) the performance of the time-parallel algorithm is compared to the sequential algorithm. In all time-parallel cases, an absolute convergence tolerance conditioned upon the discretization error matching the time-sequential case is used.

### 9.2.1 Transient Couette Flow

To confirm the adaptive MGRIT algorithm converges to the same solution as that from the sequential time stepping, a grid convergence study is performed. The base grid is progressively refined from $16^2$ cells to $32^2$ cells and lastly $64^2$ cells using uniformly refined MGRIT levels. This provides mesh spacings of $\Delta x = 6.25 \times 10^{-3}\,\mathrm{m}$, $3.125 \times 10^{-3}\,\mathrm{m}$, and $1.5625 \times 10^{-3}\,\mathrm{m}$ at the three refinements. The problem is solved from $t_0 = 0\,\mathrm{s}$ to $t_f = 50\,\mathrm{s}$ of solution time with $\Delta t = 0.2\,\mathrm{s}$, $0.05\,\mathrm{s}$, and $0.0125\,\mathrm{s}$. At each refinement the error from the analytic solution is measured and the convergence rate is calculated. The code preserves fourth-order accuracy as measured by the $L_1$, $L_2$, and $L_\infty$ norms from the analytic solution. This is the same convergence rate that has previously been verified for the time-sequential algorithm [16, 29–33]. Additionally, the error at each resolution is comparable to the sequential algorithm. Changing the number of temporal and spatial processors has no effect on the grid convergence rate.

For a CFD application code based on finite volume methods, the conservation of mass, momentum, and energy must also be maintained. The conserved quantities is compared in a case with a base grid of $16^2$ cells and 250 coarse time points with 4 levels of refinement. Each level is refined in time with a factor of $m = 4$ and refined in space with a factor of $n = 2$. An adaptive refinement

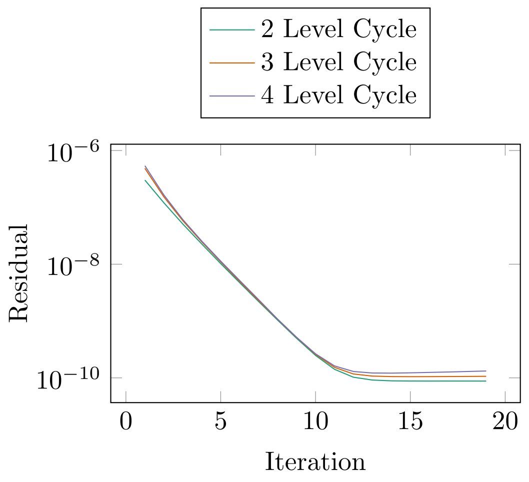


**Figure 9.3:** The convergence rate of the temporal residual over MGRIT iterations, showing that the convergence rate is independent of the number of MGRIT levels.

technique is used where high gradient regions near the wall are refined. Over time, as the gradient smooths out, the mesh is coarsened. The solution time runs from $t = 0\,\mathrm{s}$ to $t = 50\,\mathrm{s}$. The total $x$-momentum is measured at each time step in the domain for both the time-sequential case and the time-parallel case. The conservative quantities for the two cases track each other with machine-precision zero difference. Another method to verify the conservative nature of the algorithm is to initialize the Couette flow at the steady state condition and run a large number of time steps. Upon doing so, the conservative quantities show machine-precision zero difference over time steps which further confirms the conservative property. Varying the number of cores used in space and time has no effect on maintaining the conservation of the system.

To test the convergence properties of the adaptive MGRIT algorithm, the convergence rates of the residual and solution error over iterations are measured. Three different cases are run with base spatial resolution of $32^2$ ($\Delta x = 3.125 \times 10^{-3}\,\mathrm{m}$) and 250 coarse time points with 3 levels of adaptive refinement. These cases run from $t = 0\,\mathrm{s}$ to $t = 12.5\,\mathrm{s}$ with a coarse level $\Delta t = 0.05\,\mathrm{s}$. An adaptive temporal refinement factor of $m = 4$ is used with an adaptive spatial refinement factor of $m = 2$. Once the last refinement is performed, the test cases are varied by the number of temporal coarsenings used by MGRIT. Using this procedure allows the same fine space-time mesh with a different number of temporal levels, that is the MGRIT algorithm performs V-cycles with either 2,

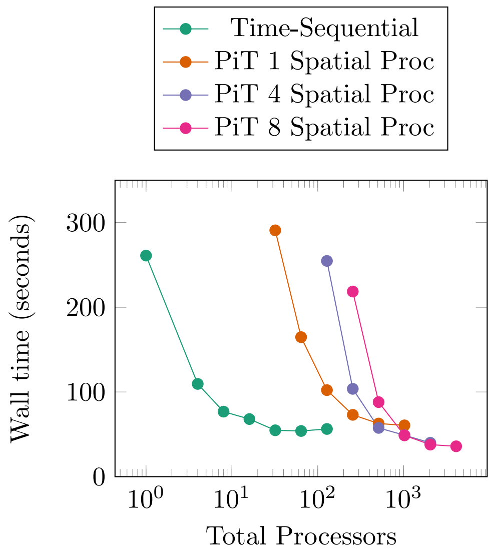


**Figure 9.4:** Strong scaling comparison of time-sequential and parallel-in-time codes. The time-parallel code is tested with two different configurations: one processor in space, and eight processors in space.

3, or 4 temporal levels. Figure 9.3 shows the MGRIT residual history decreasing over a number of iterations. This shows that the number of temporal levels has little impact on convergence rate of the combined AMR-MGRIT algorithm. As expected, the number of cores used in space and time has no effect on the convergence rate.

Performance is compared in terms of the strong scaling in sequential and time-parallel modes. For all strong scaling tests, a base spatial mesh of $16^2$ with four additional levels of AMR refinement are used. A total of approximately $8600$ time points are used on the finest temporal grid and the solution is solved from the quiescent state to 7 seconds of solution time. The coarse level spatial resolution is $\Delta x = 6.25 \times 10^{-3}\,\mathrm{m}$ and the coarse time step size is $\Delta t = 0.2\,\mathrm{s}$. The temporal refinement ratio is $m = 4$ and the spatial refinement ratio is $n = 2$.

A set of baseline cases is run in the unmodified time-sequential algorithm with increasing number of processors parallelizing the spatial dimension. The baseline cases are shown in green in Fig. 9.4 with scaling from 1 processor to 128 processors of spatial parallelization. For this particular case, diminishing returns are seen at 8 processors and communication overhead prevents any further speedups beyond 32 processors.

The time-parallel algorithm is run with just a single processor in space but the number of processors parallelizing the time domain vary from 32 up to 1024 processors. A five-level MGRIT method is used for the parallelization with a coarsening factor of $m = 4$ leading to approximately 32 coarsest level time points. At 32 processors in time, the wall time is approximately equal to the sequential case with 1 processor. Needing 32 processors in time to match the sequential wall-clock time is similar to the number needed for other MGRIT studies [46, 107]. As can be seen, the time parallelization is saturated by 256 processors in time and the wall-clock time is about the same as the spatially saturated time-sequential case. The fastest time parallelization for a single processor in space is at 1024 processors in time; this case achieves a $4.3\times$ speedup over the time-sequential single processor case (left-most green point).

De Sterck [40] suggests that adding spatial parallelization after saturating the temporal parallelization may result in further speedups. To test that, time-parallel cases were run with four and eight spatial processors as shown in Fig. 9.4. The number of temporal processors is scaled from 32 processors in time to 512 processors in time for a maximum of 4096 processors. At the maximum number of processors, a $1.5\times$ speedup is achieved over the fastest time-sequential case (at 64 processors in space).

Furthermore, the overhead introduced by the adaptive MGRIT algorithm is compared to the sequential application of the CFD code. This overhead can be measured by restricting the adaptive MGRIT to the finest time grid for an equivalent sequential time integration. It is found that the overhead is problem dependent on parameters such as the number of AMR and multigrid levels used, the degrees of freedom, etc., however nearly all of the overhead occurs in the FMG process. More quantitative measurements will be performed in the future. Nevertheless, this overhead is not a concern for the performance of time-parallel computations because the FMG process provides a good initial guess for the remaining multigrid cycles.

Another area of research is finding ways to increase the size of the time domain to be parallelized. The Couette flow case has a small final solution time compared to the solution time needed to reach steady state. This is due in part to the fact that optimal performance of the MGRIT algorithm is achieved when there are a small number of time points on the coarsest time grid. Due to the explicit time marching method, the time step size on the coarsest time grid is limited and further spatial coarsening is not possible. Finding methods to take larger time steps on the coarsest time grid, perhaps by using an implicit method on only those coarse time grids needing it, would allow for larger time domains where parallel-in-time methods would excel.

### 9.2.2 Stokes Second Problem

As with the transient Couette flow case, this time periodic case is verified for fourth-order error convergence with respect to the analytic solution using a grid convergence study on $16^2$ cells up to $512^2$ cells. The spatial resolutions vary from $\Delta x = 6.25 \times 10^{-3}\,\mathrm{m}$ to $\Delta x = 1.953\,125 \times 10^{-4}\,\mathrm{m}$ and the time step sizes decrease from $\Delta t = 0.2\,\mathrm{s}$ to $\Delta t = 1.953\,125 \times 10^{-4}\,\mathrm{s}$.

The conservation of mass, momentum, and energy is tested on a 4-level AMR grid with a base mesh resolution of $\Delta x = 6.25 \times 10^{-3}\,\mathrm{m}$ and a time step size of $\Delta t = 0.2\,\mathrm{s}$ along with a spatial refinement of $n = 2$ and a temporal refinement of $m = 4$. The values of conserved quantities from the parallel-in-time case are within machine-precision zero difference from the time-sequential case.

The numerical solutions at four characteristic times are compared for the time-parallel and time-sequential algorithms against the analytic solution. Figure 9.5 shows the time-parallel solution after convergence of MGRIT compared to the time-sequential solution after convergence. Convergence requires four MGRIT iterations for the time-parallel case and ten time intervals for the time-sequential case. These velocity profiles are taken through the center line ($x = 0$) of the domain. At times $t = \pi/2\,\mathrm{s}$ and $t = 3\pi/2\,\mathrm{s}$ the wall speed is zero while the momentum of the fluid continues to push the fluid in the positive $x$-direction and negative $x$-direction, respectively. The error is at its maximum in the high gradient region. Conversely, at times $t = \pi\,\mathrm{s}$ and $t = 2\pi\,\mathrm{s}$ the wall is at its maximum positive $x$-direction and negative $x$-direction speed, respectively. Again the error is mostly concentrated in the high gradient region near the wall. Nevertheless, the parallel-in-time solution matches closely to the time-sequential solution; both agree well with the analytic

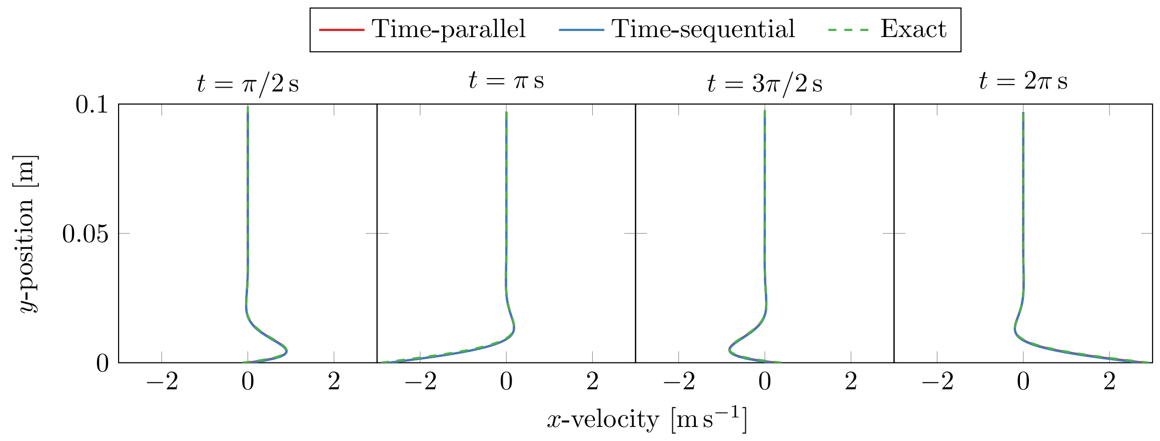


**Figure 9.5:** Velocity profiles at the characteristic times after convergence.

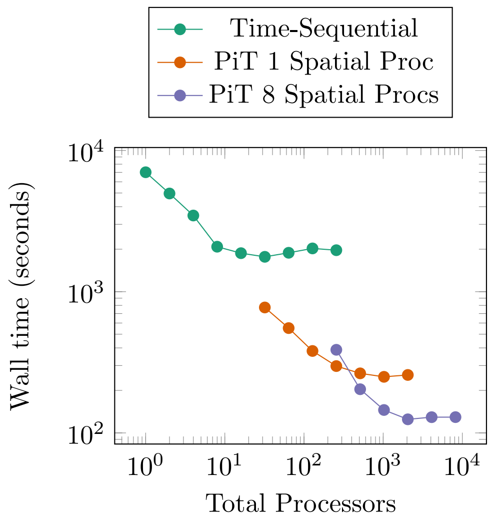


**Figure 9.6:** Strong scaling study comparing time-sequential with time-parallel.

profiles. The error is the largest in the high gradient regions near the wall, but is significantly lower than in the first iteration or time interval.

A strong scaling test is performed to analyze the speedup obtained by time-parallelization. A fixed problem size is used with a $16^2$ base spatial mesh and four levels of refinement, leading to a five level AMR and multigrid hierarchy. Using the maximum stable time-step size on all levels leads to a total of 8960 time points on the finest temporal grid. For the time-sequential algorithm, the spatial parallelization is increased from 1 to 256 processors. The results are shown in Fig. 9.6. Spatial parallelization saturates fairly quickly in the time-sequential case, the maximum speedup due to spatial parallelization occurred at 64 processors in space but diminishing returns are seen after 8 processors. Two different time-parallel cases are run; the first solely parallelizes the time domain (no spatial parallelization) while the second uses 8 processors for spatial parallelization. Eight processors are used for spatial parallelization because that is the point in which additional processors provide severely diminishing returns in the time-sequential case. When only parallelizing the temporal domain, saturation occurs around 1024 processors and it is $7.0\times$ faster than time-sequential. When combining space and time parallelization, saturation is reached at 4096 processors with a speedup of $13.7\times$ over the time-sequential case.

# Chapter 10

# Results and Discussion of MGRIT with AMR

The valid MGRIT+AMR algorithm is eventually expected to solve practical fluid dynamics problems which are often characterized by hyperbolic dominant features. This is the second objective of the PinT research component. However, MGRIT has difficulties in converging a hyperbolic system [41] using traditional multigrid operations and space-time discretizations. To cope with this difficulty, a new method is developed by modifying the multigrid operators to enable MGRIT to solve highly turbulent flows by separation of scales. To aid the development of multigrid operators that can solve highly turbulent flows, an invisicd Taylor-Green vortex case is considered. Next, the challenges faced by MGRIT for turbulent flows are described in the context of the Taylor-Green vortex problem and various methods to overcome these challenges are evaluated. Results of this new method are discussed.

## 10.1 Inviscid Taylor-Green Vortex

In order to demonstrate the concepts, an infinite-Reynolds number Taylor-Green vortex case is used. In a fully-periodic cube of side-length $D$, an initial vortex is initialized by a sinusoidal initial velocity

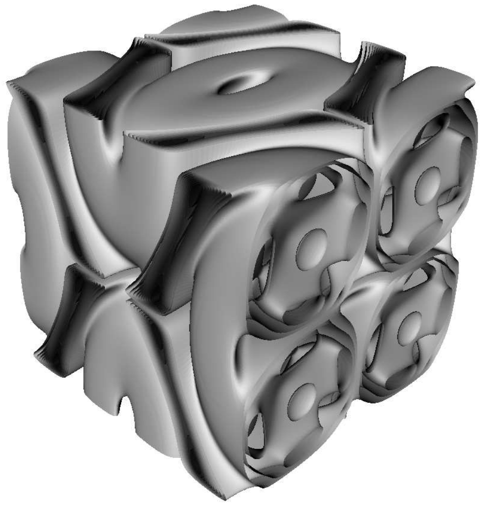
$$u = -U_0 \sin\left(\frac{n\pi x}{D}\right) \cos\left(\frac{n\pi y}{D}\right) \sin\left(\frac{n\pi z}{D}\right) \,, \tag{10.1}$$

$$v = -U_0 \cos\left(\frac{n\pi x}{D}\right) \sin\left(\frac{n\pi y}{D}\right) \sin\left(\frac{n\pi z}{D}\right) \,, \tag{10.2}$$

$$w = 0 \,, \tag{10.3}$$

$$p = p_0 + \frac{\rho_0 U_0^2}{16} \left(\cos\left(\frac{2n\pi x}{D}\right) + \cos\left(\frac{2n\pi y}{D}\right)\right) \left(\cos\left(\frac{2n\pi z}{D}\right) + 2\right) \,, \tag{10.4}$$

$$\rho = \frac{p\rho_0}{p_0} \,, \tag{10.5}$$


**(a)** $\tau = 2.1$.

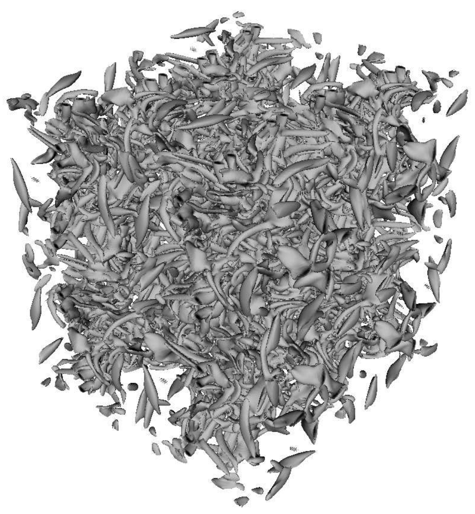

**(b)** $\tau = 18.7$.

**Figure 10.1:** Iso-surfaces of enstrophy in a Taylor-Green flow.

where $U_0$ is the magnitude of velocity fluctuation and $n$ is the number of vortices in the domain in each coordinate direction. This case has a Mach number of $\mathrm{Ma} = 0.1$ based on $U_0$, a Prandtl number for air of $\mathrm{Pr} = 0.71$, and a specific heat ratio $\gamma = 1.4$. A non-dimensional, charactersitic time, $\tau$ is defined as

$$\tau = t\frac{U_0}{D}, \tag{10.6}$$

where $t$ is the simulation time. Fig. 10.1 illustrates the iso-surfaces of enstrophy at $\tau = 2.1$ when initial roll-up occurs, and at $\tau = 18.7$ when the turbulence is fully developed. The transition to fully developed turbulence occurs by $\tau = 10$ in a decaying Taylor-Green flow. To test in a regime where a well distributed spectrum of energy presents, the solution is first advanced sequentially-in-time to characteristic time $\tau = 20$. The time-sequential solution at $\tau = 20$ is then used as the initial condition for PinT cases. A two-level MGRIT scheme is used with a two-level AMR scheme, the coarse spatial level contains $32^3$ cells and the fine spatial level contains $64^3$ cells. The stability constraint, imposed by the CFL condition, requires a temporal grid of 226 time points on the fine space-time grid per characteristic time.

## 10.2 Challenges of MGRIT Application to Turbulent Flows

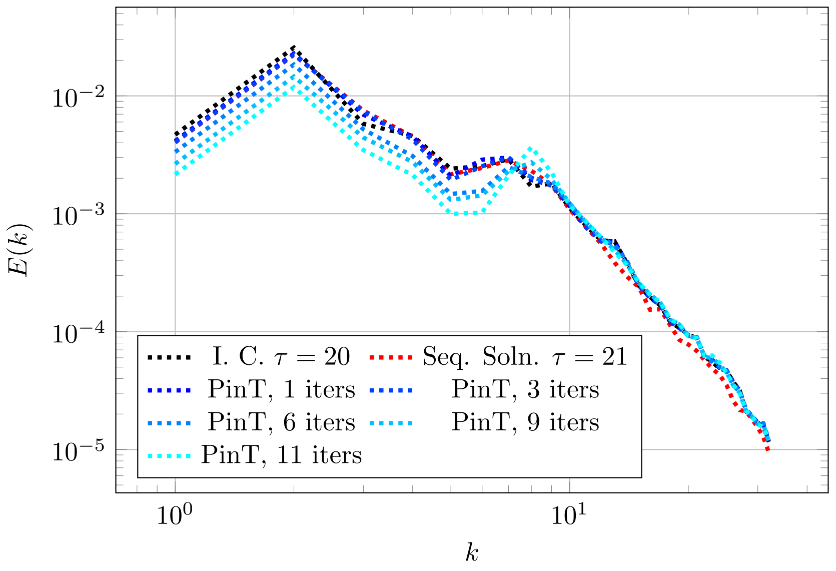


**Figure 10.2:** The energy spectrum for the inviscid Taylor-Green case at characteristic time $\tau = 21$ when the AMR with MGRIT operators are applied. As the MGRIT iterations increase, the solution deviates from the sequential solution.

The challenges faced by the standard multigrid operators used in the adaptive MGRIT algorithm can be clearly seen in the energy spectrum as shown in Fig. 10.2. The PinT solution is initialized at the $\tau = 20$ characteristic time, shown in black. In red is the sequential solution after advancing forward by one characteristic time, which represents the ideal solution obtained by MGRIT. In shades of blue are several iterations of the solution obtained with MGRIT at the same solution time as the sequentially-run case. As can be seen, the low-frequency modes are deviating from the sequential solution as more iterations are performed. Furthermore, the mid-range scales around $k = 8$ are increasing in energy above both the initial and sequential solutions. On the coarse mesh, the largest frequency that may be represented according to the Nyquist theorem is $k = 16$, while frequencies between $8 \leq k \leq 16$ may experience significant aliasing error. These poorly-represented scales on the coarse mesh create coarse-grid corrections that likely corrupt the solution, leading to divergence of the MGRIT algorithm and motivating the need to further separate the scales solved on the multigrid levels.

Walters et al. [75] demonstrated that by applying a stretched-vortex subgrid-scale (SGS) model at a coarser length scale than the grid filter, the large-scale dynamics converge independently of grid resolutions and numerical schemes. By using a coarser length scale for the SGS model, the high-frequency information is dampened or filtered out yet the low-frequency data remains well resolved. This demonstrates a separation of scales desirable for MGRIT applications, namely that the localized high-frequency information can be filtered out of the solution while still accurately resolving the large-scale dynamics. The large scale-dynamics are sufficient for many engineering applications.

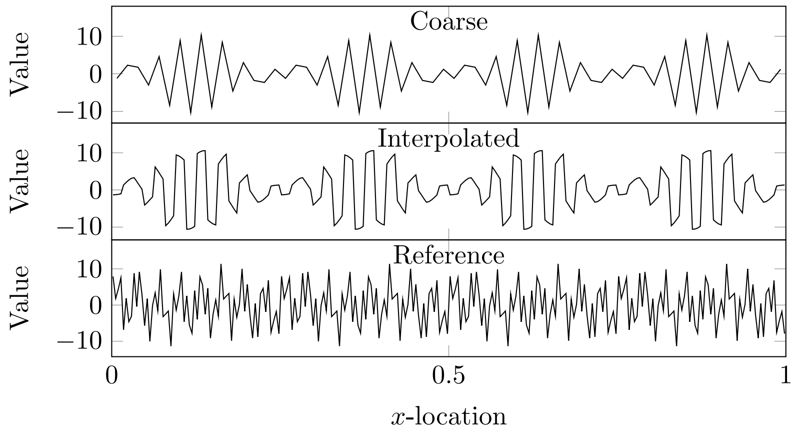


**Figure 10.3:** Representation of a sum of frequencies on a coarse mesh, interpolation to a fine mesh, and a reference representation of the frequencies on the fine mesh. The coarse mesh shows many aliasing errors of the frequencies, and the interpolation further exacerbates the problem.

For a demonstration of the need to filter out high frequency, consider a sum of sine waves

$$f(x) = \sin\left(8\frac{2\pi}{L}x\right) + 0.5\sin\left(12\frac{2\pi}{L}x\right) + 5\sin\left(32\frac{2\pi}{L}x\right) + 7.3\sin\left(100\frac{2\pi}{L}x\right) , \quad (10.7)$$

as a cell-averaged solution on a coarse mesh of 64 cells in a 1D periodic domain with length $L$. Note that the sine wave with the frequency of 100 cannot be properly represented on a mesh of 64 cells and so presents significant aliasing errors. Similarly, the frequency of 32 presents aliasing errors as it is right at the Nyquist frequency, especially if any phase shift is applied to the wave. This function is shown on the coarse 64-cell mesh in Fig. 10.3, demonstrating significant aliasing. The fourth-order conservative interpolation of the coarse mesh solution onto a fine mesh of 256 is also shown in Fig. 10.3, along with a reference solution of the sine waves on the 256-cell mesh. Shown in Fig. 10.4 is the frequency space of the coarse solution, reference solution, and

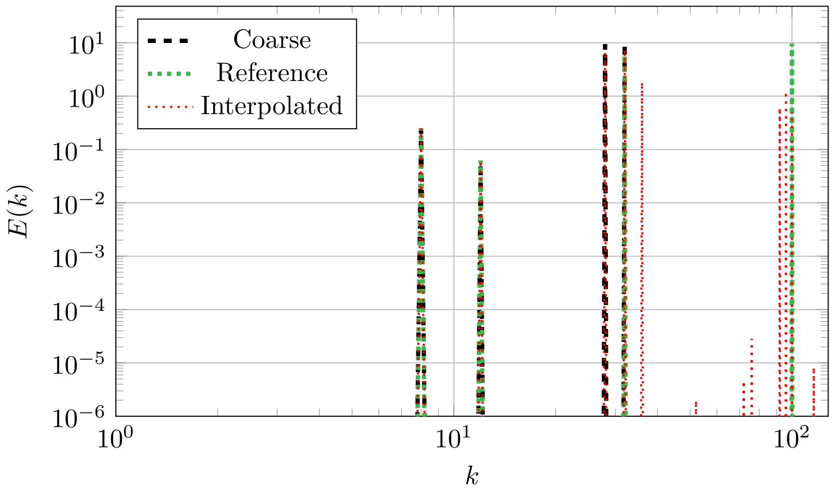


**Figure 10.4:** Energy spectrum of the coarse solution, reference solution, and interpolated solution. Interpolation of a solution with significant aliasing error introduces high-frequency information at magnitudes equivalent to the solution domain.

interpolated solution. The reference solution shows the waves at $k = \{8,\ 12,\ 32, 100\}$, correctly representing the frequencies of the analytic function. On the coarse solution, waves are shown at $k = \{8,\ 12,\ 32\}$ and an aliasing of highest frequency at $k = 28$. An interpolation of the coarse solution from 64 cells to a 256-cell mesh introduces high-frequency information at significant magnitudes. This high-frequency information quickly redistributes and, in fluid flows, that is typically done through the generation of acoustic energy which affects the large-scale dynamics. Therefore, to properly separate the scales on each multigrid level and prevent the coarse-grid-corrections from incorrectly adjusting high-frequency information and prevent adverse effects on the large-scale dynamics, high-frequency information from interpolations must be eliminated.

## 10.3 Techniques to Address the Challenges

To overcome the identified challenges, several techniques are developed to achieve rapid convergence of the large-scale dynamics and recover the high-frequency information in turbulent flows. Spectral filtering is the primary method to further separate the scales onto multigrid levels. Deconvolution is used to reconstruct some of the fine scales lost by interpolation during the FMG process. Lastly, to help reconstruct the high-frequencies faster, the C-points are stretched further apart in MGRIT to allow more relaxation on the F-points between coarse-grid corrections.

### 10.3.1 Filtering

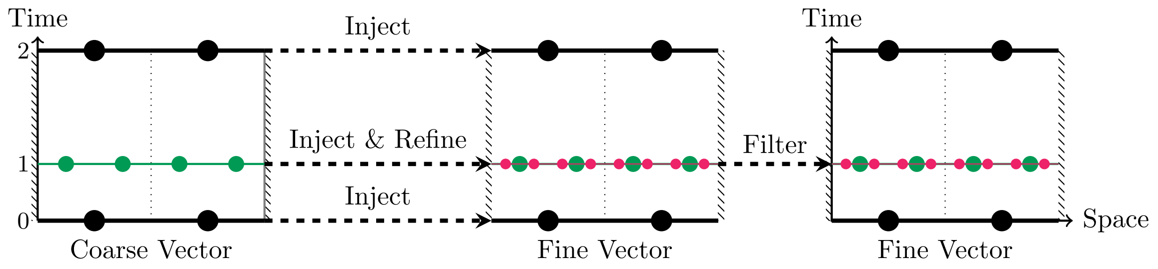
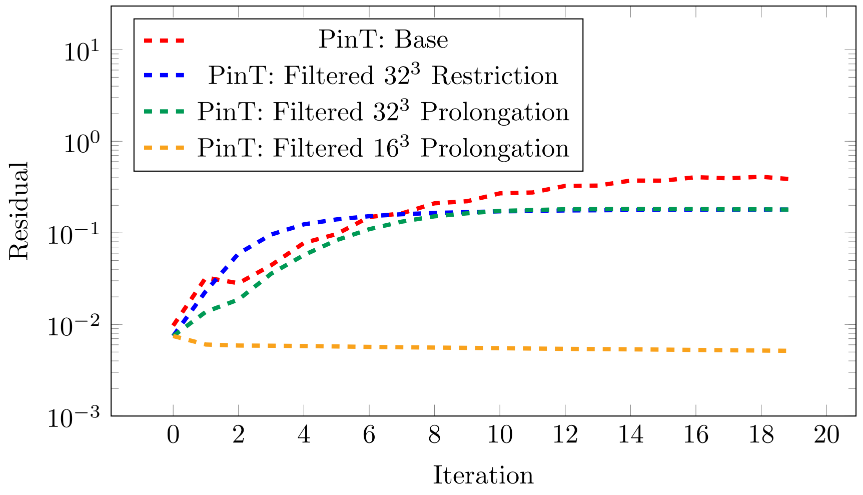


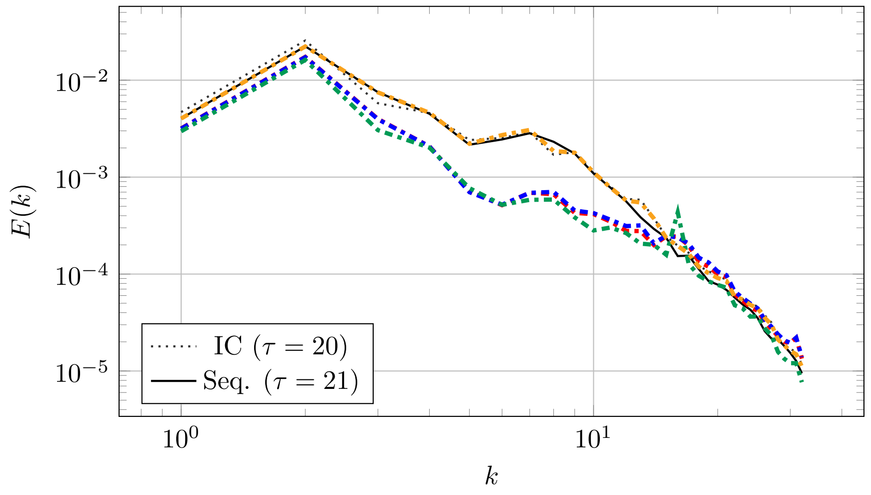
**Figure 10.5:** Prolongation is modified to add a spectral filter.

A low-pass sharp-isotropic spectral filter is employed to filter the solution upon prolongation as shown in Fig. 10.5. The filter is added to prolongation after spatial refinement in order to ensure only the low-frequency data is used in the coarse-grid-correction of the fine level. Through testing, a filter width of $\Delta_f = \Delta x_{\text{coarse}}$ is found insufficient to prevent corruption of the fine scales. This mirrors the lack of scheme-independence noted by Walters et al. [75], and a filter width of $\Delta_f = 2\Delta x_{\text{coarse}}$ is sufficient and used for this study.

Shown in Fig. 10.6b is the energy spectrum diagram for a PinT case with the initial condition at $\tau = 20$ and run over a one-$\tau$ time domain to a final time of $\tau = 21$. Also shown on the energy spectrum diagram are the initial condition at $\tau = 20$ and the sequential solution advanced to $\tau = 21$. The initial condition and initial guesses on the fine multigrid level are filled exactly from the $\tau = 20$ solution information, and the case is run for 12 MGRIT iterations.



**(a)** Residuals.



**(b)** Energy spectrum at iteration 12.

**Figure 10.6:** The base algorithm (optimized for diffusive flows) and the effect of filtering scales during restriction and prolongation.

If the MGRIT strategy optimized for diffusive flows is employed for the turbulent Taylor-Green flow, the solution does not converge to the sequential solution. In fact, the PinT solution diverges, as shown by the "Base" case in Fig. 10.6, which shows both the residual and the energy spectrum of the turbulent flow. The residual is $\boldsymbol{r} = \mathrm{A}\mathbf{U} - \boldsymbol{g}$ from Eqn. (8.3). To neglect the fine scale features, which are likely chaotic, the fine grid solution is first spatially coarsened to the resolution of the coarse spatial mesh before the residual is computed. This ensures the residual is assessing the convergence of the coarse scales.

After the implementation of the spectral filter, filtering is added separately to the restriction and prolongation operators. As demonstrated in Fig. 10.6, weak convergence was only observed when the prolonged coarse-grid correction was filtered to a resolution of $16^3$ before being added to the fine-grid solution. The resolution of $16^3$ corresponds to $2\Delta x_{\text{coarse}}$, which is analogous to Walters' application of a SGS model at a coarser length scale than the grid filter. Although the convergence is weak, the large-scale dynamics converge to the sequential solution. The high-frequency information has barely evolved and still closely follows the initial condition. As the small turbulence scales at $\tau = 20$ are nearly identical to those at $\tau = 21$, the influence of the small scales on the large scales are also nearly identical. This indicates that corruption of features in the range $8 < k < 16$ is caused by discretization error on the coarse grid. This error is caused by the PPM limiter where the turbulence as recognized as discontinuities, introducing additional dissipation. The limiter is necessary, however, as it is required to maintain stability for this inviscid flow. Therefore, the prolongation must only transfer corrections from the well-resolved scales on the coarse grid in order to solve turbulent flows with MGRIT, thus demonstrating the need for filtering during prolongation.

### 10.3.2 Deconvolution

In the previous section, the initial conditions and initial guesses were initialized to the sequential solution at $\tau = 20$. However, this is not ideal for general problems and the fine-grid solution will need to be approximated from the coarse-grid solution via the FMG startup procedure as

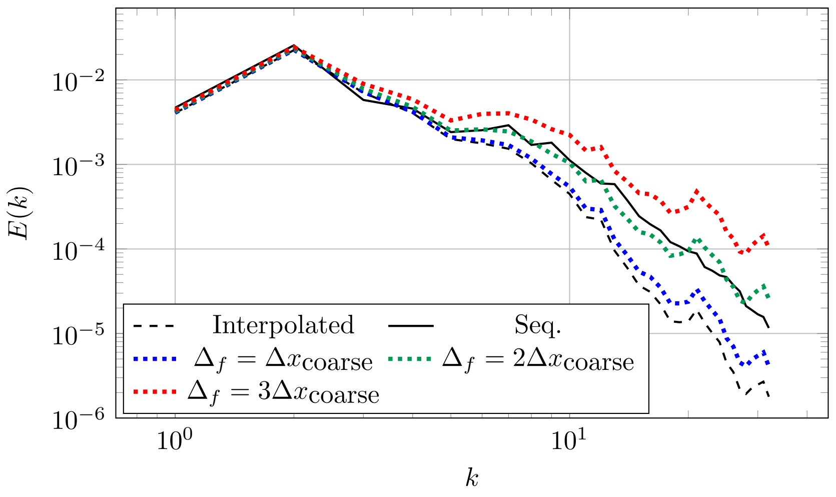


**Figure 10.7:** The effect on the initialization of the fine-grid solution as part of the FMG starup procedure by interpolation and interpolation followed by deconvolution with various filter widths.

demonstrated in Fig. 8.12. Spatial interpolation is used to fill the fine-grid solution, however this does not provide a complete population of the high-frequency scales. To this end, deconvolution is introduced which, for non-projective filters, is the inverse of filtering. Approximate deconvolution methods are well established for usage in structural subgrid-scale models [108–112] for incompressible and compressible turbulence. For this work, a differential deconvolution algorithm given by

$$\phi = \langle \phi \rangle - \frac{\Delta_f^2}{24} \Delta^{(2)} \langle \phi \rangle \, , \tag{10.8}$$

is employed, where $\Delta_f$ is the filter width, $\Delta^{(2)}$ is a second-order Laplacian, and $\langle \phi \rangle$ is the cell-averaged solution state.

The initial guess of the fine-grid solution from the FMG process is shown in Fig. 10.7 for interpolation alone, then interpolation followed by deconvolution with several different filter widths. Although $\Delta_f = 2\Delta x_{\text{coarse}}$ provides a close approximation of the sequential energy spectrum, the number of iterations to reach equilibrium in the fine scales is not significantly reduced compared to $\delta_f = \Delta x_{\text{coarse}}$. Furthermore, adding in excessive high-frequency energy content carries some risk. In the remaining results, the filter width of $\Delta_f = \Delta x_{\text{coarse}}$ was used, though a more rigorous study of the filter width's impact on different turbulent flows would be required to evaluate the best filter width.

### 10.3.3 Number of C-points

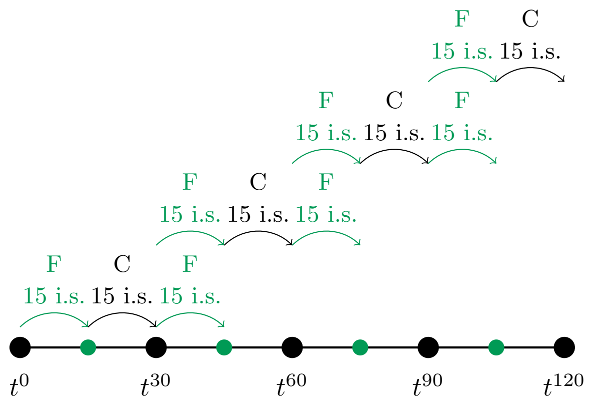


**Figure 10.8:** Internal steps (i.s.) allow for more applications of the time integrator in FCF-relaxation. The time point numbers indicate the number of steps taken by the explicit time-integration method.

The primary concern when solving turbulent flows with the MGRIT algorithm was in the convergence of the low-frequency modes as that is sufficient for many engineering applications. However, ideally the high-frequency modes would be reconstructed by PinT. To accelerate convergence of the high-frequency modes, additional relaxation must be applied on the finest MGRIT level relative to the error corrections from the coarse grid. With the existing MGRIT algorithm, the number of applications of the time integrator between C-points is equal to the coarsening ratio, $m$, which is typically $m = 2$ or $m = 4$ when using an explicit time integration method. To increase the relaxations on the fine grid, the multigrid F- and C-points are stretched further apart in the temporal domain and, to accommodate the explicit time integration method, internal steps are taken in-between. Internal steps are time integrations made by the explicit Runge-Kutta method within the stability region of the method, potentially tens or hundreds of times smaller than the time interval of the stretched-out F- and C-points. This concept is demonstrated in Fig. 10.8, where the large black points represent C-points and the small green points represent F-points. Each arrow shows 15 internal steps taken by the time integrator for each point in the multigrid algorithm. Each row represents a concurrent operation in FCF-relaxation, which begins at each C-point. Stretching the time between F- and C-points reduces the number of C-points in the time domain, and thus reduces the temporal parallelization. However, this is a reduction in parallelism corresponding to a reduction in the algorithm's memory footprint. Taking internal steps, therefore, often improves the overall performance of the algorithm. The ideal number of internal steps is currently determined by numerical experiments, and needs to balance relaxation to equilibrium in the fine scales to feedback from their influence on coarse scales.

For this section, the time domain is changed to the full simulation of the the Taylor-Green problem, i.e. $0 \leq \tau \leq 20$, and the fine scales are evolved to equilibrium. Consider the two extremes for the number of C-points in the temporal domain. In the first extreme, there is a C-point every $m$ time points, which is the number of C-points used thus far in this study. As previously demonstrated, this extreme leads to slow negligible convergence towards equilibrium in the high-frequency modes. On the other extreme, there are only two C-points: one at the initial condition and one at the final time point. This extreme would mean that a sequential solve is performed on the fine mesh at every iteration, that is no parallelization would be used. Furthermore, this corresponds to the high-frequency modes being resolved to their equilibrium value at each multigrid iteration. Ideally the high-frequency modes will reach equilibrium at the same number of multigrid iterations as the low-frequency modes converge, therefore a balanced number of C-points would require a few multigrid iterations which is dictated by the number of C-points in the temporal domain.

However, as the number of C-points in the domain is reduced, so does the maximum allowable number of iterations. This is because the exact solution is propagated sequentially forward by one C-point in every multigrid iteration. When the number of multirid iterations performed equals the number of C-points in the temporal domain, the solution has been sequentially stepped on the fine mesh, obviating the point of MGRIT. This is the exactness property of MGRIT. Convergence must be achieved in fewer iterations to provide an opportunity of a parallel speedup.

Cases were run with 5, 10, and 20 total C-points, which corresponds to temporal separations of 4, 2, and 1 $\tau$ between C-points. Each case was iterated until the exactness property reached the

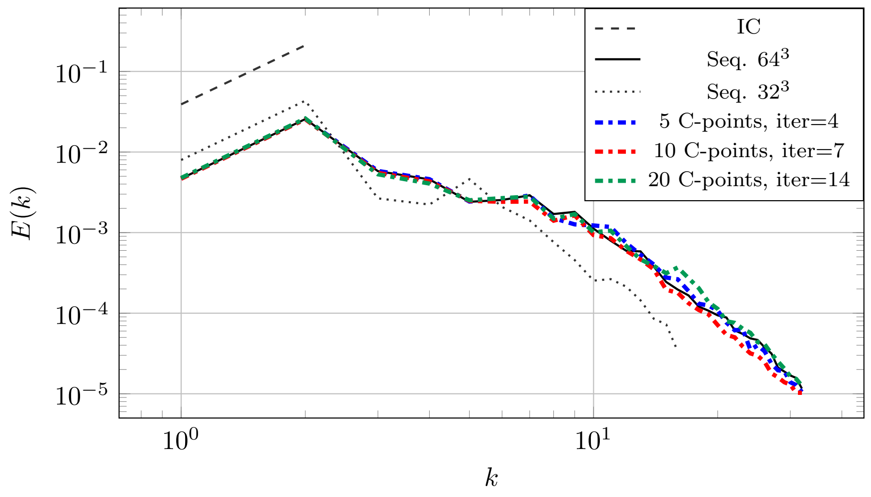


**Figure 10.9:** Energy spectrum at $\tau = 20$ of a Taylor-Green flow solved from $\tau = 0$ to $\tau = 20$. Comparison of cases with different numbers of C-points.

final time point so that the solution at each iteration may be thoroughly studied. The results of this are shown in Fig. 10.9, along with the initial condition, the coarse-grid $32^3$ sequential solution, and the fine-grid $64^3$ sequential solution. The coarse-grid solution is shown to demonstrate the effect of the high-frequency modes on the low-frequency modes. If the coarse-grid solution's low-frequency modes were the same as the fine-grid solution's, then little effort would have been required of the MGRIT algorithm as the FMG process would nearly exactly set the correct solution.

The PinT cases are shown after convergence, which for the 5 C-point case required 4 iterations (80% of the iterations to reach the exactness property). The 10 and 20 C-point cases required 70% of the iterations to reach the exactness property, or 7 and 14 iteartions, respectively. For this case, the optimal number of C-points likely lies between 5 and 10 C-points. The number of multigrid levels may affect the optimal choice of C-points, which will need further investigations.

## 10.4 Results of the Inviscid Taylor-Green Vortex Problem

With the new strategies of filtering, deconvolution, and stretched out C-points, this modified MGRIT algorithm is re-applied to solve the inviscid Taylor-Green vortex problem. First, convergence of both the energy spectrum and the MGRIT residuals is detailed in Section 10.4.1. Second,

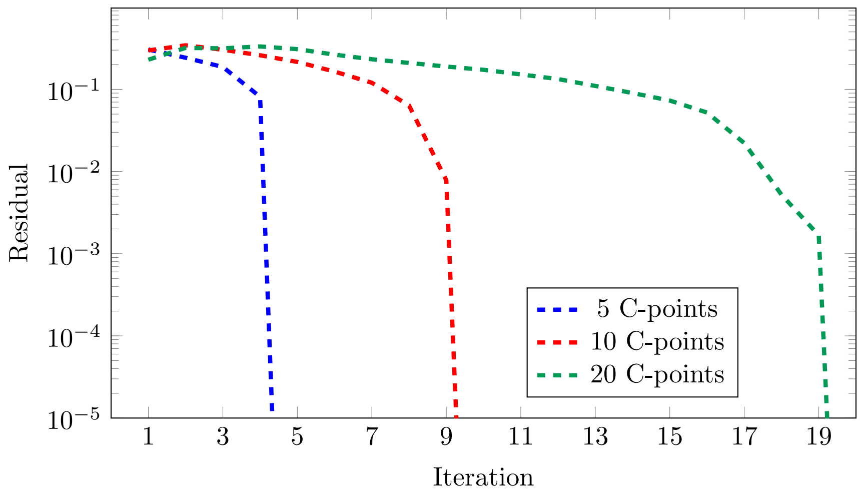


**Figure 10.10:** Residual at each iteration. The exactness property takes effect on iteration 5, 10, and 20 for the case with 5 C-points, 10 C-points, and 20 C-points, respectively.

a performance comparison between the solution obtained by the new final MGRIT algorithm and a sequential solution is presented in Section 10.4.2.

### 10.4.1 Convergence

The residuals for the 5, 10, and 20 C-point cases are shown in Fig. 10.10, and they demonstrate the exactness property as they go to machine zero at the 5th, 10th, and 20th C-point, respectively. Each demonstrates weak convergence until the exactness property is reached. The energy spectrum for the 5 and 20 C-point cases are shown in Figs. 10.11–10.12 for a number of iterations. Despite the weak convergence of the residual, convergence of the low-frequency and high-frequency scales occurs before the exactness property, demonstrating that the different scales of turbulence may be utilized to achieve MGRIT convergence. Despite this convergence, the intermediate scales from $4 \leq k \leq 8$ require the most iterations to approach the sequential solution. This is shown by the fact that the low and high-frequency information is reasonably converged by the 12th and 2nd iteration of Figs. 10.11–10.12, respectively, while the intermediate scales need up to iterations 4 and 16 to converge. Currently, "convergence" is defined by a qualitative comparison of the difference between the PinT and sequential energy spectra. Research is ongoing into a quantitative measure of convergence. While the residual of Fig. 10.10 seems to provide a quantitative measure, there is

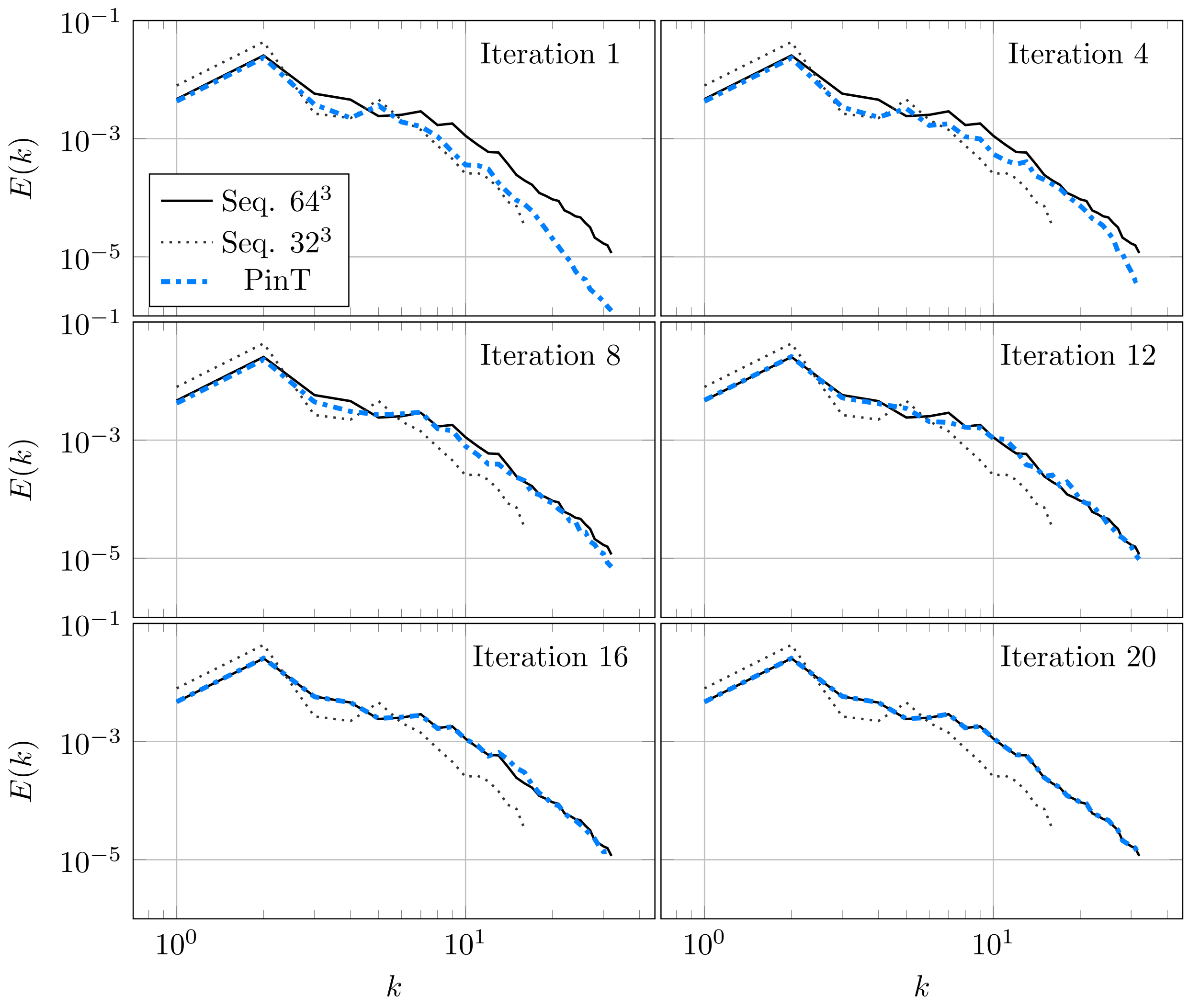


**Figure 10.11:** Energy spectrum at $\tau = 20$ of a Taylor-Green flow solved from $\tau = 0$ to $\tau = 20$ using an MGRIT algorithm with 20 C-points.

obviously a discrepancy between the residual and the qualitative analysis of the energy spectra that must be accounted for.

### 10.4.2 Performance

**Table 10.1:** Wall-clock run time information for 2-level cases with varying numbers of C-points.

| Case | Total Time (s) | FMG (s) | Time/Iter (s) | Final Sweep (s) |
|---|---|---|---|---|
| PinT, 10 C-Pts | 10 266.62 | 321 | 1105 | 362.76 |
| PinT, 20 C-Pts | 13 561.42 | 254 | 650 | 261.42 |
| Seq. | 3696.46 | – | – | – |

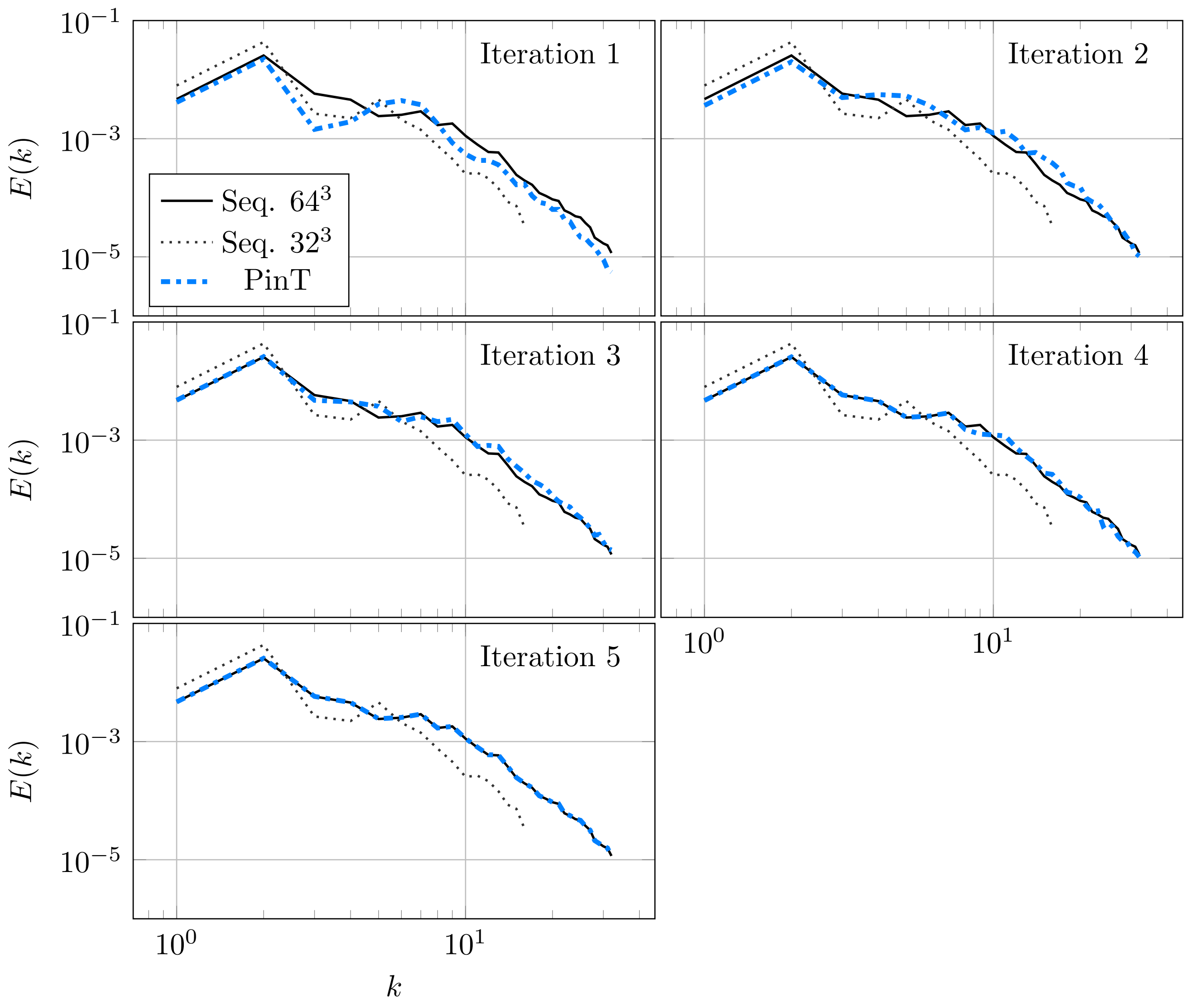


**Figure 10.12:** Energy spectrum at $\tau = 20$ of a Taylor-Green flow solved from $\tau = 0$ to $\tau = 20$ using an MGRIT algorithm with 5 C-points.

To evaluate the performance characteristics of the algorithm, a 2-level Taylor-Green vortex case is run with varying numbers of C-points, and compared against the sequential case. The results are shown in Table 10.1, where the FMG time is the time spent in the FMG process, and the Final Sweep is time spent after convergence performing one last FCF-relaxation to write out the final solution information. These cases are are run from $\tau = 0$ to $\tau = 20$ with 64 spatial processors. The PinT cases used 10 C-points for temporal parallelization and were run until the exactness property, which is 10 iterations for the 10 C-point case and 20 iterations for the 20 C-point case. As demonstrated in the previous sections, fewer iterations are typically needed to converge all of the scales. However, only 3 MGRIT iterations in the 10 C-point case leads to a longer solution time than the sequential case. Similarly, the 20 C-point case has a longer solution time after only 5 iterations.

**Table 10.2:** Wall-clock run time information for a 3-level case.

| Case | Total Time (s) | FMG (s) | Time/Iter (s) | Final Sweep (s) |
|---|---|---|---|---|
| PinT | 13 819.96 | 130 | 2148 | 419.96 |
| Seq. | 9048.577 | – | – | – |

One of the biggest effects on MGRIT solution time is the coarse grid solve, which is performed sequentially at each iteration. By increasing the fine-grid problem size through spatial refinement, the relative cost of the coarse-grid becomes negligible. Therefore, for this test 3 levels of MGRIT are used with the fine grid consisting of a $128^3$ solution mesh, and the coarse and intermediate remain at $32^3$ and $64^3$, respectively. Both cases use 128 processors for spatial parallelization, while the PinT case uses 12 temporal processors for a total of 1536 processors. This case is evolved from $\tau = 0$ to $\tau = 6$, with the PinT case using 6 C-points. The results, in Table 10.2, show a total PinT run time for 6 iterations, which reaches the exactness property. For this particular case, the low and high-frequency information shows convergence at 3 iterations, which requires a wall clock time of $6993.93\,\mathrm{s}$, or over $2000\,\mathrm{s}$ faster than the sequential case. Based on these results, it is expected that with finer spatial meshes MGRIT may be exploited to achieve significant speedups in turbulent flows.

# Chapter 11

# Conclusion and Future Work

## 11.1 Conclusions

The research includes two components: the ARK4+AMR algorithm and the MGRIT+AMR algorithm. In the first component, the 2-ARK$_4$(3)$6L$[2]$SA$ scheme has been implemented with AMR and validated in Chord, a fourth-order finite-volume CFD software infrastructure where numerical capabilities of solving complex fluid dynamics problems include the mapped multiblock technique for accommodating real geometries, the large eddy simulation for turbulence modeling, and the species transport and chemical kinetics of a range of fuels (hydrogen and hydrocarbons) for simulating reacting flows. The AMR technique is one of the foundational numerical features in the framework for maximizing computational efficiency in solving stiff problems. While ARK4 is not new, a successful building of ARK4 into this complex software framework has required a few new strategies, particularly in the context of AMR. For example, the integration overcomes the challenge of subcycling by utilizing dense output to enable temporal interpolation of ghost cells on finer levels. The incorporation of ARK4 has enabled Chord to efficiently model practical combustion problems occurring in a realistic combustor geometry with stiff chemical mechanisms, where the standard ERK4 method is intractable.

The computational efficiency and accuracy have been demonstrated and assessed for the lean $C_3H_8$-air premixed flame in the bluff-body combustor, using the chemical kinetics of 24 species and 66 reactions. The $C_3H_8$ combustion is much stiffer than the $CH_4$-air combustion under the same operating conditions. The ARK4 time step on level 0 (the base mesh) is on the order of $8.0 \times 10^{-7}\,\mathrm{s}$ while the global chemical step is on the order of $8.0 \times 10^{-11}\,\mathrm{s}$; that is a 10000$\times$ increase in time step sizes. On level 1, the ARK4 time step is about $4.0 \times 10^{-7}\,\mathrm{s}$ while the global chemical time step is about $2.0 \times 10^{-10}\,\mathrm{s}$; that is a 2000$\times$ increase in time step size. If using ERK4 with a uniform mesh resolution matching the fine AMR level, this $C_3H_8$-air premixed flame in the bluff-body combustor simulation would take a hundred years of wall-clock time on 4096 CPUs to achieve one flow-through time, which is evidently impractical. However, using ARK4, the simulation can be performed efficiently in days on 4096 CPUs. As shown, the global chemical time step increases as the mesh resolution is refined. At some fine level, one would expect the chemical time step would eventually not be the limiting time scale for the overall computation. At this point, treating the chemical term implicitly becomes unnecessary. Therefore, in the algorithm, ARK4 is programmed to respond to the level of AMR where it is automatically engaged or disengaged based on the chemical time scale. It is anticipated that the viscous term might become the stiff term as the mesh is refined. Then, the viscous term would be treated implicitly. Nevertheless, for all applications of interest to the present study, the viscous physics has always been less stiff than the chemical reactions.

Another major challenge is the species correction strategy. A new robust method is implemented to cope with the unphysical phenomenon in species caused by numerical errors. The nonlinear solvers employed during the solution process experienced convergence difficulty when a unphysical species mass fractions occurs. To prevent this, an optimization strategy based on the L-BFGS solver is employed to redistribute the species mass fraction when either a single species or the sum of all species is out of bounds. This method has proven to be superior to any simple redistribution scheme by re-normalizing and weighting. The analytical chemical Jacobian is found to perform better than the approximate one in terms of computational efficiency and accuracy for the stiff chemical reactions.

In the second component, a new adaptive parallel space-time algorithm is developed. This algorithm is created by coupling the space-time adaptivity of a structured AMR algorithm with the temporal parallelization of MGRIT. Implementation details and challenges are discussed when adding adaptivity to MGRIT, specifically the incorporation of subcycling into relaxation.

Verification and validation of the new adaptive parallel space-time algorithm is performed using the Couette flow and Stokes second problem. Grid convergence studies show that the error converges at the same rate as the time-sequential algorithm, and is the same magnitude as the time-sequential algorithm. Conservation of mass, momentum, and energy is maintained thanks to careful construction of the space-time mesh, and conservation is verified by comparison to the time-sequential algorithm. The multigrid algorithm's residuals converge at a rate that is comparable to other MGRIT studies. The residual convergence rate is also consistent between different numbers of temporal levels on the same fine space-time mesh. A $4.3\times$ speedup was achieved when comparing to single-spatial-processor cases on the Couette flow. When comparing the best performing time-sequential case with the best performing time-parallel case, a $1.5\times$ speedup was achieved. The time periodic case shows a greater speedup, with a $7.0\times$ speedup when parallelizing only the temporal domain, and a speedup of $13.7\times$ when parallelizing both space and time.

Lastly, the adaptive parallel space-time algorithm is extended to solve a purely inertial turbulent flow. The prolongation operator is extended with a spectral filter to properly separate the scales solved at each level of the multigrid algorithm, and ensure that the low-frequency modes are correctly resolved on the fine level. On application to an infinite-Reynolds number Taylor-Green vortex test case, the low-frequency modes converge within several iterations.

## 11.2 Original Contributions

Many novel contributions are made towards the time integration of fluid flow governed by the thermally perfect, compressible Navier-Stokes equations in complex geometry. Some important original features of the research are as follows:

1. Developed robust and consistent strategies for coping with unphysical phenomena. For the reacting source term integrated implicitly in time, stiff reaction mechanisms lead to unphysical temperature and species mass fractions during nonlinear iterations. A novel chemical species constraint method is developed for the nonlinear solver to provide stability and maintain a reasonable thermodynamic state. In the rare situation where the constraints fail to provide a physical thermodynamic state, a nonlinear optimizer based on L-BFGS was created and implemented to find a thermodynamic state that allows the simulation to advance.

2. Integrated ARK4 and AMR with high-order spatial discretization schemes. To use ARK4 with AMR, time interpolation of invalid ghost cell data needs to be performed at each stage of the Runge-Kutta scheme. The development and implementation of dense output for temporal interpolations on AMR hierarchies allows for fourth-order accuracy in both the spatial and temporal discretization.

3. Applied the AKR4+AMR algorithm to practical combustion occurring in complex geometry with large, stiff chemical kinetics. Previous applications of ARK4 with AMR were performed on simple geometries and with small reaction mechanisms. This work introduces ARK4 with AMR to the complex practical geometry of a bluff-body combustor and the use of the much larger $C_3H_8$-air reaction mechanism.

4. Conducted a comprehensive parametric sensitivity study. This work contributes to the literature and understanding of the impacts of AMR, case geometry, and chemical reaction mechanisms on the time scales of the inertial, viscous, and chemical physics.

5. Created an efficient solution-adaptive space-time parallel method. This required extensive development of novel multigrid operations for relaxation with subcycling and prolongation and restriction with adaptive grid hierarchies. The creation of space-time adaptive grids for use with MGRIT requires a unique use of FMG to generate successively finer space-time meshes and in the process construct a hierarchy of multigrid levels.

6. Implemented novel restriction and prolongation operators to enable the adaptive MGRIT algorithm to solve turbulence.

## 11.3 Future Work

There are many areas for future work, ranging from improvements to the existing algorithm to new areas of exploration based on the results.

- Applications of ARK4 to cases with strong shocks provided less-than-satisfactory results. For example, a reacting Richtmyer-Meshkov instability case was run and found to be numerically unstable when using the inertial scales for the time step sizes. Limiting the step size with the PID-controller resulted in step sizes that were smaller than the chemical time scales. The causes of the numerical instability need to be further investigated, and potentially more severe constraints on the nonlinear solver employed. Another approach would be to evaluate different error-based step size controls for efficacy with strong shocks.

- MGRIT has been demonstrated for the infinite-Reynolds number Taylor-Green vortex, however, MGRIT does not show convergence of residuals. New convergence criteria based on the large-scale dynamics need to be devised and evaluated. More test problems such as a time-evolving mixing-layer case and decaying homogeneous turbulence would further prove the robustness of the algorithm. Of particular interest would be the interaction of turbulence models with MGRIT algorithm.

- The overarching goal of the MGRIT algorithm is to enable a time-parallel solve of reacting turbulent flow on complex geometry. In order to realize that goal, the MGRIT algorithm must be evaluated on flows containing inertial and diffusive fluxes. Further development of the MGRIT algorithm will be required to achieve convergence of scales of interest. Furthermore, the reacting source term is untested for MGRIT. While source terms have not presented a problem to MGRIT in the past, it is possible the chaotic nature of turbulent chemistry may prove challenging.

# Bibliography

[1] E. Lindblad, D. M. Valiev, B. Müller, J. Rantakokko, P. Lötstedt, and M. A. Liberman. Implicit-explicit runge-kutta method for combustion simulation. In *Proc. ECCOMAS CFD Conference 2006*, pages 20–. Tech. Univ. Delft, 2006.

[2] X. Gao and C. P. T. Groth. Parallel adaptive mesh refinement scheme for three-dimensional turbulent non-premixed combustion. AIAA 2008-1017, 46th AIAA Aerospace Sciences Meeting, 2008.

[3] L. D. Owen, S. M. J. Guzik, and X. Gao. A fourth-order finite-volume algorithm for compressible flow with chemical reactions on mapped grids. Denver, CO, USA, 2017. 23rd AIAA Computational Fluid Dynamics Conference, AIAA Aviation Forum.

[4] 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. *Int. J. Numer. Meth. Fluids*, 40:209–216, 2002.

[5] J. B. Bell, M. S. Day, J. F. Grcar, J. F. Driscoll, M. J. Lijewski, and S. A. Filatyev. Numerical simulation of a laboratory-scale turbulent slot flame. *Proc. Combust. Inst*, 31:1299–1307, 2007.

[6] X. Gao, M. S. Day, and J. B. Bell. Characterization of freely propagating hydrogen flames. In *Fall Technical Meeting of the Western States Section of the Combustion Institute*, 2009.

[7] P. Colella, M. R. Dorr, J. A. F. Hittinger, and D. F. Martin. High-order finite-volume methods in mapped coordinates. *J. Comput. Phys.*, 230:2952–2976, 2011.

[8] Q. Zhang, H. Johansen, and P. Colella. A fourth-order accurate finite-volume method with structured adaptive mesh refinement for solving the advection-diffusion equation. *SIAM J. Sci. Comput.*, 34(2):B179–B201, January 2012.

[9] D. F. Martin, P. Colella, M. Anghel, and F. J. Alexander. Adaptive mesh refinement for multiscale nonequilibrium physics. *Comput. Sci. Eng.*, 7(3):24–31, May 2005.

[10] D. F. Martin, P. Colella, and D. Graves. A cell-centered adaptive projection method for the incompressible Navier–Stokes equations in three dimensions. *J. of Comput. Phys.*, 227(3):1863–1886, January 2008.

[11] X. Gao and C. P. T. Groth. A parallel adaptive mesh refinement algorithm for predicting turbulent non-premixed combusting flows. *Int. J. Comut. Fluid Dyn.*, 20(5):349–357, 2006.

[12] X. Gao and C. P. T. Groth. A parallel solution-adaptive method for three-dimensional turbulent non-premixed combusting flows. *J. Comput. Phys.*, 229:3250–3275, 2010.

[13] X. Gao, S. Northrup, and C.P.T. Groth. Parallel solution-adaptive method for two-dimensional non-premixed combusting flows. *Prog. Comput. Fluid Dyn. Int. J.*, 11(2):76–95, 2011.

[14] M. S. Day, X. Gao, and J. B. Bell. Properties of lean turbulent methane-air flames with significant hydrogen addition. *Proc. Combust. Inst.*, 33(1):1601–1608, 2011.

[15] P. McCorquodale and P. Colella. A high-order finite-volume method for conservation laws on locally refined grids. *Comm. App. Math. Comput. Sci.*, 6(1):1–25, 2011.

[16] X Gao, S. M. J. Guzik, and P. Colella. Fourth order boundary treatment for viscous fluxes on cartesian grid finite-volume methods. AIAA 2014-1277, 52nd AIAA Aerospace Sciences Meeting, 2014.

[17] L. D. Owen, S. M. Guzik, and X. Gao. High-order CFD modeling of multispecies flows. In *Fall Meeting of the Western States Section of the Combustion Institute*, October 2015. WSSCI 2015-134IE-0030.

[18] S. M. J. Guzik, P. McCorquodale, and P. Colella. A freestream-preserving high-order finite-volume method for mapped grids with adaptive-mesh refinement. AIAA 2012-0574, 50th AIAA Aerospace Sciences Meeting, 2012.

[19] S. M. Guzik and C. P. T. Groth. Heterogeneous parallelism of high-order residual distribution schemes using central and graphics processing units. In *Proceedings of the 21st International Conference on Parallel Computational Fluid Dynamics*, pages 267–271, 2009.

[20] J. B. Bell, M. S. Day, A. S. Almgren, M. J. Lijewski, C. A. Rendleman, R. K. Cheng, and I. G. Shepherd. *Simulation of Lean Premixed Turbulent Combustion*, volume 46 of *Journal of Physics Conference Series: SciDAC 2006 (W. Tang, Ed.)*. Institute of Physics Publishing, Denver, CO, 2006.

[21] J. Bell. Combustion simulation on next generation architectures. Presentation in Los Alamos National Laboratory, May 2011.

[22] M. S. Day, J. B. Bell, X. Gao, and P. Glarborg. Numerical simulation of nitrogen oxide formation in lean premixed turbulent h2/o2/n2 flames. *Proc. Combust. Inst.*, 33(1):1591–1599, 2011.

[23] X. Gao and C. P. T. Groth. Parallel adaptive mesh refinement scheme for turbulent non-premixed combusting flow prediction. AIAA 2006-1448, 44th AIAA Aerospace Sciences Meeting, 2006.

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

[25] J. B. Bell, M. S. Day, X. Gao, and M. J. Lijewski. Simulation of nitrogen emissions in a low swirl burner. *J. Phys.: Conf. Ser.*, July 2010.

[26] H. D. Simon, editor. *Parallel Computational Fluid Dynamics: Implementations and Results*. MIT Press, Cambridge, MA, USA, 1992.

[27] C. Groth, D. De Zeeuw, K. Powell, T. Gombosi, and Q. Stout. A parallel solution-adaptive scheme for ideal magnetohydrodynamics. In *14th Computational Fluid Dynamics Conference*, Norfolk,VA,U.S.A., November 1999. American Institute of Aeronautics and Astronautics.

[28] F. Miniati and P. Colella. Block structured adaptive mesh and time refinement for hybrid, hyperbolic + N-body systems. *J. Comput. Phys.*, 227(1):400–430, November 2007.

[29] X. Gao and S. M. J. Guzik. A fourth-order scheme for the compressible Navier-Stokes equations. AIAA 2015-0298, 53rd AIAA Aerospace Sciences Meeting, 2015.

[30] S. M. Guzik, X. Gao, L. D. Owen, P. McCorquodale, and P. Colella. A freestream-preserving fourth-order finite-volume method in mapped coordinates with adaptive-mesh refinement. *Comput. Fluids*, 123:202–217, 2015.

[31] S. M. Guzik, X. Gao, and C. Olschanowsky. A high-performance finite-volume algorithm for solving partial differential equations governing compressible viscous flows on structured grids. *Comput. Math Appl.*, 72:2098–2118, 2016.

[32] X. Gao, L. D. Owen, and S. M. J. Guzik. A parallel adaptive numerical method with generalized curvilinear coordinate transformation for compressible Navier-Stokes equations. *Int. J. Numer. Meth. Fluids*, 82:664–688, 2016.

[33] X. Gao, L. D. Owen, and S. M. Guzik. A high-order finite-volume method for combustion. AIAA 2016-1808, 54th AIAA Aerospace Sciences Meeting, 2016.

[34] L. D. Owen, X. Gao, and S. M. Guzik. Techniques for improving monotonicity in a fourth-order finite-volume algorithm solving shocks and detonations. *J. Comput. Phys.*, 415, 2020.

[35] XBraid: Parallel multigrid in time. https://github.com/XBraid/xbraid/.

[36] R. D. Falgout, A. Katz, Tz. V. Kolev, J. B. Schroder, A. Wissink, and U. M. Yang. Parallel time integration with multigrid reduction for a compressible fluid dynamics application. Technical Report LLNL-JRNL-663416, Lawrence Livermore National Laboratory, 2015.

[37] S. Friedhoff, J. Hahne, I. Kulchytska-Ruchka, and S. Schöps. Exploring parallel-in-time approaches for eddy current problems. In *Progress in Industrial Mathematics at ECMI 2018*, volume 30. Mathematics in Industry, 2018.

[38] S. Günther, N. R. Gauger, and J. B. Schroder. A non-intrusive parallel-in-time adjoint solver with the XBraid library. *Comput. Vis. Sci.*, 19(3-4):85–95, 2018.

[39] S. Günther, L. Ruthotto, J.B. Schroder, E.C. Cyr, and N.R. Gauger. Layer-parallel training of deep residual neural networks. *SIAM J. Math. Data Sci.*, 2, 2020.

[40] A. J. Howse, H. D. Sterck, R. D. Falgout, S. MacLachlan, and J. Schroder. Parallel-in-time multigrid with adaptive spatial coarsening for the linear advection and inviscid burgers equations. *SIAM J. Math. Data Sci.*, 41(1):A538–A565, 2019.

[41] Hans De Sterck, Robert D. Falgout, Stephanie Friedhoff, Oliver A. Krzysik, and Scott P. MacLachlan. Optimizing MGRIT and Parareal coarse-grid operators for linear advection. *arXiv:1910.03726 [cs, math]*, October 2019.

[42] R. D. Falgout, T. A. Manteuffel, J. B. Schroder, and B. Southworth. Parallel-in-time for moving meshes. Technical Report LLNL-TR-681918, Lawrence Livermore National Laboratory, 2016.

[43] M. Lecouvez, R. D. Falgout, C. S. Woodward, and P. Top. A parallel multigrid reduction in time method for power systems. *Power and Energy Society General Meeting (PESGM)*, pages 1–5, 2016.

[44] J. B. Schroder, M. Lecouvez, R. D. Falgout, C. S. Woodward, and P. Top. Parallel-in-time solution of power systems with scheduled events. *Power and Energy Society General Meeting (PESGM)*, pages 1–5, 2018.

[45] A. Hessenthaler, D. Nordsletten, O. Röhrle, J. B. Schroder, and R. D. Falgout. Convergence of the multigrid-reduction-in-time algorithm for the linear elasticity equations. *Numer. Linear Algebra Appl.*, 25(3):e2155, 2018.

[46] R. D. Falgout, S. Friedhoff, Tz. V. Kolev, S. P. MacLachlan, and J. B. Schroder. Parallel time integration with multigrid. *SIAM J. Sci. Comput.*, 36(6):C635–C661, 2014.

[47] R. D. Falgout, S. Friedhoff, S. Vandewalle, Tz. V. Kolev, S. P. MacLachlan, and J. B. Schroder. Multigrid methods with space-time concurrency. *Comput. Vis. Sci.*, 18(4-5):123–143, 2017.

[48] X. Yue, S. Shu, X. Xu, W. Bu, and K. Pan. Parallel-in-time multigrid for space–time finite element approximations of two-dimensional space-fractional diffusion equations. *Comput. Math. Appl.*, 78(11):3471–3484, December 2019.

[49] S. Günther, N. R. Gauger, and J. B. Schroder. A non-intrusive parallel-in-time approach for simultaneous optimization with unsteady PDEs. *Optim. Methods Softw.*, pages 1–16, 2018.

[50] C. Kennedy and M. Carpenter. Additive Runge-Kutta schemes for convection-diffusion-reaction equations. *Appl. Numer. Math.*, 44:139–181, 2003.

[51] C. M. Chaplin. *An improved all-speed projection algorithm for low Mach number flows*. PhD thesis, UC Berkeley, 2018.

[52] A. J. Chorin. Numerical solution of the Navier-Stokes equations. *Math. Comput.*, 22(104):745–762, 1968.

[53] G. Strang. *Computational Science and Engineering*. Wellesley: Wellesley-Cambridge Press, 2007.

[54] S. MacNamara and G. Strang. *Operator Splitting*, pages 95–114. Springer International Publishing, Cham, 2016.

[55] S. V. Patankar and D. B. Spalding. A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. *Int. J. Heat Mass Transf.*, 15(10):1787 – 1806, 1972.

[56] S. Patankar. *Numerical Heat Transfer and Fluid Flow*. CRC Press, New York, 1st edition edition, January 1980.

[57] W. Hundsdorfer and S. J. Ruuth. IMEX extensions of linear multistep methods with general monotonicity and boundedness properties. *J. Comput. Phys.*, 225(2):2016–2042, August 2007.

[58] M. J. Gander. 50 years of time parallel time integration. In T. Carraro, M. Geiger, S. Körkel, and R. Rannacher, editors, *Multiple Shooting and Time Domain Decomposition*, pages 69–114. Springer, 2015.

[59] U. Langer, S. Matculevich, and S. Repin. Guaranteed error bounds and local indicators for adaptive solvers using stabilised space–time IgA approximations to parabolic problems. *Comput. Math. Appl.*, 78(8):2641–2671, October 2019.

[60] O. Steinbach. Space-time finite element methods for parabolic problems. *Comput. Meth. Appl. Math.*, 15(4), January 2015.

[61] O. Steinbach and H. Yang. Comparison of algebraic multigrid methods for an adaptive space-time finite-element discretization of the heat equation in 3D and 4D: Comparison of AMG for adaptive space-time FEM in 3D and 4D. *Numer. Linear Algebra Appl.*, 25(3):e2143, May 2018.

[62] M. Adams, P. Colella, D. T. Graves, J. N. Johnson, H. S. Johansen, N. D. Keen, T. J. Ligocki, D. F. Martin, P. W. McCorquodale, D. Modiano, P. O. Schwartz, T. D. Sternberg, and B. Van Straalen. Chombo software package for AMR applications - design document. Technical Report LBNL-6616E, Lawrence Berkeley National Laboratory, 2014.

[63] L. D. Owen, S. M. Guzik, and X. Gao. A high-order adaptive algorithm for multispecies gaseous flows on mapped domains. *Comput. Fluids*, 170:249–260, July 2018.

[64] L. D. Owen. *A Fourth-Order Solution-Adaptive Finite-Volume Algorithm for Compressible Reacting Flows on Mapped Domains*. Ph.D. dissertation, Colorado State University, 2019.

[65] P. McCorquodale, M. R. Dorr, J. A. F. Hittinger, and P. Colella. High-order finite-volume methods for hyperbolic conservation laws on mapped multiblock grids. *J. Comput. Phys.*, (288):181–195, May 2015.

[66] N. Overton, X. Gao, and S. Guzik. Applying high-order, adaptively-refined, finite-volume methods to discrete structured representations of arbitrary geometry. In *2020 AIAA SciTech Forum*, 2020.

[67] B. J. McBride, S. Gordon, and M. A. Reno. Coefficients for calculating thermodynamic and transport properties of individual species. Reference Publication 4513, NASA, 1993.

[68] 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.

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

[70] M. W. Chase. *NIST-JANAF Thermochemical Tables*, volume 9 of *J. Phys. Chem. Ref. Data*. American Inst. of Physics, 4 edition, 1998.

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

[72] X. Gao, L. D. Owen, and S. M. J. Guzik. A parallel adaptive numerical method with generalized curvilinear coordinate transformation for compressible Navier-Stokes equations. *Int. J. Numer. Meth. Fluids*, 82:664–688, 2016.

[73] D. J. Hill, C. P., and D. I. Pullin. Large-eddy simulation and multiscale modelling of a richtmyer–meshkov instability with reshock. *J. Fluid Mech.*, 557:29 – 61, 2006.

[74] W. Gao, W. Zhang, W. Cheng, and R. Samtaney. Wall-modelled large-eddy simulation of turbulent flow past airfoils. *J. Fluid Mech.*, 873:174–210, 2019.

[75] S. Walters, X. Gao, H. Johansen, and S. Guzik. Assessing stretched-vortex subgrid-scale models in finite volume methods for unbounded turbulent flows. *Flow Turbul. Combust.*, 2020. https://doi.org/10.1007/s10494-020-00206-1.

[76] P. Colella and M. D. Sekora. A limiter for PPM that preserves accuracy at smooth extrema. *J. Comput. Phys.*, 227(15):7069–7076, July 2008.

[77] P. Colella and P. Woodward. The piecewise parabolic method for gas-dynamical simulations. *J. Comput. Phys.*, 54:174–201, 1984.

[78] C. A. Kennedy and M. H. Carpenter. Additive Runge-Kutta schemes for convection-diffusion-reaction equations. Technical report, NASA Langley, 07 2001.

[79] M. R. Kristensen, J. B. Jørgensen, P. G. Thomsen, and S. B. Jørgensen. An ESDIRK method with sensitivity analysis capabilities. *Comput. Chem. Eng.*, 28(12):2695–2707, 2004.

[80] S. P. Nørsett and P. G. Thomsen. Local error control in SDIRK-methods. *BIT Numer. Math.*, 26:100–113, March 1986.

[81] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*. Springer-Verlag, Berlin Heidelberg, second edition, 1996.

[82] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, third edition, 1999.

[83] J. Dennis and R. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Society for Industrial and Applied Mathematics, 1996.

[84] M. S. Day and J. B. Bell. Numerical simulation of laminar reacting flows with complex chemistry. *Combust. Theory Model.*, 4:535–556, 2000.

[85] R. J. McDermott and J. E. Floyd. Enforcing realizability in explicit multi-component species transport. *Fire Saf. J.*, 78:180—187, November 2015.

[86] H. Lomax, T. H. Pulliam, and D. W. Zingg. *Fundamentals of Computational Fluid Dynamics*. Scientific Computation. Springer-Verlag Berlin Heidelberg, 1 edition, 2001.

[87] R. J. Kee, F. M. Rupley, J. A. Miller, M. E. Coltrin, J. F. Grcar, E. Meeks, H. K. Moffat, A. E. Lutz, G. Dixon-Lewis, M. D. Smooke, et al. Chemkin collection, release 3.6, reaction design. Technical Report SAND96-8216, Sandia National Laboratories, 2000.

[88] F. Perini, E. Galligani, and R. D. Reitz. An analytical Jacobian approach to sparse reaction kinetics for computationally efficient combustion modeling with large reaction mechanisms. *Energy Fuels*, 26(8):4804–4822, 2012.

[89] R. Xu, K. Wang, S. Banerjee, J. Shao, T. Parise, Y. Zhu, S. Wang, A. Movaghar, D. J. Lee, R. Zhao, X. Han, Y. Gao, T. Lu, K. Brezinsky, F. N. Egolfopoulos, D. F. Davidson, R. K. Hanson, C. T. Bowman, and H. Wang. A physics-based approach to modeling real-fuel combustion chemistry – ii. reaction kinetic models of jet and rocket fuels. *Combust. Flame*, 193:520–537, 2018.

[90] N. Zettervall, K. Nordin-Bates, E. J. K. Nilsson, and C. Fureby. Large eddy simulation of a premixed bluff body stabilized flame using global and skeletal reaction mechanisms. *Combust. Flame*, 179:1–22, 2017.

[91] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM J. Sci. Comput.*, 16:1190–1208, September 1995. Publisher: Society for Industrial and Applied Mathematics.

[92] G. Billet. Improvement of convective concentration fluxes in a one step reactive flow solver. *J. Comput. Phys.*, 204(1):319–352, 2005.

[93] G. Billet, V. Giovangigli, and G. de Gassowski. Impact of volume viscosity on a shock–hydrogen-bubble interaction. *Combust. Theory Model.*, 12(2):221–248, 2008.

[94] N. Attal, P. Ramaprabhu, J. Hossain, V. Karkhanis, M. Uddin, J. R. Gord, and S. Roy. Development and validation of a chemical reaction solver coupled to the FLASH code for combustion applications. *Comput. Fluids*, 107:59–76, 2015.

[95] P. A. T. Cocks, M. Soteriou, and V. Sankaran. Impact of numerics on the predictive capabilities of reacting flow LES. *Combust. Flame*, 162:3394–3411, 2015.

[96] S. J. Shanbhogue, S. Husain, and T. Lieuwen. Lean blowoff of bluff body stabilized flames: Scaling and dynamics. *Prog. Energy Combust. Sci.*, 35:98–120, 2009.

[97] C. Fugger, B. Paxton, J. R. Gord, B. A. Rankin, and A. W. Caswell. Measurements and analysis of flow-flame interactions in bluff-body-stabilized turbulent premixed propane-air flames. *2019 AIAA SciTech Forum*, pages 10.2514/6.2019–0733, 2019.

[98] R. Xu and H. Wang. A skeletal model for methane oxygen combustion in rocket engines. Stanford University, 2018.

[99] XBraid: Parallel time integration with multigrid. https://computation.llnl.gov/projects/parallel-time-integration-multigrid.

[100] M. J. Gander and M. Neumüller. Analysis of a new space-time parallel multigrid algorithm for parabolic problems. *SIAM J. Sci. Comput.*, 38(4):A2173–A2208, 2016.

[101] G. Horton and S. Vandewalle. A space-time multigrid method for parabolic partial differential equations. *SIAM J. Sci. Comput.*, 16(4):848–864, 1995.

[102] A. Brandt. Multi-level adaptive computations in fluid dynamics, 1979. Technical Report AIAA-79-1455, AIAA, Williamsburg, VA.

[103] W. L. Briggs, V. E. Henson, and S. F. McCormick. *A Multigrid Tutorial (2nd Ed.)*. Society for Industrial and Applied Mathematics, USA, 2000.

[104] J. Anderson. *Modern Compressible Flow: With Historical Perspective*. McGraw-Hill Education, Boston, 3 edition edition, July 2002.

[105] G. K. Batchelor. *An Introduction to Fluid Dynamics*. Cambridge University Press, Cambridge, February 2000.

[106] J. Christopher, X. Gao, S. M. Guzik, R. D. Falgout, and J. B. Schroder. Parallel in time for a fully space-time adaptive mesh refinement algorithm. In *AIAA Scitech 2020 Forum*, Orlando, FL, January 2020. American Institute of Aeronautics and Astronautics.

[107] R. D. Falgout, A. Katz, Tz. V. Kolev, J.B. Schroder, A. M. Wissink, and U.M. Yang. Parallel time integration with multigrid reduction for a compressible fluid dynamics application. Technical report, Lawrence Livermore National Lab, 2014.

[108] J. A. Domaradzki and N. A. Adams. Direct modelling of subgrid scales of turbulence in large eddy simulations. *J. Turbul.*, 3:N24, 2002.

[109] P. Sagaut. *Large Eddy Simulation for Incompressible Flows: An Introduction*. Scientific Computation. Springer-Verlag, Berlin Heidelberg, 3 edition, 2006.

[110] Eric Garnier, Nikolaus Adams, and P. Sagaut. *Large Eddy Simulation for Compressible Flows*. Scientific Computation. Springer Netherlands, 2009.

[111] S. Stolz and A. Adams. An approximate deconvolution procedure for large-eddy simulation. *Physics of Fluids*, 11(7):1699–1701, July 1999.

[112] Qing Wang and Matthias Ihme. A regularized deconvolution method for turbulent closure modeling in implicitly filtered large-eddy simulation. *Combustion and Flame*, 204:341–355, 2019.

[113] R. P. Fedkiw, B. Merriman, and S. Osher. High accuracy numerical methods for thermally perfect gas flows with chemistry. *J. Comput. Phys.*, 132:175–90, 1997.

# Appendix A

# Derivation of Analytic Reacting Source Jacobian

The analytic reacting source Jacobian is derived with the following considerations:

1. The mapped terms in the Jacobian are independent of this formulation.

2. The Jacobian is evaluated with cell averaged quantities, but for simplicity in expressions the cell-averaged notation, $\langle \cdot \rangle$, is dropped.

3. Since only the species are updated at each iteration of the nonlinear solver, only a subset of the source vector and state vectors need to be used. With $n = 1, \ldots, N_s$ for $N_s$ the number of species, the vectors are

   (a) Source vector $\hat{\mathbf{S}} = \{\rho \dot{\omega}_n\}$,

   (b) Primitive quantity vector $\hat{\mathbf{W}} = \{T, c_n\}$,

   (c) Conservative state vector $\hat{\mathbf{U}} = \{\rho c_n\}$.

From experimentation, it was discovered that density has a negligible impact on the reacting source Jacobian. First, the chain rule is applied for convenience, mathematically there is no difference

$$\frac{\partial \hat{\mathbf{S}}}{\partial \hat{\mathbf{U}}} = \frac{\partial \hat{\mathbf{S}}}{\partial \hat{\mathbf{W}}} \frac{\partial \hat{\mathbf{W}}}{\partial \hat{\mathbf{U}}}, \tag{A.1}$$

leading to two matrices that must be evaluated.

For evaluating the Jacobian $\partial \hat{\mathbf{W}} / \partial \hat{\mathbf{U}}$, either the process of Fedkiw et al. [113] or Gao [24] may be used. Due to the steps taken to reduce complexity, the matrix is not identical to that of either reference. The matrix in use is a $(1+N_s)\times N_s$ sized matrix

$$\frac{\partial\hat{\mathbf{W}}}{\partial\mathbf{U}} = \begin{bmatrix} \frac{(\gamma-1)(R_1T-h_1)}{\rho R} & \frac{(\gamma-1)(R_2T-h_2)}{\rho R} & \dots & \frac{(\gamma-1)(R_{\mathrm{N}_s}T-h_{\mathrm{N}_s})}{\rho R} \\ 1/\rho & 0 & \dots & 0 \\ 0 & 1/\rho & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1/\rho \end{bmatrix}.$$

The Jacobian $\partial\hat{\mathbf{S}}/\partial\hat{\mathbf{W}}$ follows much of the same procedure as Perini et al. [88], except pressure dependent reactions are neglected. This leads to a $N_s\times(1+N_s)$ sized matrix

$$\frac{\partial\hat{\mathbf{S}}}{\partial\mathbf{W}} = \begin{bmatrix} \partial\rho\dot{\omega}_1/\partial T & \partial\rho\dot{\omega}_1/\partial c_1 & \partial\rho\dot{\omega}_1/\partial c_2 & \dots & \partial\rho\dot{\omega}_1/\partial c_{\mathrm{N}_s} \\ \partial\rho\dot{\omega}_2/\partial T & \partial\rho\dot{\omega}_2/\partial c_1 & \partial\rho\dot{\omega}_2/\partial c_2 & \dots & \partial\rho\dot{\omega}_2/\partial c_{\mathrm{N}_s} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \partial\rho\dot{\omega}_{\mathrm{N}_s}/\partial T & \partial\rho\dot{\omega}_{\mathrm{N}_s}/\partial c_1 & \partial\rho\dot{\omega}_{\mathrm{N}_s}/\partial c_2 & \dots & \partial\rho\dot{\omega}_{\mathrm{N}_s}/\partial c_{\mathrm{N}_s} \end{bmatrix}.$$

The derivatives with respect to temperature are

$$\begin{aligned} \frac{\partial\rho\dot{\omega}_i}{\partial T} = M_i\sum_{r=1}^{N_r}\Bigg( & (\nu_{i,r}^{''}-\nu_{i,r}^{'})\left(\sum_{j=1}^{N_s}\alpha_{j,r}\left[\frac{\rho c_j}{M_j}\right]\right) \\ & \left(\frac{E_{a,r}}{R}A_rT^{\beta_r-2}\exp\left(-\frac{E_{a,r}}{RT}\right)\prod_{k=1}^{N_s}\left[\frac{\rho c_j}{M_j}\right]^{\nu_{k,r}^{'}}\right. \\ & + A_rT^{\beta_r-1}\exp\left(-\frac{E_{a,r}}{RT}\right)\beta_r\prod_{j=1}^{N_s}\left[\frac{\rho c_j}{M_j}\right]^{\nu_{j,r}^{'}} \\ & - \frac{E_{a,r}}{R\kappa_{eq}}A_rT^{\beta_r-2}\exp\left(-\frac{E_{a,r}}{RT}\right)\prod_{k=1}^{N_s}\left[\frac{\rho c_j}{M_j}\right]^{\nu_{k,r}^{''}} \\ & \left. - \frac{1}{\kappa_{eq}}A_rT^{\beta_r-1}\exp\left(-\frac{E_{a,r}}{RT}\right)\beta_r\prod_{j=1}^{N_s}\left[\frac{\rho c_j}{M_j}\right]^{\nu_{j,r}^{''}}\right)\Bigg), \end{aligned} \tag{A.2}$$

and the derivatives with respect to the mass fraction are

$$
\begin{aligned}
\frac{\partial \rho \dot{\omega}_i}{\partial c_n} = M_i \sum_{r=1}^{N_r} \Bigg( & (\nu_{i,r}^{''} - \nu_{i,r}^{'}) \\
& \Bigg[ \left( \sum_{j=1}^{N_s} \alpha_{j,r} \left[ \frac{\rho c_j}{M_j} \right] \right) \Bigg( \nu_{n,r}^{'} \kappa_{f,r} \frac{\rho}{M_n} \left( \frac{\rho c_n}{M_n} \right)^{-1} \prod_{j=1}^{N_s} \left[ \frac{\rho c_j}{M_j} \right]^{\nu_{j,r}^{'}} \\
& - \nu_{n,r}^{''} \kappa_{b,r} \frac{\rho}{M_n} \left( \frac{\rho c_n}{M_n} \right)^{-1} \prod_{j=1}^{N_s} \left[ \frac{\rho c_j}{M_j} \right]^{\nu_{j,r}^{''}} \Bigg) \\
& + \left( \alpha_{n,r} \frac{\rho}{M_n} \right) \left( \kappa_{f,r} \prod_{j=1}^{N_s} \left[ \frac{\rho c_j}{M_j} \right]^{\nu_{j,r}^{'}} - \kappa_{b,r} \prod_{j=1}^{N_s} \left[ \frac{\rho c_j}{M_j} \right]^{\nu_{j,r}^{''}} \right) \Bigg] \Bigg) .
\end{aligned}
\tag{A.3}
$$

For reactions that do not involve a third body-term, the following terms simplify to,

$$\sum_{j=1}^{N_s} \alpha_{j,r} \left[ \frac{\rho c_j}{M_j} \right] = 1 \, , \tag{A.4}$$

$$\sum_{j=1}^{N_s} \alpha_{j,r} \left[ \frac{c_j}{M_j} \right] = 0 \, , \tag{A.5}$$

$$\alpha_{n,r} \frac{\rho}{M_n} = 0 \, . \tag{A.6}$$