PARALLEL WAVELET-ADAPTIVE DIRECT NUMERICAL SIMULATION OF MULTIPHASE FLOWS WITH PHASE-CHANGE

A Thesis
Presented to
The Academic Faculty

by

Christopher J. Forster

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
Woodruff School of Mechanical Engineering

Georgia Institute of Technology
August 2016

PARALLEL WAVELET-ADAPTIVE DIRECT NUMERICAL SIMULATION OF MULTIPHASE FLOWS WITH PHASE-CHANGE

Approved by:

Professor Marc K. Smith, Advisor
Woodruff School of Mechanical Engineering
Georgia Institute of Technology

Roman Grigoriev
School of Physics
Georgia Institute of Technology

Ari Glezer
Woodruff School of Mechanical Engineering
Georgia Institute of Technology

Richard Vuduc
School of Computational Science and Engineering
Georgia Institute of Technology

Cyrus K. Aidun
Woodruff School of Mechanical Engineering
Georgia Institute of Technology

Jonathan Regele
School of Aerospace Engineering
Iowa State University

Date Approved: 1 August 2016

To my parents

ACKNOWLEDGEMENTS

The completion of this work was made possible by support from the U.S. Department of Energy, Office of Science, Office of Workforce Development for Teachers and Scientists, Office of Science Graduate Student Research (SCGSR) program that allowed me to collaborate with the Combustion Research Facility at Sandia National Laboratories in Livermore, CA. Partial support was provided by the National Institute of Justice (Award Number 2013-DN-BX-K003).

I would like to thank my thesis advisor, Dr. Smith, who has taught me invaluable lessons and encouraged me over the years. I would like to thank Guilhem Lacaze for his support and being my mentor at Sandia National Laboratories. I would also like to thank Joe Oefelein, Marco Arienti, Anthony Ruiz, Layal Hakim, Francois Doisneau, and Zach Vane from Sandia National Laboratories for their support and insightful conversations. I would like to thank my thesis committee, Dr. Aidun, Dr. Glezer, Dr. Grigoriev, Dr. Vuduc, and Dr. Regele for their time and feedback. I would also like to thank Jonathan Regele, Oleg Vasilyev, and Joe Powers for many insightful and thought-provoking conversations. The path to my Ph.D. has taken many interesting turns and allowed me to meet many encouraging friends and colleagues, all of whom I am grateful to know and look forward to interacting with throughout my career.

I would like to thank my parents, younger sister, and younger brother for supporting and encouraging me throughout this entire process. Finally, I would like to thank my soon-to-be wife, Kirsten Jackson, for always being there for me and keeping my spirits up.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

High-powered and high-energy density electronics are becoming more common with advances in computing, electric vehicles, and modern defense systems. Applications like these require efficient, compact, and economical heat exchanger designs capable of extremely large heat fluxes. Phase-change cooling methods allow for these characteristics; however, the design and optimization of these devices is extremely challenging. Numerical simulations can assist in this effort by providing details of the flow that are inaccessible to experimental measurements. One such system of interest to this work is acoustically enhanced nucleate boiling, which is capable of dramatic increases in the Critical Heat Flux (CHF). The focus of the present work is the development of a numerical simulation capable of predicting the behavior of acoustically enhanced nucleate boiling up to the CHF.

A general-purpose wavelet-adaptive Direct Numerical Simulation (DNS) that runs entirely on the Graphics Processing Unit (GPU) architecture has been developed in this work to allow accurate, error-controlled simulation of a wide range of applications with multiphase flow at all Mach numbers. This work focuses on the development of a high-order simulation framework that can adequately address the challenges posed by acoustically enhanced nucleate boiling processes. Nucleate boiling in the presence of acoustic fields suffers from a large disparity of important time scales, namely the acoustic time scale and the convective time scale near the incompressible limit. To address this issue, the compressible Navier-Stokes equations are solved using a preconditioned dual-time stepping method to allow for accurate simulation of the flow for all Mach numbers, everywhere in the domain. The governing equations are solved on a wavelet-adaptive grid that provides a direct measure of local error and

is adapted at every time step to follow the evolution of the flow for a significant reduction in computational resources and expense. The use of the wavelet-adaptive grid and the dual-time stepping method together allows for rigorous error control in both space and time. All components of this simulation have been redesigned and optimized for efficient implementation on the GPU architecture to offset the overhead of grid adaptation and further reduce time-to-solution. The development of the high-performance, error-controlled computational framework and its verification and validation is presented.

CHAPTER I

INTRODUCTION

High-powered and high-energy density electronics are becoming more common with advances in computing, electric vehicles, and modern defense systems. Applications like these require efficient, compact, and economical heat exchanger designs that allow for extremely large heat fluxes. Air cooling and single-phase liquid cooling have limitations in heat flux capacity due to size constraints, natural convection, and/or pumping losses, with heat flux limits substantially lower than what is possible with phase-change (*i.e.*, boiling and/or condensation) processes. Phase change provides a significant improvement to the overall heat transfer of a heat exchanger by taking advantage of cooling fluids with a large latent heat of vaporization to remove a large amount of heat at an approximately constant temperature. A comparison of the various cooling methods is presented in Table 1. Phase-change cooling devices, without optimization, can provide stable cooling for heat fluxes of approximately 115 W/cm^2 with a wall superheat of approximately 15 K, resulting in a heat transfer coefficient of approximately 75,000 W/m^{2}K. While this is a substantial improvement over air and liquid cooling, there is demand for even larger heat flux removal from emerging electronic devices while reducing power requirements [Capozzoli and Primiceri (2015) [18]]. One of the main limitations of phase-change cooling technologies is the Critical Heat Flux (CHF) which is the peak heat flux before the device transitions from nucleate boiling to film boiling. This transition in the boiling process is often catastrophic, resulting in severe overheating, since the resulting vapor film acts as a thermal insulator, and there is hysteresis in attempting to return to the preferred nucleate boiling state. While the CHF limitation cannot be removed completely, heat

exchanger designs can be optimized to maximize the CHF to allow for a much wider range of safe operation. Improvements to heat exchanger design for increased CHF has been observed experimentally but many questions remain unanswered about the fundamental mechanisms that allow for these improvements and how to use them to further optimize heat exchanger design [Dhir (2006) [32], Cooke and Kandlikar (2011) [21], Douglas et al. (2012) [34]].

The remainder of this introduction is intended as a high-level overview of the motivations and direction of this work.

Table 1: Various cooling methods with their typical heat transfer coefficients [Incropera et al. (1996) [52], Boziuk et al. (2010) [13]]

Cooling Method	h (W/m^{2}K)
Natural Convection - Air	2-25
Forced Convection - Air	25-250
Natural Convection - Liquid Water	50-1,000
Forced Convection - Liquid Water	100-20,000
Nucleate Boiling - Water	2,500-100,000
Enhanced Nucleate Boiling (μ-channels+acoustics) - Water	$\sim 220,000$

The mechanistic prediction of nucleate boiling has been pursued through the development of empirical correlations since the mid-20^{th} century. These often are only valid over small operating ranges and require knowledge about the nucleation site distribution on a particular working surface, which limits the reliability and usefulness of the correlation as a predictive tool [Dhir (2006) [32]]. There are many physical processes occurring in nucleate boiling that make it difficult to build a model that captures the full range of behavior in this system. These processes include, but are not limited to, transient conduction in the solid and fluid, evaporation at the bubble

base on the solid (*i.e.*, microlayer evaporation), evaporation/condensation on the interface of the bubble, thermocapillary convection, bubble-motion-induced convection, bubble departure size and frequency, and nucleation site activation that depends on heat transfer in both the fluid and solid heater. These processes span a large range of spatial and time scales that makes it difficult, if not impossible, to measure them without disturbing the flow due to the physical size of measurement devices.

There exist a number of methods for enhancing the heat transfer characteristics of a device, including textured or micro-machined surfaces that increase surface area and influence the hydrodynamics, surface coatings that affect surface tension and contact angles, nano/micro-porous structures, and acoustic forcing of the liquid-vapor interface [Das et al. (2009) [23], Boziuk et al. (2010) [13], El-Genk and Ali (2010) [37]]. These enhancement methods can be used individually or in conjunction to increase the heat transfer capability of a particular device, but their use adds further complexity to the system and the resulting non-linear interactions are not currently well understood.

Acoustic forcing of nucleate boiling is of particular interest to the present work. One way to obtain an understanding of this complicated system with its intricate coupling of many physical processes is through the use of a numerical simulation. Thus, the main focus of this work is to develop a numerical simulation of acoustically forced nucleate boiling with a sufficient level of accuracy and real-time performance that can be used to gain insight into the many complicated physical processes involved in this enhanced boiling system.

The status quo of the boiling research community was assessed by Dhir (2006) [32], and in addition to noting the limitations of the existing empirical models, Dhir offered insight (and hope) into how Direct Numerical Simulation (DNS) can be used to further progress and develop improved boiling models. Dhir also commented on the assumption of many models that sub-processes act independently, which ignores

the non-linear interaction dynamics of multiple nucleation sites and bubbles. This can be addressed more readily in numerical simulation than in experiment since the nucleation sites can be specified directly by the simulation to study their interdependence.

Much of the existing work in the DNS of nucleate boiling, *e.g.*, [Son and Dhir (1999) [104], Juric and Tryggvason (1998) [58]], solves the incompressible Navier-Stokes equations and uses a Boussinesq approximation for the buoyancy forces in natural convection. These assumptions may be violated or become inaccurate when there is excessive wall super-heating or sub-cooling of the ambient working fluid, when pressure variations created by channel pressure drops or surface tension in small bubbles create density variations in the vapor phase, and/or when acoustic forcing is applied. The incompressible assumption does not allow for acoustic waves in the solution, nor does it capture the bulk mode of oscillation in bubbles. There have been some studies using the weakly compressible form of the Navier-Stokes equations [Paolucci (1982) [84]], which improves the situation by capturing the bulk mode of bubble oscillation and eliminates the risk of violating the Boussinesq approximation; however, it is not sufficient for the simulation of nucleate boiling with acoustic enhancement since the weakly compressible form does not capture acoustics. The fully compressible form of the governing equations permits acoustic waves in the solution, and it allows for the use of advanced equations of state for increased fidelity. At the present time, there are no known studies of nucleate boiling using the fully compressible form of the Navier-Stokes equations. The present work will fill this gap in knowledge by producing a numerical simulation of a fully compressible flow system. This simulation will reduce the assumptions in the underlying equations and produce a result that is more faithful to the underlying physics of the system.

The choice of the numerical methods and their implementation in the simulation

are affected by the physics of the system, the desired accuracy, and the type of computing hardware that the simulation will be run on. To be effective, a simulation needs to faithfully reproduce the physics through the proper choice of governing equations, choice of discretization techniques, and the efficient implementation of discretization methods on a given computing architecture. Historically, efficient implementation simply required optimization for the minimum number of operations when computers were single-threaded; however, modern computing architectures such as the Central Processing Unit (CPU), many-core (*e.g.*, Xeon Phi), and Graphics Processing Unit (GPU) architectures operate with O(10), O(10^2), and O(10^5) concurrent threads, respectively. Depending on the choice of computing architecture, different choices of numerical methods are needed to take advantage of the resources of the dramatically different underlying architectures to obtain maximum performance. A particular numerical method often has order-of-operation dependencies that may limit the amount of parallelism that can be exposed, and in the case with many threads, it may be best to optimize not just for the minimum number of operations but also for the maximum number of concurrent threads to reduce the overall time-to-solution. This parallelized numerical method should still solve the governing equations with sufficient accuracy for the end application. This requires a more coupled approach to simulation development than in years past. It requires knowledge of the problem from the physics, applied mathematics, and computer science perspectives in order to create a high-fidelity simulation that runs efficiently on these modern computing architectures. It is crucial that equal emphasis be placed on each of these three disciplines since a shortcoming in any single area can affect the overall performance of the simulation.

With the choice of equations to describe the underlying physics decided with acoustics in mind, the method of discretization of the equations for numerical solution needs to be specified. The underlying physics are important in this choice since the processes involved span a large range of length and time scales, which can be expensive

to resolve numerically. The disparity of scales in the boiling process suggests the use of adaptive methods to reduce computational expense. In this work, a wavelet-adaptive finite-difference method is chosen for the work savings from the reduced grid point count relative to a uniform grid with the finest resolution, and for the data locality that finite differences offer for computational performance. The grid is updated at every time step to evolve with the solution features, and the wavelet-adaptive method provides a mechanism for rigorous local error control. The GPU architecture is chosen for its potentially large performance advantage over the CPU architecture, even with multi-threading and vectorization. At the beginning of this work, the many-core architecture (*e.g.*, Xeon Phi) was not yet readily available, but it may be another viable option for future work. A major challenge in this work was to implement the wavelet transform and finite differences on a sparse grid in a way that is efficient on the GPU architecture.

The compressible Navier-Stokes equations have numerical problems at low Mach numbers due to the pressure singularity problem [Tannehill et al. (2012) [88]]. Stiffness arises in low-speed flows due to the disparity in the eigenvalues (*i.e.*, the acoustic speed vs. the convective speed) as the Mach number approaches zero. This results in an ill-conditioned system that is not only increasingly expensive to solve, but one in which the pressure and density become decoupled. To circumvent this problem, a dual-time stepping method with preconditioning is used to correct for the pressure singularity and remove the stiffness in the coupled set of equations. The challenge with this dual-time stepping method is that it must be implemented to work effectively with the wavelet-adaptive method and to run efficiently on the GPU.

In the present simulation framework for multiphase flows with phase-change, the multiphase flow characteristics of the problem, *i.e.*, the interface tracking, normal vector, and interface curvature calculations, will be performed using a level-set method.

The approach taken in the present work is to define a diffuse interface of user-prescribed thickness that the adaptive-grid fully resolves throughout the simulation. With the level of refinement possible with the adaptive grid, the interface thickness is set to be much smaller than the bubble diameter. This thin but fully resolved interface allows for a high order of accuracy to be maintained everywhere in the domain. The surface tension forces on the interface are implemented through a continuum surface force method that uses an approximate delta function of commensurate order of accuracy with the discretization of the governing equations to maintain the overall order of accuracy of the simulation.

The choice of wavelet-adaptive grids, dual-time stepping, and preconditioning with the compressible form of the Navier-Stokes equations, and the diffuse interface model with appropriate order-of-accuracy to match the rest of the simulation are meant to work together to provide an error-controlled framework for the simulation of multiphase flow with phase change and surface tension. The wavelet-adaptive grid enforces local error tolerances, the dual-time stepping procedure provides a mechanism for controlling the relative tolerance of the residuals of each equation to improve the control of error in time, and the diffuse interface with appropriate discretization maintains the overall accuracy of the scheme for multiphase flows with surface tension. Each of these methods was also chosen with the GPU architecture in mind. They were efficiently implemented in a simulation that runs completely on a single GPU with high performance in both memory use and execution time.

This document is organized in a way that follows the development process for the general purpose simulation that will handle a variety of single- and multi-phase flows at all Mach numbers, including nucleate boiling with acoustic forcing. The motivation and choice of methods has been outlined above, and the literature review in Chapter 2 will further reinforce the need for high-fidelity simulation to gain a better understanding of the fundamental mechanisms in nucleate boiling. In Chapter 3, the

simulation development is documented by first introducing the wavelet-adaptive grid on the GPU for high-speed compressible flows (*i.e.*, no preconditioning or level-set included). Next, Chapter 4 introduces the dual-time stepping method with preconditioning for computation of low-Mach number flows and covers the additional development that was needed to allow the dual-time stepping method and preconditioning to be implemented on wavelet-adaptive grids and to perform efficiently on the GPU. The final development stage in Chapter 5 introduces the level-set method and realistic equations of state for multiphase flow. At the end in Chapter 6, final remarks are made about the overall performance and applicability of this approach for simulation of for general multiphase flows and nucleate boiling with and without acoustic enhancement.

CHAPTER II

LITERATURE REVIEW

A brief survey of the tumultuous history of boiling and associated hydrodynamic theory is provided here to give some perspective on the current state of boiling research. An exhaustive history of boiling, covering the years 1732–1985, is presented by Lienhard and Witte (1985) [70]. The first widely credited study of boiling dates back to 1756, where Leidenfrost placed drops of water into a heated spoon at varying temperatures and measured the time to evaporation. He noticed that spoon temperatures with small superheating values caused the droplets to evaporate in seconds, but at very high temperatures (*i.e.*, large superheating values) the droplets would dance around and take more than a minute to evaporate. This is attributed to the large temperature difference between the spoon and the droplet saturation temperature that allows a vapor film to be created and maintained between the metal and liquid, which acts as a thermal insulator that reduces the heat transfer to the liquid drop compared to the higher heat transfer caused by convection of the liquid in direct contact with the metal. Additionally, the vapor has lower viscosity than the liquid which reduces friction and allows for the dancing-like motion. The support of a droplet on its own vapor is now called the *Leidenfrost effect* and corresponds to points on the boiling curve at or above the *Leidenfrost Point*, which is also the minimum heat flux point on the boiling curve that would be discovered later by Nukiyama [Nukiyama (1934) [80]] (see Figure 1). This effect was first observed by Boerhaave in 1732, but the detailed investigation was done by Leidenfrost (1756) [67].

In 1934, Nukiyama (1934) [80, 100] recreated the Leidenfrost experiment with a thin horizontal wire submerged in water. The wire acted as the heater and a

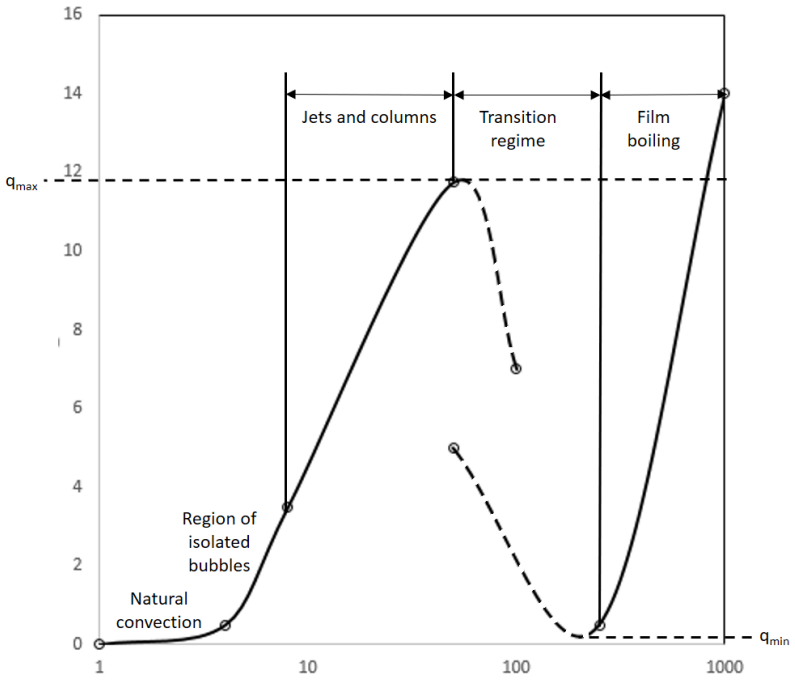

Figure 1: The boiling curve with hysteresis is shown as a function of wall superheat. The upper segment of the curve is the nucleate boiling regime, and the lower segment of the curve is the film boiling regime. The transition region with abrupt changes between nucleate and film boiling is indicated with dashed lines, and it follows different paths depending on if it is approached from the nucleate or film boiling side.

resistance thermometer simultaneously, which provides both the heat flux and wall superheat. His results found a large hysteresis loop depending on the direction of travel along the boiling curve. The transition from the peak heat flux to film boiling was accompanied by a sudden temperature jump of roughly 1000 K. If the heat flux was reduced gradually to return to nucleate boiling, the wall superheat dropped suddenly below a certain point. This work is typically considered the first modern quantitative treatment of boiling [Lienhard and Witte (1985) [70]] and resulted in the development of a boiling curve that is now common in undergraduate heat transfer texts [Incropera et al. (1996) [52]], as shown in Figure 2.

At very low wall superheats on the boiling curve (see Figure 1), near the saturation temperature of the liquid, there is little or no bubble formation and the heat transfer is dominated by natural convection. As the heat flux is further increased, the wall

superheat increases and bubbles begin to nucleate. Initially, nucleate sites appear across the surface but become more active and interact more as the wall becomes more superheated. As the peak heat flux is approached, the jets and vapor columns start to impede the inflow of fresh liquid to the wall and intermittent patches of film boiling begin to appear. This is called transition boiling. If the wall super heat is further increased, past the CHF, there is a sudden transition to complete film boiling along the heater surface. To recover from this transition, the wall superheat must be decreased until the vapor film suddenly collapses and the flow returns to nucleate boiling. Applications with two-phase heat exchangers typically attempt to avoid the transition from nucleate to film boiling since it is accompanied by a large jump in surface temperature due to the vapor film acting as a thermal insulator.

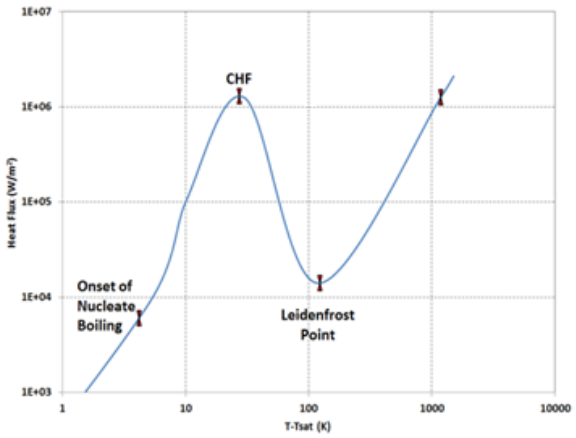

Figure 2: Typical continuous boiling curve for water at atmospheric pressure. The heat flux is shown as a function of wall superheat.

The transition region of the boiling curve has been very difficult to study due to the tendency to transition quickly and the resulting difficulty in controlling the power input to the electric heater used in experiments [Lienhard and Witte (1985) [70]]. There are concerns about the accuracy of the shape of the boiling curve that is commonly illustrated in textbooks [Incropera et al. (1996) [52]] and similarly illustrated in Figure 2. Nukiyama had hypothesized that if he were able to truly control the heat flux and temperature independently, it may be possible to follow

the transition region in experiment [Nukiyama (1934) [80, 100], Lienhard and Witte (1985) [70]]. This was thought to be partially confirmed by Drew and Mueller (1937) [35] who more accurately controlled the temperature of the heated surface and thus were able to measure a few points along the transition region. These events appear to have set the classical image of the boiling curve with a continuous transition region in the minds of many researchers [Lienhard and Witte (1985) [70]]. Doubts about Drew and Mueller's experiment to truly independently control the wall temperature have been cast by [Lienhard and Witte (1985) [70]]. Drew and Mueller were not the only investigators to find evidence for what they thought should exist. Farber and Scorah (1948) [39, 70] claimed to have followed the continuous transition through "artful manipulation of the electric supply - something that no investigator has subsequently been able to do" [Lienhard and Witte (1985) [70]].

This first successful experiment to map out the transition region was performed by Sarkurai and Shiotsu (1974) [94] using a closed-loop control system to maintain the heater temperature in a much more precise and accurate way. This eliminated the "artfulness" of the approach and offered more confidence in the repeatability of the experiment to the boiling community. They found something much different than previously reported. Their results showed a significant hysteresis loop in the boiling curve and not a continuous path, as illustrated in Figure 1. They were not able to trace a complete path through the transition region even with their more advanced heater temperature control system. Additionally, they pointed out that the lower segment illustrated in Figure 1 is really a locus of averages of film and transition boiling occurring over the surface of the heated wire. Lienhard and Witte (1985) [70] attribute the misconception of a continuous transition region to Drew and Mueller's work, which in turn caused many other researchers to search for and incorrectly find evidence of the continuous transition. Lienhard and Witte explain how this misconception persisted for so long, while acknowledging their advantage

of 30 years of hindsight, by pointing to the work of Kuhn (1970) [63] that found historical precedent in the pattern of researchers repeatedly seeing what they expect to see in experiments based on preconceptions.

In search of a better understanding of the physics of boiling, Bonilla and Perry (1941) [10, 70] made the first observation that the escape of vapor from a heater surface will impede the inflow of liquid and compared it to the flooding (*i.e.*, forcing gas up through a liquid) of a distillation column. In the 1940's, Kutateladze, a Russian engineer, began studying burnout in boiling flows and drew similar conclusions about the hydrodynamic behavior of flooding. This appears to be the precursor to the work of Zuber and Tribus (1959) [131], which also drew from the findings of Chang (1957) [19], and which noted that there appeared to be a correlation between the shape of an interface in film boiling and the thermo-convective instability based on the wavelengths predicted and observed in film boiling. The combination of working with Chang (with his insights about the hydrodynamic stability of boiling flows) and Zuber's fluency in Russian (that allowed him access to Kutateladze's work) appears to have been the spark that inspired Zuber's doctoral research [Lienhard and Witte (1985) [70]]. Additionally, there were advances in the analysis of interfacial stability around that time (post-WWII), and Liendhard and Witte indicate that the convergence of all of these events motivated Zuber's PhD thesis on the "Hydrodynamic Aspects of Boiling Heat Transfer". This is the first major work on the hydrodynamic stability of boiling, which produced a theoretical prediction of CHF.

Zuber and Tribus identified three hydrodynamic interfacial instabilities in boiling; the surface-tension driven instability of a cylindrical interface, the gravitationally driven instability of a liquid over a vapor (*i.e.*, Rayleigh-Taylor instability), and the shear-flow instability of liquid and vapor moving in parallel opposite directions (*i.e.*, Kelvin-Helmholtz instability). Zuber's ambitious thesis (1959) [131] covered the entire boiling regime with chapters: 1) "A Review of Nucleate Boiling," 2) "The Problem of

Bubble Growth," 3) "Hydrodynamic Aspects of Nucleate Boiling," 4) "Hydrodynamic Aspects of Transitional Boiling," 5) "The Minimum Heat Flux Density in Transitional Boiling From a Horizontal Surface," and 6) "The Critical Heat Flux in Boiling From a Horizontal Surface." An important aspect of Zuber's work is that he not only wanted to correlate data but *predict* it too. Although much of Zuber's work was later proven piece-by-piece to not be completely correct by subsequent researchers, it did set the foundation for much of the correlations and mechanistic models that followed. It is still worth mentioning some of the highlights from his large body of work.

In Zuber's third chapter, he analyzes the problem of vapor removal from the heater surface. In analogy to a submerged orifice in a distillation column, he identifies stages in the progression from low-heat flux nucleate boiling to the approach to the CHF and transition. At very low vapor generation rates, he considers bubble formation to be a problem of hydrostatics where the bubble diameter is independent of the vapor generation rate. It is assumed that after each bubble pinch-off the liquid near the nucleation site comes to rest, thus each new bubble departure initially looks the same as the previous (uninfluenced by previous states). As the vapor generation rate increases, bubble pairs are formed from a nucleation site and they touch during ascent, with the first assuming a hemispherical shape and the second vertically elongated. Further increases in vapor generation rate cause the coalescence to take place closer to the nucleation site, until the coalescence occurs directly at the nucleation site. An additional increase in vapor generation rate causes a pair of already coalesced bubbles to coalesce (*i.e.*, double coalescence), eventually leading to a continuous turbulent vapor jet. The hypothesis is that the vapor jets over a heater surface become unstable and collapse, causing burnout [Zuber (1959) [131]]. Jumping ahead to Zuber's sixth chapter, a prediction of maximum heat flux was developed by assuming that the nature of CHF is governed by the Rayleigh-Taylor and Kelvin-Helmholtz instabilities. Zuber noted that the phenomenon of transitional boiling "bears similarity to a release

of bubbles with variable frequency from a set of regularly spaced orifices of fixed geometry" and that this is due to Rayleigh-Taylor instability. Using this observation, he assumes the vapor columns near CHF have diameter $\lambda_0/2$ and spacing λ_0, where λ_0 is the critical wavelength of the Rayleigh-Taylor instability. The Kelvin-Helmholtz stability of the cylindrical jet is analyzed to find when it will collapse, and Zuber formed an expression for the heat flux density at CHF based on this finding. As the final part of the transition, when two vapor jets begin to interfere with each other, the flow of liquid toward the surface is interrupted and a vapor patch (*i.e.*, patch of transitional boiling) is formed. This progresses to a fully developed stable film as the heat flux is further increased.

Lienhard and Witte counted eight basic problems with Zuber's analysis [Lienhard and Witte (1985) [70]]. These will not be recounted here, except to say that a number of incorrect assumptions were made. There was a divide between the many supporters and skeptics of Zuber's hydrodynamic theory for many years. A large variability in the experimental data was present because not all of the important factors, such as heater size and gravity (acceleration), had been accounted for by the boiling research community. This made it difficult to determine who was right. Berenson (1960) [5] did a considerable amount of work in resolving the problems found in Zuber's thesis work (1959) [131]. He was able to show that Zuber's prediction of the maximum heat flux at CHF was correct for a sufficiently large heater, *i.e.*, an infinite heater. This was an assumption in Zuber's original analysis, but it was not made clear in his thesis.

Lienhard and Witte comment on Zuber's work and give due credit by citing Kuhn's analysis of scientific progress by quoting "that good ideas are seldom completely correct in their original presentations." Zuber's work shed new light on the problem of boiling and others began to think about it in new ways that allowed progress to be made. The above discussion also gets to the root of the motivation for the present

work, which is to develop a high-fidelity numerical simulation that makes a *minimum number of assumptions* about the flow. This will allow accurate numerical experiments to be performed, provide valuable data currently inaccessible to experiments, help to verify (or not) current boiling models, and potentially guide the development of new, improved boiling models.

The present work is not the first numerical investigation of boiling. One of the earliest is the numerical simulation of bubble evolution over a heated surface by Lee and Nydahl (1989) [66]. They performed an axisymmetric simulation that used an elliptic grid generation technique to map a Cartesian computational grid onto the complex computational domain, which consisted of the working fluid (pool), the bubble void (excluded from the computational domain), and the microlayer region underneath the bubble. Using this simulation they were able to determine significant heat flux contributions attributed to different physical mechanisms and their percentage contribution to the overall heat flux. Their findings indicate that the microlayer contributes roughly 87% of the enhanced heat transfer (over convection alone), microconvection (*i.e.*, enhanced convection due to bubble growth and motion) is negligible, and the heat transfer following bubble departure accounts for approximately 13% of the enhanced heat transfer. The analysis was performed for water over a flat plate at atmospheric pressure and 8.5 K wall superheat. Several assumptions and simplifications were made in their work to make the problem more tractable, including: 1) constant properties, 2) bubble thermodynamic equilibrium, 3) negligible evaporative resistance, 4) an isothermal wall, 5) negligible free convection, 6) no-slip velocity boundary condition around the perimeter of the bubble, and 7) specified bubble and microlayer shape. The microlayer shape was prescribed using the formulation from Cooper and Lloyd (1969) [22].

In an improved simulation, Welch (1998) [124] directly simulated the liquid and vapor flow with conjugate heat transfer in the solid heater. The simulation was

implemented using a finite volume scheme with front tracking and a moving mesh. The work was presented as progress towards a full simulation of the development of a microlayer and concluded that the numerical tool developed could "eventually facilitate a deeper understanding of nucleate bubble growth as the hydrodynamics and the bubble shape can be obtained as part of the solution." Unfortunately, it appears that there is no continuation of this work in the literature.

Juric and Tryggvason (1998) [58] developed an explicit front tracking method to simulate film boiling. The explicit front tracking method generates a mesh on the interfacial boundary between two fluid domains that are coupled through the use of interfacial boundary conditions. This provides a sharp interface to implement accurate boundary conditions and allows for very good mass conservation. They compared the numerical results to experimental correlations and found reasonable agreement. They concluded with mention of future plans for making some minor modifications to allow the technique to be applied to nucleate boiling and to the transition region of the boiling curve.

The first complete numerical simulation of bubble growth was performed by Son et al. (1999) [104]. In contrast to Welch (1998) [124], Son et al. use a fixed Cartesian grid with a level-set method to implicitly track the interface. Additionally, Son et al. does not include the solid heater in the computational domain, replacing it with a fixed wall temperature boundary condition. The microlayer between the bubble and the heater surface is not directly resolved. Instead, a lubrication model is used to approximate its behavior and influence on the macro-region of the fluid (*i.e.*, the microlayer model is solved based on inputs from the simulation and feeds back to the simulation through a boundary condition specified near the interface). This model captures the large contribution of the microlayer to the overall heat flux while reducing the computational expense since the thin microlayer region does not require additional grid points. With wall superheats between 7–8 K, they determined that the heat flux

contribution from the microlayer is approximately 20%, which is much different than the 87% found by Lee and Nydahl [66].

Tryggvason et al. (2008) [114] reported on simulations based on the future plans in Juric and Tryggvason (1998) [58]. In this brief work, they simulated the evolution and detachment of vapor bubbles from prescribed nucleation sites on a flat surface. Example solutions were presented to demonstrate the simulation capabilities without verification. The paper was part of a conference proceeding that provided an update on their progress and plans to parallelize the solver for larger numbers of nucleation sites to study nonlinear interactions between nucleation sites.

More recent work by Dhir et al. (2012) [31] performed micro-gravity experiments on the International Space Station and compared the data with numerical results from the simulation developed in Son et al. (1999) [104] run with matching conditions. In these experiments perfluoro-n-hexane was used as the working fluid under pressures ranging from 51–243 kPa. Under micro-gravity conditions the bubbles remain attached to the flat heater surface with a spherical cap shape, and there is essentially no natural convection. The experiments applied very low heat fluxes between 0.01–1.3 W/cm^2 over long bubble growth periods on the order of 100 s. In these experiments, they showed comparisons of the experimentally determined heat flux and the numerical results as a function of radius from the center of the bubble contact patch. Inside of the bubble (vapor) region, they indicate a heat flux of 4×10^{-4} W/cm^2. A peak heat flux of 574 W/cm^2 is shown at the contact line in the numerical solution, but there is no comparison with the experiment due to the limited resolution in the experimental setup. In the liquid region, there is roughly a factor of two difference between the experimental and numerical results. They mention the uncertainty in heat flux measurement ranges from 26% to 1.6% over the heat flux input range of 0.01–1.3 W/cm^2. The prediction of heat flux over the heater surface appears to be inconclusive inside the bubble with significant differences in the liquid

region; however, the simulation results appear to capture the spherical cap bubble shape and volume well.

Dhir et al. (2013) [30] provides a review of the progress in numerical simulation of nucleate boiling with an emphasis on his work with collaborators. Most of the simulations cited in Dhir et al. (2013) [30] use a fixed wall temperature (*i.e.*, prescribed superheat), with the exception of Aktinol and Dhir (2012) [2] that includes the thermal response of the solid. These simulations are in general qualitative agreement in the evolution of the bubble size and shape throughout the growth and pinch-off phases. Dhir's simulation of saturated film boiling on a horizontal cylinder gives Nusselt number results within approximately 10% of experimental correlations by Sakurai et al. (1990) [93]. In the future, they plan to perform full three-dimensional simulations of nucleate boiling. Their previous attempts at 3D simulation were severely limited due "to the enormous computing power required and the corresponding memory requirements." They estimate approximately 10^9 grid points for a cubical domain with a side length of 2.5 cm with water at standard atmospheric pressure and gravity. With this estimate of the problem size for a full 3D simulation, they specifically state that serial computations are insufficient and that parallel computations would be required to solve this problem in a reasonable amount of time.

CHAPTER III

HIGH-MACH NUMBER COMPRESSIBLE FLOW

3.1 Introduction

The high heat flux capability of sub-cooled nucleate boiling makes it an attractive method for cooling in many applications, such as high-speed microprocessors, power generation systems, and the motion control systems of hybrid electric vehicles, for example. One limit on the performance of this boiling process is the Critical Heat Flux (CHF) limit, which is an upper bound on the heat flux for any particular system. Above this limit, nucleate boiling undergoes a transition to film boiling where a thin insulating vapor layer appears between the heated surface and the cooling fluid, which then leads to a catastrophic increase in the temperature of the heated surface. Many passive and active control methods have been proposed with the goal of increasing the CHF by delaying the transition to film boiling, thus enabling safe operation of the system at significantly higher heat fluxes. Passive methods include textured boiling surfaces, such as open microchannels etched into the heated surface. Active methods use forced convection, jets, sprays, acoustic forcing, and/or a variety of external forcing methods to increase the heat transfer coefficient and CHF. In electronics cooling applications, passive flow control is of particular interest since it requires less energy, provides improved reliability compared to active flow control methods, and requires no user input during operation. While there are many experimental studies that quantify the effects of these boiling enhancement techniques, there are still many unanswered questions regarding the fundamental physics that enable the improved boiling characteristics. This is mainly due to difficulties in obtaining experimental measurements without disrupting the flow and with the extremely small length scales

of critical flow features, namely the thin microlayer region near the contact line of a vapor bubble on the heated surface. Additionally, many of the mechanistic boiling models that have been proposed in attempts to explain the CHF transition do not account for all of the physical processes involved, or treat them as independent processes, when they actually form a highly coupled non-linear system. Thus, there is a need for high-fidelity numerical simulations closely coupled with detailed experiments to provide additional insight into the boiling process. This could then lead to improved mechanistic models for the prediction of boiling behavior over a larger parameter regime than current models [Dhir (2006) [32]].

Numerical simulations of nucleate boiling are extremely challenged by the time-dependent, three-dimensional, multiscale nature of the thermal-fluid processes involved. Length scales range up to six orders of magnitude from the mesoscale associated with the heater itself down to the microscale associated with the sharp, localized property variations across the vapor-liquid interface of the vapor bubbles. These bubbles can also move in a time dependent, almost chaotic way throughout large portions of the domain during the course of a simulation, and their location is typically not known before the start of the simulation. Thus, a successful boiling simulation must have extremely fast execution times and be capable of very fine, localized spatial resolution. Fast execution times require parallel computation, while fine spatial resolution requires large uniform grids, large unstructured grids, or time-dependent, spatially adaptive grids. A structured, time-dependent, adaptive grid is preferred for several reasons: it reduces overall memory requirements, maintains predictable memory access patterns, and significantly reduces the number of required computations while still maintaining Single Instruction, Multiple Data (SIMD) parallelism. All of these features contribute to an improvement in the total execution time of the simulation.

The multiphase flow simulation presented in this work uses a wavelet-adaptive grid with finite differences to achieve the required spatial resolution along with strict

error control on the solution. A wavelet-adaptive grid is capable of resolving length scales spanning six orders of magnitude and can realize grid point reduction (compression) ratios up to $O(10^5)$ for solutions with highly localized structures compared to a uniform grid with a resolution equal to the finest wavelet scale present. Traditional Adaptive Mesh Refinement (AMR) techniques implement *ad hoc* indicators to identify regions that require refinement. The error these indicators introduce to the solution is not well-defined, and so *a posteriori* approaches to estimate the error are required. In addition, ill-behaved mesh refinement in the vicinity of a fluid-fluid interface can cause instability problems for interfacial flows. These problems could be unphysical forcing of the instability by perturbations in the solution due to grid changes near the interface, or the refinement criteria not requiring sufficient resolution, and/or numerical diffusion to prevent the onset of an unphysical interfacial instability. Wavelets alleviate these problems through the use of Lagrange interpolating polynomials that provide a well-defined measure of local error. In addition, relating wavelets to the finite difference formulation of the derivatives provides a single parameter (tolerance) that controls the point-wise magnitude of the error in both the values and derivatives of the solution. This tolerance is then used to determine the local level of grid refinement. The wavelet-adaptive grid is updated at every time step in the solution to ensure that the error tolerance and the correct level of grid refinement is always maintained. The result is that these wavelet simulations can be performed with *a priori* control over the error in the solution rather than specifying a grid and determining the error *a posteriori*.

In place of the typical grid independence study needed for structured or unstructured grids, a wavelet-adaptive grid only requires a test of the adapted solution's sensitivity to the user-specified wavelet tolerance. This reduces the number of runs needed to provide accurate results and increases overall confidence in the solution.

The sensitivity study on the user-prescribed wavelet tolerance is necessary to determine the system's global sensitivity to the controlled local error.

Wavelet-based methods were initially used to solve PDEs with a Galerkin approach [Bertoluzza et al. (1994) [6]]. Wavelets and their companion scaling functions were used as basis functions to expand the solution and the problem was rewritten in terms of the expansion coefficients. Here, derivatives were performed using the wavelet and scaling basis functions directly and the number of grid points used was reduced with wavelet coefficient thresholding. While successful, this approach has difficulties in treating non-linear terms (*e.g.*, real fluid properties) and irregular boundaries, although several techniques have been developed to circumvent these problems [Schneider and Vasilyev (2010) [96]]. Wavelet collocation methods were introduced to more easily treat non-linear terms and finite domains. The collocation method operates on the function values at each grid point rather than on the wavelet coefficients (*i.e.*, they solve in physical space rather than wavelet space), and this results in a more efficient method that requires fewer operations per grid point [Vasilyev and Bowman (2000) [119]]. The most recent wavelet methods use wavelets to adapt the grid and finite differences on the adapted grid to evaluate derivatives using physical values. The evaluation of finite difference stencils on a wavelet-adapted grid is much less expensive than performing derivatives using the wavelet basis functions directly, largely due to serial data dependencies in the wavelet operations. The combination of wavelet-adapted grids and standard finite-difference techniques applied in physical space allows for an optimum sparse grid representation of the solution (*i.e.*, maximum grid compression for a given error tolerance) together with increased computational efficiency [Schneider and Vasilyev (2010) [96]].

The performance of the wavelet-adaptive method is significantly enhanced if it is implemented using High-Performance Computing (HPC) hardware. The path forward for HPC, at least until the end of the decade [Ahern et al. (2011) [1], Wells

(2015) [125]], relies on the Graphics Processing Unit (GPU) and many-core processor architectures. GPUs currently offer the highest peak theoretical double-precision floating point performance and memory bandwidth [AMD (2016) [3], Intel (2016) [53], NVIDIA (2016) [81]]. For these reasons, the present wavelet-adaptive multiphase flow simulation was designed and implemented from the ground up to run entirely on a GPU. Note that this design avoids all of the communication bottlenecks in data transfer that may occur between the CPU host and the GPU.

This is the first of three chapters that present the development of a complete direct numerical simulation of nucleate boiling with acoustic forcing. This GPU-based multiphase flow simulation uses the compressible form of the governing equations, a finite difference formulation for the derivatives, and a wavelet-adaptive grid to achieve the required spatial resolution with strict error control on the solution. The motivation behind this approach is to faithfully capture the system's essential physics while minimizing the number of assumptions and artificial treatments, such as incompressibility, constant fluid properties, and explicit source terms at the liquid-vapor interface that enforce conservation of mass, momentum, and energy. This multiphase simulation is a robust solver for flows at all Mach numbers. It handles incompressible flows by solving the compressible form of the Navier-Stokes equations with a preconditioned dual-time stepping method. The simulation also implements a fully resolved, diffuse, fluid-fluid interface with a user-defined thickness. It uses high-order finite differences to maintain accuracy near these interfaces, thereby eliminating the need to apply *ad-hoc* corrections for conserved quantities at the interface.

This chapter is organized as follows. The governing equations are outlined in § 3.2 and details of the wavelet method are described in § 3.3. The mathematical verification of the wavelet and derivative calculations is presented in § 3.4, along with three verification test cases for fully compressible flows. Finally, the parallel performance of the wavelet finite-difference method on the GPU is discussed in § 3.5

and conclusions are given in § 3.6.

Chapter 4 of this thesis will describe the use of a preconditioned, dual-time stepping method to compute transient, non-isothermal, single-phase flows approaching the incompressible limit. The efficacy of this technique combined with a wavelet-adaptive grid on a GPU will be fully explored. Chapter 5 will introduce the level-set method to model fluid-fluid interfaces and the implementation of general equations of state to produce a full nucleate boiling simulation with acoustic forcing. The performance of this simulation and its accuracy compared to experimental results will be fully documented.

3.2 Governing Equations

In these simulations, the physics for the evolution of a compressible, viscous flow is governed by the following set of conservation equations for mass, momentum, energy, and species transport.

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_j}{\partial x_j} = 0 \tag{1}$$

$$\frac{\partial \rho u_i}{\partial t} + \frac{\partial \rho u_i u_j}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j} + \rho g_i \tag{2}$$

$$\frac{\partial \rho E}{\partial t} + \frac{\partial}{\partial x_j}[(\rho E + p)u_j] = -\frac{\partial q_j}{\partial x_j} + \frac{\partial u_i \tau_{ij}}{\partial x_j} + \rho u_i g_i \tag{3}$$

$$\frac{\partial \rho Y_k}{\partial t} + \frac{\partial \rho u_j Y_k}{\partial x_j} = -\frac{\partial J_{k,j}}{\partial x_j} \tag{4}$$

Here, ρ is the density, u_i is the velocity vector, p is the pressure, τ_{ij} is the viscous stress tensor, g_i is the acceleration of gravity vector, E is the total specific energy defined as the sum of the specific internal and kinetic energy ($E = e + \frac{1}{2}u_i u_i$), and q_j is the heat flux. In the last equation for species transport, Y_k is the mass fraction for the k^{th} species and J_k is the diffusive flux of species Y_k. The sum of the mass

fractions of all species must be identically equal to one. There are two methods to ensure this [Poinsot and Veyante (2005) [89]]. The first is to explicitly track the density (Equation 1) and $N-1$ species, where the final component is calculated as $Y_N = 1 - \sum_{m=1}^{N-1} \rho_m Y_m$. This lumps the accumulation of rounding and other errors into the last (N^{th}) species, which is often chosen as an inert component (*e.g.*, nitrogen) to minimize the effect of this error accumulation. The second method is to introduce correction velocities to ensure not only the global conservation of mass but the mass of each of the species. The first method is sufficient for the demonstration cases in this chapter; however, for different conditions, it may become necessary to include correction velocities.

These conservation equations are augmented by a set of constitutive equations for the viscous stress, heat flux, and mass diffusion. The results in this work employ a Newtonian fluid model and Fick's law for diffusion, although other models can be implemented. Thermodynamic properties will use an equation of state appropriate to the material. For example, air will be modeled as a calorically perfect ideal gas.

3.3 The Wavelet-Adaptive Method

3.3.1 Background

The ultimate goal of a functional representation of a given data set is to represent the data with as few bits or points as possible. Wavelets are one way to accomplish this. They date back to the early 1900's with the introduction of the Haar wavelet (a localized step function) [Haar (1910) [50]]. Wavelets and their associated transforms represent a natural progression from the earlier Fourier and sliding-window Fourier transforms. A Fourier transform provides frequency information about a signal or function, but no correlation to its temporal location. The sliding-window extension provides information about both time and frequency; however, the window length places limits on the frequency resolution, necessitating a trade-off between frequency

and spatial resolution. Wavelets offer improved resolution in both the time and frequency domains. The wavelet transform utilizes specific basis functions that are dilated and shifted to multiple levels of resolution on a signal, an image, or a solution to a PDE. The principle idea of the transform is to find a correlation between the data points so that a smaller number of points can be used to reconstruct the original data using the wavelet basis functions, either exactly or to within a specified accuracy. Typically, the data reduction is achieved if the data points in the set are well correlated. The transform is called "lossy" if the reconstruction thresholds the data and replaces "small" coefficients with zeros. This recreates the data with a small error that is less than a specified tolerance.

Wavelets have a long history in signal and image processing [Daubechies (1996) [26], Mallat (2008) [73]], and an analogy can be drawn between the pixels of an image and the points of a grid used to solve a PDE. Much like JPEG2000 wavelet compression [Skodras et al. (2001) [102]], a wavelet transform is used to compress solution data on an adaptive grid, although different wavelet basis functions more suitable to solving PDEs than image compression are used. This was demonstrated in the prior work of [Vasilyev and Bowman (2000) [119], Paolucci et al. (2014) [85], Holmstrom (1999) [51], Griebel and Koster (2000) [49]] who showed that the solution to a finite-difference, time-dependent simulation on a wavelet-adaptive grid can be successfully and automatically compressed as the solution evolves in time. This essentially moves points from smooth regions to regions with sharper gradients where more resolution is required, thus allowing the use of larger simulation domains and/or the resolution of finer details within a fixed memory budget.

The wavelet method used in this work is developed on a dyadic grid, which inherently provides multiple levels of resolution. The base (coarsest) grid is labeled J_0 and each successive grid level doubles the resolution of the previous level up to a maximum level J_{max}. To provide context for the wavelet decomposition of a function

(or a solution to a PDE) on this dyadic grid, a set of function spaces $V_j \in L^2(\mathbb{R})$ are defined where $j \in \mathbb{N}$ [Daubechies (1992) [25]]. This set of function spaces is nested according to the relation

$$V^{J_0} \subset V^{J_0+1} \subset \cdots \subset V^j \subset \cdots \subset V^{J_{max}-1} \subset V^{J_{max}}. \tag{5}$$

In addition, each of the spaces V_{j+1} comprises two complementary subspaces V_j and W_j, *i.e.*,

$$V_{j+1} = V_j \oplus W_j. \tag{6}$$

The basis functions for the space V_j are the scaling functions ϕ_j while the wavelet functions ψ_j are the basis functions for the complementary space W_j. In first-generation wavelet methods, the scaling and wavelet functions at each level of resolution are created through successive translations and dilations of "mother" functions and the Fourier transform is used to develop the associated wavelet transforms. As a result, first-generation wavelets require periodic or infinite domains. Second-generation wavelets [Sweldens (1998) [107], Vasilyev and Bowman (2000) [119], Wirasaet (2007) [126]] alleviate this finite-domain restriction by allowing the use of modified scaling and wavelet functions that are not based on simple translations and dilations of a mother function. The construction methods for second-generation wavelets are done entirely in the spatial domain and so complex domains, non-uniform dyadic grids, and boundaries can be handled and a wide range of wavelet bases can be used.

One such construction method for second-generation wavelets is partly based on the work of [Deslauriers and Dubuc (1989) [29]] who were the first to define a recursive procedure for interpolating function values onto a dyadic grid, a process called the interpolating subdivision scheme. The left side of Figure 3 illustrates this recursive refinement procedure using 4^{th}-order Lagrange interpolating polynomials. The right side of Figure 3 introduces and defines the concept of a *detail*, which is the difference

between the original and interpolated (predicted) function value. The detail is an essential element of the forward and inverse wavelet transforms that are discussed in § 3.3.2. This recursive refinement procedure generates what Deslauriers and Dubuc called the *fundamental solution*, but which is now more generally known as the *scaling function* [Sweldens (1998) [107]]. The interpolating subdivision scheme has a strong link to modern wavelet methods and is discussed in detail by [Donoho and Yu (1999) [33]].

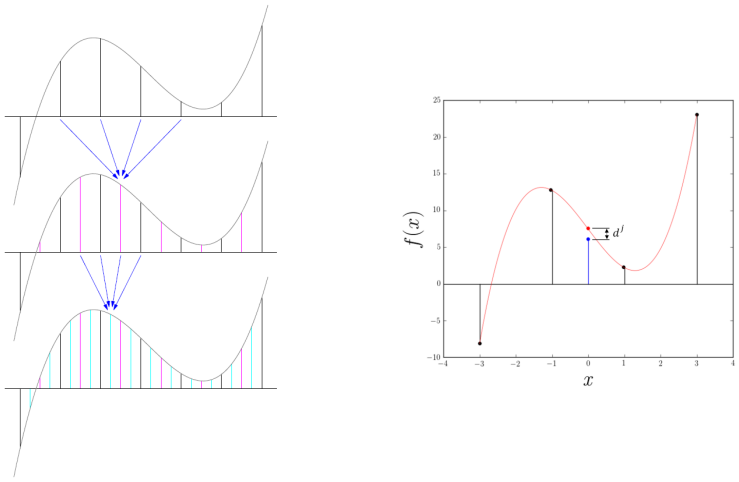

Figure 3: Left: An example of the interpolating subdivision scheme on a dyadic grid using a cubic interpolating polynomial. Right: The definition of the detail. The original coarse values are shown in black, the cubic interpolating polynomial and the predicted value is shown in red, and the original value at the finer resolution is shown in blue. The detail is defined by the difference between the original and predicted values.

Solving PDEs on finite domains is the primary focus of the present work and so second-generation wavelets along with the lifted wavelet construction process will be used. The lifted wavelet construction process has two steps and builds on the Deslauriers-Dubuc interpolating wavelets and scaling functions that are, in turn, the autocorrelation of the Daubechies scaling functions [Mallat (2008) [73]].

The first step, called the *predict* step, interpolates from a coarse grid to the next finer grid using Lagrange interpolating polynomials of order p as shown in Figure 3 and computes the wavelet coefficients (the details) for the coarse grid points. The second step, called the *update* step, interpolates the details using the grid points on the finer grid back down to the coarse grid and computes new coefficients for the scaling functions at the coarse grid. This second step, also called the *lifting* step, creates a wavelet of order p with p vanishing moments (see the example in Figure 4).

The vanishing moments in the wavelet prevent a loss of information when coarsening the grid (*e.g.*, removing every other point) by preserving the mean, variance, and higher moments of the solution [Sweldens (1998) [107]]. This reduces the possibility of significant aliasing effects that could lead to incorrect solutions [Vasilyev and Bowman (2000) [119]].

Although the scaling and wavelet basis functions are crucial to the development and functional analysis of the wavelet method, they are never computed directly in the wavelet construction process outlined above. Instead, given the function values on the finest grid, the *forward wavelet transform* computes all of the scaling and wavelet coefficients through successive applications of Lagrange interpolating polynomials starting at the grid level just below the finest grid. This is particularly convenient since the dyadic structure of the grid allows the interpolation weights to be precomputed to reduce the number of operations at run time [Sweldens and Schroder (2000) [109]]. Similarly, the *inverse wavelet transform* computes all of the function values on the finest grid starting with the full set of scaling and wavelet coefficients. For reference, Figure 4 shows an explicit representation of the 4^{th}-order scaling and wavelet functions. Both of these functions have compact support, which is an important property that allows for efficient parallel computation of the forward and inverse wavelet transforms.

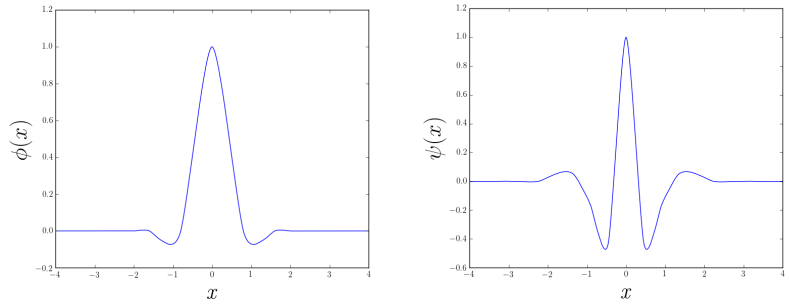

Figure 4: The 4^{th}-order scaling function (left) and wavelet function (right) computed from a single nonzero coefficient using lifted, 4^{th}-order Lagrange interpolating polynomials in the inverse wavelet transform.

3.3.2 Mathematical Description

The second-generation wavelets used in the current work are constructed using lifted, p^{th}-order Lagrange interpolating polynomials. The following summary of the construction process closely follows the work and notation of Sweldens (1998) [107] and Vasilyev and Bowman (2000) [119].

A given function $f(x)$ is expanded in a wavelet multiresolution representation to a finite level of resolution using a set of scaling and wavelet basis functions as follows:

$$f^{J_{max}}(x) = \sum_{k \in \kappa} c_k^0 \phi_k^0(x) + \sum_{j=0}^{J_{max}-1} \sum_{k \in \kappa} d_k^j \psi_k^j(x), \tag{7}$$

where κ is the set of grid points used to represent the function.

Note that in a dyadic grid, the set of points at each finer level contain all the points from coarser levels in addition to its own. An important aspect for notation is that every other point is a point that also belongs to coarser levels with the points in between existing at the current level. More formally, this is described as

$$x_k^j = x_{2k}^{j+1} \tag{8}$$

$$f(x_k^j) = f(x_{2k}^{j+1}). \tag{9}$$

Equations 8 and 9 simply describe the nestedness of the grid levels, and the relationship and indexing of even and odd points and function values on adjacent levels of

resolution.

Using this notation, the scaling coefficients at the maximum level of resolution are given by

$$c_k^{J_{max}} = f(x_k^{J_{max}}). \tag{10}$$

The fast wavelet transform is performed using the following two equations in a recursive manner down to the base level J_0

$$d_k^j = c_{2k+1}^{j+1} - \sum_l w_{k,l}^j c_{2k+2l}^{j+1} \tag{11}$$

$$c_k^j = c_{2k}^{j+1} + \sum_l \tilde{w}_{k,l}^j d_{k+l}^j \tag{12}$$

where $w_{k,l}^j$ are the interpolation coefficients for the scaling functions, $\tilde{w}_{k,l}^j$ are the interpolation coefficients for the wavelet functions, and l is the local interpolation index.

Similarly, the inverse transform is performed recursively up from the base level using

$$c_{2k}^{j+1} = c_k^j - \sum_l \tilde{w}_{k,l}^j d_{k+l}^j \tag{13}$$

$$c_{2k+1}^{j+1} = d_k^j + \sum_l w_{k,l}^j c_{2k+2l}^{j+1} \tag{14}$$

with the end result for the function values given by

$$f(x_k^{J_{max}}) = c_k^{J_{max}}. \tag{15}$$

The critical step in adapting the grid and compressing the data is thresholding. Here, any wavelet coefficient less than the user-specified wavelet tolerance ϵ is set to zero. After this thresholding process, the resulting approximation function denoted with the $\geq$ subscript is represented as

$$f_{\geq}^{J_{max}}(x) = \sum_{k\in\kappa} c_k^0 \phi_k^0(x) + \sum_{j=0}^{J_{max}-1} \sum_{k\in\kappa} d_k^j \psi_k^j(x). \tag{16}$$

Thresholding can be performed on wavelet coefficients using either absolute or relative tolerances. In many cases, where the governing equations are solved in dimensionless form, an absolute comparison is sufficient. If a disparity in the magnitude of function values exists, then a relative tolerance is preferred.

For more details, comprehensive discussions on the use of wavelet methods for the solution of PDEs can be found in the work of [Holmstrom (1999) [51], Vasilyev and Bowman (2000) [119], and Paolucci et al. (2014) [85]]; and more generally, the development of the wavelet method is discussed by [Daubechies (1992) [25], Sweldens (1998) [107], and Mallat (2008) [73]].

3.3.3 Implementation

Up until this point the discussion has been limited to wavelet transforms in one dimension. Wavelets can be constructed in multiple dimensions, called non-separable wavelets, or constructed from tensor products of one-dimensional scaling functions and wavelets, called separable scaling functions and wavelets. The non-separable wavelet method generally provide better reconstruction of the solution [Tymczak et al. (2002) [115]], especially in directions diagonal to the grid, but separable wavelet methods require fewer points in cache (*i.e.*, shared and local memory) at any given time, and thus give higher performance on current computing hardware. The work presented here exclusively uses separable wavelets. For clarity, 4^{th}-order wavelets will be used to demonstrate the algorithm, but the idea extends to wavelets of higher order.

A dataset of dimension d represented with separable wavelets uses $2^d - 1$ wavelet families, (*i.e.*, three wavelet families in 2D and seven in 3D). The wavelet families resulting from the tensor product of scaling and wavelet functions are listed in Table 1 [Nejadmalayeri et al. (2015) [79], Vasilyev (2003) [118]]. In 2D, Figure 5 illustrates the data dependencies of the direction-by-direction computation in the tensor product

wavelet transform, which is important for efficient implementation of the transform.

Table 2: Wavelet families in 2D and 3D.

$$\psi_{i,k}^{\mu,j}(\mathbf{x}) = \begin{cases} \psi_i^j(x)\phi_k^j(y), & \mu = 1 \\ \phi_i^j(x)\psi_k^j(y), & \mu = 2 \\ \psi_i^j(x)\psi_k^j(y), & \mu = 3 \end{cases} \qquad \psi_{i,k,m}^{\mu,j}(\mathbf{x}) = \begin{cases} \psi_i^j(x)\phi_k^j(y)\phi_m^j(z), & \mu = 1 \\ \psi_i^j(x)\phi_k^j(y)\psi_m^j(z), & \mu = 2 \\ \psi_i^j(x)\psi_k^j(y)\phi_m^j(z), & \mu = 3 \\ \phi_i^j(x)\psi_k^j(y)\psi_m^j(z), & \mu = 4 \\ \phi_i^j(x)\phi_k^j(y)\psi_m^j(z), & \mu = 5 \\ \phi_i^j(x)\psi_k^j(y)\phi_m^j(z), & \mu = 6 \\ \psi_i^j(x)\psi_k^j(y)\psi_m^j(z), & \mu = 7 \end{cases}$$

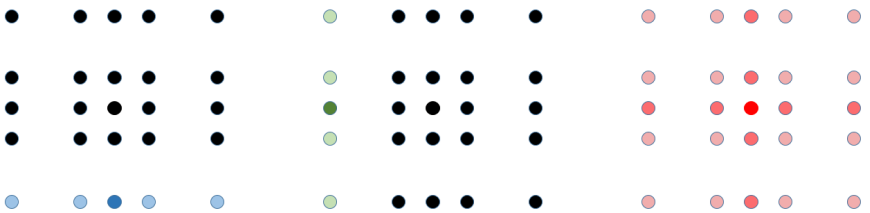

Figure 5: Wavelet families on a 2D grid are shown with the point belonging to each family shaded in a darker color and its dependencies drawn in a lighter shade of the same color. Individual Family 1, Family 2, and Family 3 points are shown from left-to-right. Rows and columns containing no data dependencies for any of the three highlighted points have been omitted for clarity.

The higher dimensional fast wavelet transforms are performed by applying the 1D Lagrange interpolations in a sequence over the number of dimensions to form the multi-dimensional wavelets from a tensor product of 1D scaling and wavelet functions. In performing the two-dimensional fast wavelet transform at a particular grid level, every row and column are operated on to calculate the wavelet coefficients for Family 1, Family 2, and Family 3 points. Family 1 and Family 2 are aligned with the x- and y-axes, respectively, so they behave the same way as in the 1D wavelet transform; however, Family 3 is different in that it requires data from the coarser level and

the same level of resolution. After the transform is performed over each direction, the same sequence is applied on the next grid level. The forward transform starts the operation at the finest level and works towards the coarsest level. The inverse transform is applied in a similar manner to the forward transform, except that the operations are done in the exact reverse order starting from the coarsest grid to the finest. The serial pseudo code for the forward and inverse transforms is shown in Algorithms 1 and 2.

Algorithm 1 Forward Wavelet Transform

1: **for** each level $j = J_{max} - 1 : -1 : J_0$ **do**
2: **for** each direction n = 1:d **do**
3: *Predict*: $d_k^j = c_{2k+1}^{j+1} - \sum_l w_{k,l}^j c_{2k+2l}^{j+1}$
4: *Update*: $c_k^j = c_{2k}^{j+1} + \sum_l \tilde{w}_{k,l}^j d_{k+l}^j$
5: **end for**
6: **end for**

Algorithm 2 Inverse Wavelet Transform

1: **for** each level $j = J_0 : 1 : J_{max} - 1$ **do**
2: **for** each direction n = d:-1:1 **do**
3: *Undo Update*: $c_{2k}^{j+1} = c_k^j - \sum_l \tilde{w}_{k,l}^j d_{k+l}^j$
4: *Undo Predict*: $c_{2k+1}^{j+1} = d_k^j + \sum_l w_{k,l}^j c_{2k+2l}^{j+1}$
5: **end for**
6: **end for**

The pseudo code in Algorithms 1 and 2 ignores the complexity of parallel implementation. However, the forward and inverse wavelet transforms are readily parallelized in a vectorized (SIMD) manner for all points existing on level j and requiring operations in direction n. One important decision in the design of the wavelet transform algorithms for parallel operation on a GPU is data organization for higher-order wavelets in higher dimensions. This can take the form of one-dimensional or multi-dimensional tiling of the data in memory. GPU shared memory and register allocation sizes are small relative to the large number of concurrent threads, and they perform

best when the interior data region is large compared to the halo region on the boundary of the data. To illustrate this, consider a wavelet with a stencil width of six points, which requires three halo points on each direction. A single Family 7 point in 3D requires a 6x6x6 tile (216 points), which is a significant portion of the shared memory available while maintaining full occupancy on current GPU architectures [NVIDIA (2016) [81]]. For a 3x3x3 block of interior points (27 points), a 9x9x9 tile (729 total points) is required in shared memory, which uses nearly all of the available shared memory while maintaining a block size of 256 and 100% occupancy. The fraction of points belonging to the interior is 0.037, which indicates poor data reuse and shared memory performance. In contrast, a 1D tile containing 250 interior points and 6 halo points has an interior point fraction of 0.977. The much larger interior point fraction for 1D tiling versus multi-dimensional tiling indicates that 1D tiling would give better GPU shared memory performance. The 1D tile approach also complements the tensor product construction of the multi-dimensional wavelets used in this work. Thus, the GPU-parallelized algorithms were designed using the 1D tile shared memory optimization.

The dynamically evolving wavelet-adaptive grid used during the time integration of a set of PDEs is implemented using the forward wavelet transform and a sequence of deletion and inclusion steps. Given a set of existing points with specified function values at all levels of resolution, the forward wavelet transform calculates the wavelet coefficients for all points. The thresholding step marks for deletion all those points whose wavelet coefficient is less than the user-prescribed wavelet tolerance ϵ. It is important to note that these points are merely *marked* for deletion and not actually deleted at this point. Any point may potentially be added back to the set in the next two steps of the algorithm. The points retained after the thresholding step together with the points at the coarsest grid level are called the set of *essential points*.

Next, a zone of *buffer points* is populated around all of the essential points in

order to allow the solution of the PDEs to evolve and/or sharpen over a single time step. This is the only part of the algorithm that allows for grid refinement. The buffer zone width is chosen based on how much the solution can evolve in one time step, which is typically limited by the Courant-Friedrichs-Lewy (CFL) number of unity in time-accurate transient simulations. For this reason, the buffer zone is created by considering each essential point and including immediately adjacent points on the same level and one level finer in all directions [Vasilyev and Bowman (2000) [119]]. Note that if the wavelet coefficient on a buffer point does not become significant after a time integration step, it may simply be deleted in the next threshold operation.

The PDEs are solved on each of the essential and buffer points. This typically requires the computation of various spatial derivatives and in this work finite differences are used. To provide support for these calculations, *derivative points* are inserted into the grid at each level of resolution, if they do not already exist. These points are initialized with zero wavelet coefficients. It is important to note that when physical values are computed for these points with the inverse wavelet transform, the values are interpolated to within the specified wavelet tolerance ϵ.

The operations used to select the essential, buffer, and derivative points are purely local and so the resulting set of points may not include all of the points necessary for wavelet interpolation. The next step is to add in all of the necessary *interpolation support points* for all of the point sets in a recursive identification procedure starting from the highest level. This procedure is called the *reconstruction check* [Vasilyev and Bowman (2000) [119]]. Once done, any remaining essential points still marked for deletion are actually deleted from the set. The inverse wavelet transform is then applied to the final set of points to recover the physical values on the wavelet-adaptive grid.

The PDEs of interest to this work are the governing equations described in Equations 1-4. These equations are solved using the method of lines by computing the

right-hand-side of the governing equations on each of the essential and buffer points. All derivatives are computed using a centered finite difference scheme, with biased stencils near the boundaries, and with a user-defined order. The resulting system of ordinary differential equations is shown in Equation 17 with obvious notation.

$$\frac{\partial \boldsymbol{u}}{\partial t} = \boldsymbol{G}(t, \boldsymbol{u}) \tag{17}$$

These equations are integrated in time for a high-speed compressible flow problem using either the explicit second- and third-order Total Variation Diminishing (TVD) Runge-Kutta schemes from Gottlieb and Shu (1998) [48] or the low-storage Runge-Kutta schemes from Kennedy et al. (2000) [60].

The overall time-dependent wavelet-adaptive solver algorithm is outlined in Algorithm 3. The grid is adapted after every global time step, but remains fixed during the advancement of Equation 17. When using the Runge-Kutta integration schemes, the grid is held fixed for all intermediate stages in a global step.

Algorithm 3 Time-dependent wavelet-adaptive solver outline.

```
1: Given ϵ and t_final
2: for m=0 to t_final do
3:     Solve Equation 17 with a suitable time integration scheme
4:     Perform forward wavelet transform
5:     Threshold wavelet coefficients
6:     Insert buffer zone points
7:     Insert derivative points
8:     Perform reconstruction check to ensure interpolation support on all point sets
9:     Delete remaining thresholded points
10:    Perform inverse wavelet transform
11: end for
```

3.3.4 Grid Compression

The first characteristic of the wavelet-adaptive grid that will be demonstrated is grid compression. Figure 6 shows the result of applying the 1D wavelet transform on an approximate Heaviside step function. The approximation is a hyperbolic tangent

function defined on the domain $x = [0, 1]$ with a finite transition region. A wavelet tolerance of $\epsilon = 10^{-4}$ was used in the transform. The red grid points indicate the essential points retained by the wavelet transform. The black grid points are the excluded points in a uniform grid at each level. In this test case, 69 essential points were used in the wavelet grid to represent this function compared to a uniform grid at the highest level of 8,193 points, for a compression ratio of 118.

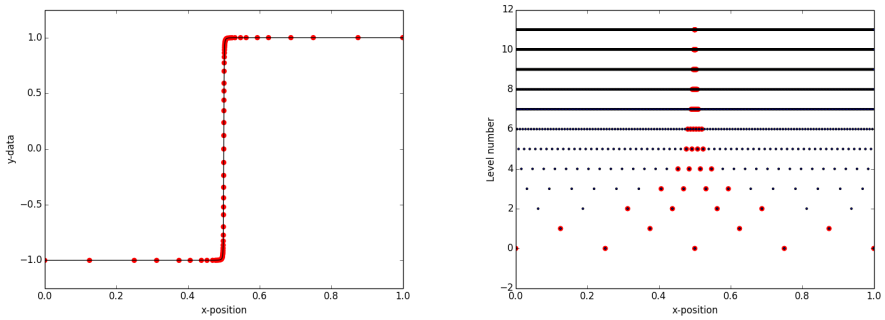

Figure 6: A one-dimensional hyperbolic tangent function with its associated wavelet adaptive grid. The wavelet tolerance is $\epsilon = 10^{-4}$.

To visually demonstrate the large span of length scales resolvable with the wavelet method, a 2D multiscale test function is defined in Equation 18 to contain significant features on length scales varying by six orders of magnitude. The wavelet representation of the function (wavelet tolerance $\epsilon = 10^{-7}$) is shown in Figure 7. The smaller-scale features of the result are made visible by zooming in on individual peaks that have the same base function defined on a smaller length scale and centered on the peak. The wavelet representation contains significant wavelet coefficients on length scales varying by six orders of magnitude. There are approximately 3.3 million total points in the wavelet grid compared to a uniform grid of $983,041^2$ points, for a compression ratio of $O(10^5)$.

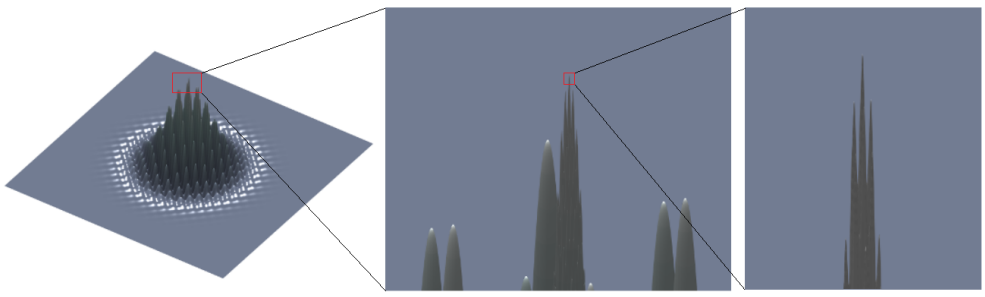

Figure 7: The 2D multiscale demonstration function used to demonstrate the ability of the wavelet-adaptive grid to capture length scales varying by six orders of magnitude.

$$
\begin{aligned}
f(x,y) = \cos(40\pi x)\cos(40\pi y)\exp(-64\sqrt{x^2+y^2}) + \\
0.05\cos(1200\pi x)\cos(1200\pi y)\exp(-25000\sqrt{x^2+y^2}) + \\
0.002\cos(30000\pi x)\cos(30000\pi y)\exp(-1500000\sqrt{x^2+y^2})
\end{aligned}
\tag{18}
$$

To demonstrate the compression capabilities in 3D, the example function described by Equation 18 is extended to a third dimension, as shown in Equation 19. Typically, a function is visualized in three spatial dimensions by mapping the function value to a color at each point. In this case, however, the ray-casting technique of Meyer-Spradow et al. (2009) [75] is used to more effectively highlight the structure of the solution, as shown in Figure 8. This technique highlights regions of specified gradient magnitudes, which produces an effect similar to iso-surfaces located in regions with larger gradients. The relationship between this visualization technique and the one used in Figure 7 is illustrated in Figure 9. Due to the size of the 3D solution and memory limitations, symmetry is used to reduce the domain size by a factor of 8, while the following results are calculated for the full domain. There are approximately 2 billion total points (250 million per octant) in the wavelet grid compared to a uniform grid of $983,041^3$, for a compression ratio of $\mathrm{O}(10^8)$. This 3D function demonstrates the compression capabilities of the wavelet-adaptive grid for solution fields with extremely

localized features; however, it should be noted that the compression ratio tends to decrease as finer scales occupy a larger fraction of the domain. The compression performance will depend on the physics of problem being solved.

$$
\begin{aligned}
f(x,y) = \cos(40\pi x)\cos(40\pi y)\cos(40\pi z)\exp(-64\sqrt{x^2+y^2+z^2})+ \\
\cos(1200\pi x)\cos(1200\pi y)\cos(1200\pi z)\exp(-25000\sqrt{x^2+y^2+z^2})+ \\
\cos(30000\pi x)\cos(30000\pi y)\cos(30000\pi z)\exp(-1500000\sqrt{x^2+y^2+z^2})
\end{aligned}
\tag{19}
$$

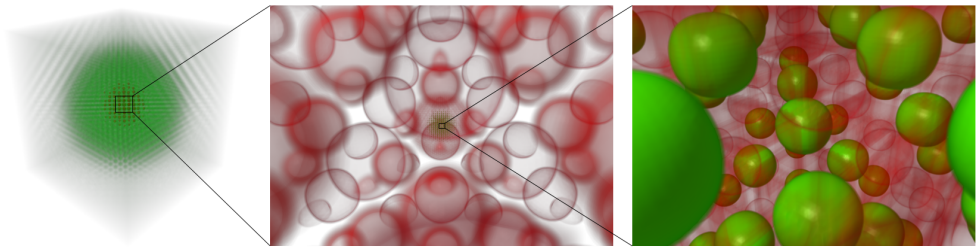

Figure 8: The 3D multiscale demonstration function used to demonstrate the ability of the wavelet-adaptive grid to capture length scales varying by six orders of magnitude. The volume rendering (generated by the Voreen framework of Meyer-Spradow et al. (2009) [75]) highlights selected gradient magnitudes of Equation 19.

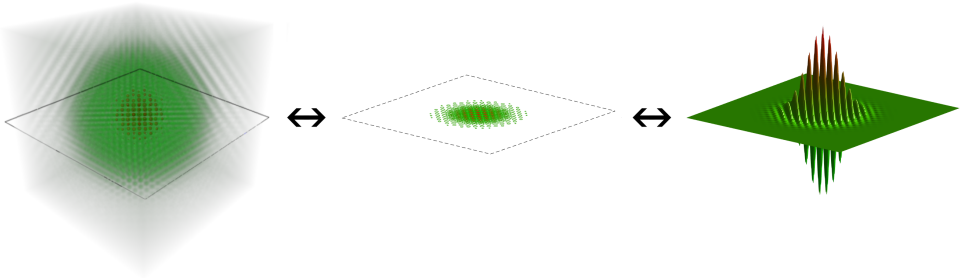

Figure 9: An illustration of the relationship between the 2D and 3D multiscale demonstration functions and visualization techniques. The volume rendering highlights selected gradient magnitudes of Equation 19, which in effect is similar to highlighting iso-surfaces of the function. The ability to visualize locations of steep gradients (*e.g.*, density) in a volume is well-suited to displaying the liquid-vapor interface in nucleate boiling problems. The volume rendering is shown on the left, a 2D example is recovered by extracting the plane at $z = 0$ (middle), and the visualization technique used in Figure 7 maps function values on the plane to the z-coordinate (right).

Figure 10 illustrates the effectiveness of wavelets to isolate detail in a 2D image from a nucleate boiling experiment. The grayscale images on the right show the magnitudes of the wavelet coefficients on each level of resolution throughout the domain. Each quadrant of this wavelet decomposition plot has a one-to-one correspondence to the 2D wavelet families enumerated in Table 2 and is related to the vertical (Family 1), horizontal (Family 2), and diagonal (Family 3) features of the original image. Each upper-left quadrant in the wavelet decomposition plot contains the decomposition of the next finer level, and each level of the wavelet transform picks up additional finer details in the boiling image. The scaling and wavelet operations can also be interpreted as low- and high-pass filters applied between adjacent levels of resolution. This representation stems from the early development of wavelet methods and their roots

in signal and image processing [Mallat (2008) [73]]. The wavelet decomposition plot highlights the localization of grid refinement about the liquid-vapor interface and the sparseness of the grid over the remainder of the domain. The localized structures of bubble formations in nucleate boiling make it a prime candidate for effective wavelet representation.

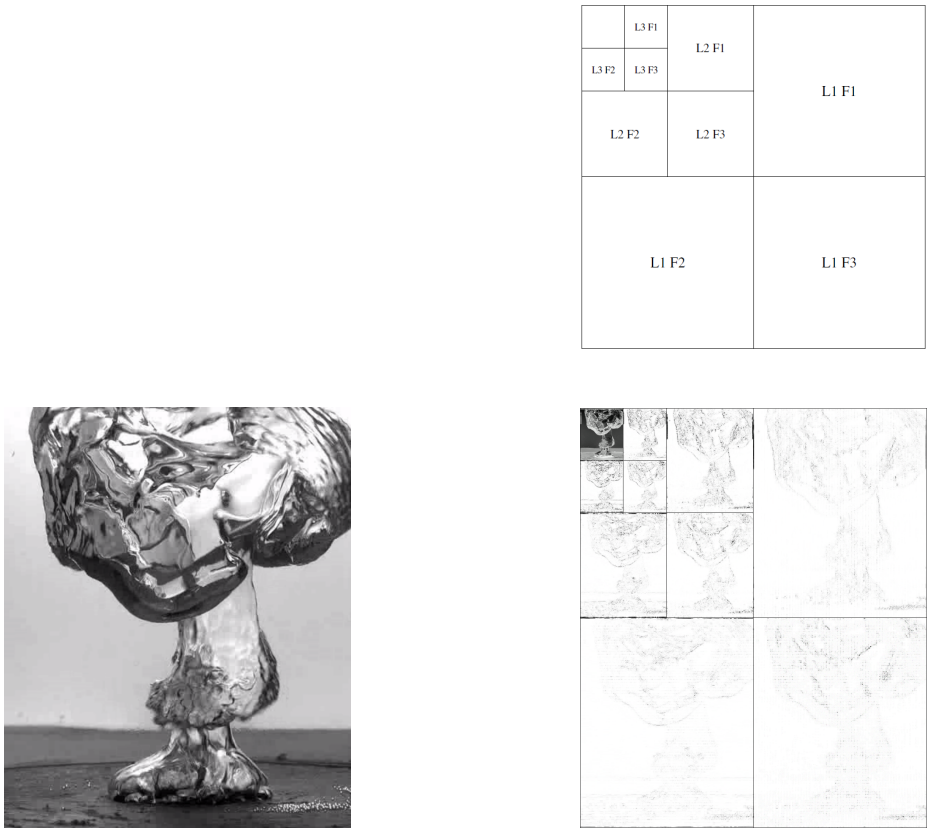

Figure 10: Multi-level wavelet representation of a 2D image from a nucleate boiling experiment with a small heater placed in the middle of a water tank [Boziuk and Glezer (2013) [14]]. The lower-left image shows the original nucleate boiling image with a vapor plume rising due to buoyancy effects. The lower-right image shows the results of multi-level wavelet decomposition. The upper-right shows a schematic of the separation of points by 2D wavelet family in the multi-level wavelet decomposition. The number preceded by 'L' indicates the level and the number preceded by 'F' is the wavelet family.

3.4 Verification and Validation

3.4.1 Error Properties

The error caused by compressing a data set with an adaptive wavelet transform and then reconstructing the values with an inverse transform is demonstrated using the test function defined in Equation 20 and shown in Figure 11. This function was chosen because it requires additional levels of refinement with the scaling and wavelet basis functions used in this work, in contrast to a polynomial function of lower order than the scaling and wavelet basis functions. Additionally, Equation 20 was used by other research groups [Holmstrom (1999) [51], Vasilyev (2003) [118], Wirasaet (2007) [126], Paolucci et al. (2014) [85]], so the various results can be easily compared. This test function demonstrates the wavelet reconstruction error relative to the user-prescribed error tolerance and the derivative convergence properties.

$$f(x,y) = \exp\{-200[(x-\frac{1}{2})^2 + (y-\frac{1}{2})^2]\} + \frac{1}{5}\sin(2\pi x)\sin(2\pi y) \qquad (20)$$

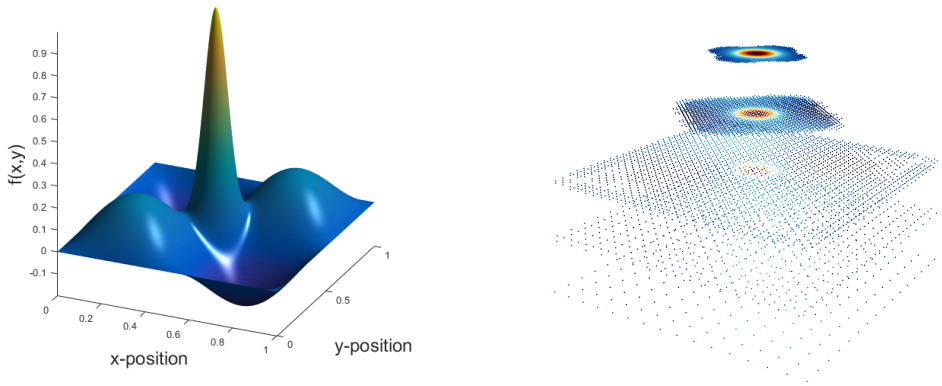

Figure 11: (Left) The 2D reconstruction test function. (Right) The wavelet-adaptive grid point representation of the test function separated by local resolution levels.

The quality of the wavelet representation is controlled by the user-defined wavelet

tolerance ϵ. This control involves a trade-off between grid compression and reconstruction accuracy. The left side of Figure 12 shows the maximum absolute error as a function of the wavelet tolerance for wavelets of order 4–10. The solid line is the one-to-one correspondence between the maximum error and the wavelet tolerance. The right side of Figure 12 is a plot of the maximum error versus the number of essential points retained by the wavelet transform for the different wavelet orders. The reconstruction error decreases and the number of points retained increases as the wavelet tolerance is decreased.

The derivative error is measured by comparing the discrete derivative values (using a centered finite difference formulation with the same order as the wavelets) to the evaluated analytical derivative of Equation 20. The same procedure is repeated for the second derivative. The results for the maximum error and the number of essential points required to represent the solution for wavelets and finite difference formulations of order 4–10 are shown in Figure 13. The same trends are seen as in the error for the function, but the error is about an order of magnitude higher for the first derivative for a given number of points and almost seven orders of magnitude higher for the second derivative. The overall order of accuracy of the method can be limited by either the wavelet or finite difference order of accuracy. While the order of accuracy of the wavelets and finite differences can be varied independently, the case where the two orders of accuracy are commensurate is shown here.

Figures 12 and 13 show that higher-order wavelets and finite differences produce much less error with significantly fewer grid points compared to lower-order wavelets. This reduces the compute time for higher-order wavelets due to the lower point count, but this benefit is somewhat offset by the increased number of time steps required for higher-order finite difference formulations due to the Courant–Friedrichs–Lewy stability condition. Depending on the time integration scheme used and the problem being solved, there is an optimum trade-off between the order of accuracy and this

time-step restriction to obtain the minimum time-to-solution. The 8^{th}-order wavelets require two orders of magnitude fewer points than 4^{th}-order wavelets, but the increase in order only stiffens the domain of dependence requirement for hyperbolic problems by a factor of two for interior points. It should be noted that this effect is worsened if the same order of accuracy for the finite differences is maintained all the way to the boundaries through the use of biased stencils. In the context of domain decomposition and larger parallel computations, this optimum will be affected and will tend to favor a lower-order stencil than in the single-GPU case due to halo region sizes associated with each stencil size. This effect will be investigated in future work.

Higher-order wavelets allow error tolerances to be prescribed closer to machine precision while maintaining a smaller grid size, which is desirable from both computational performance and accuracy perspectives. This is demonstrated using wavelet and finite difference representations of varying but equal orders for the test function defined by Equation 20 with the specified wavelet tolerance fixed at 10^{-12}. The total grid point count and the relative extent of each point type for various wavelet orders is shown in Figure 14. This shows the potential work savings of higher-order wavelets since the point count is reduced by a factor of 500 going from a 4^{th} to a 10^{th}-order wavelet for this test case.

The next three subsections present the results from three validation problems used to demonstrate the accuracy of the compressible part of the wavelet-based solver. All of these test cases use 4^{th}-order wavelets and 4^{th}-order finite differences. The accuracy of the derivatives near the boundaries is maintained using biased 4^{th}-order stencils in these locations. The wavelet tolerance is $\epsilon = 10^{-4}$.

The computing performance of the compressible part of the wavelet-based solver when higher-order wavelets and finite differences are used will be discussed in § 3.5 of this chapter.

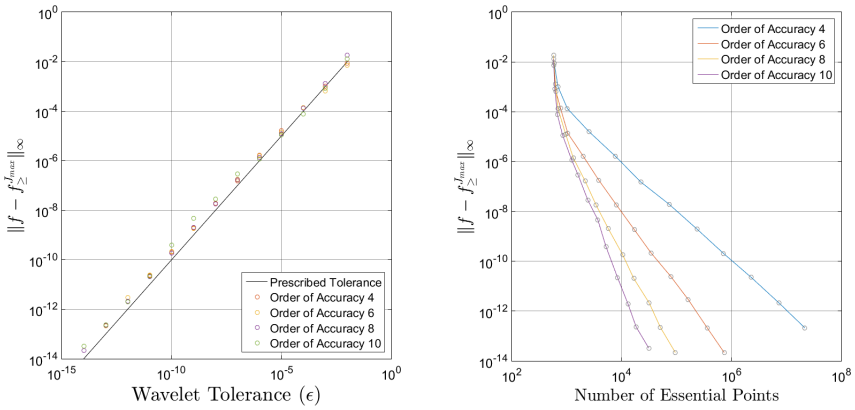

Figure 12: (Left) The maximum absolute error versus the user-prescribed wavelet tolerance for various wavelets orders. The solid line is a one-to-one relationship between the two values. (Right) The maximum absolute error versus the number of essential points, indicating that higher-order wavelets provide significantly better compression of the solution. The convergence rate for each order matches the expected slope.

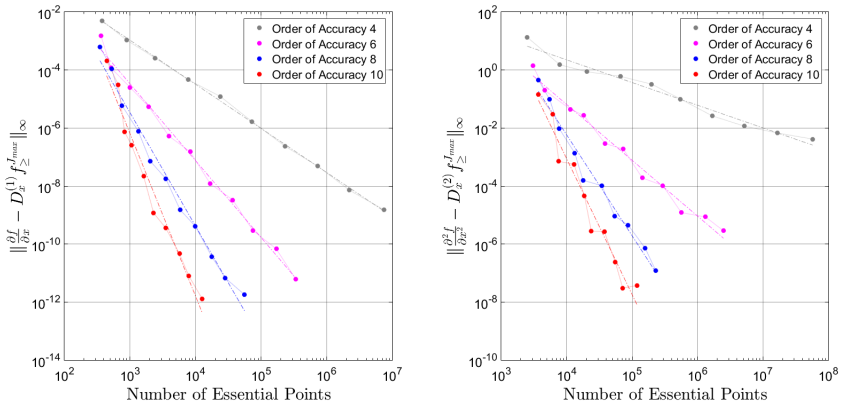

Figure 13: (Left) The maximum absolute first-derivative error versus the number of essential points. (Right) The maximum absolute second-derivative error versus the number of essential points. The convergence rate for each order matches the expected slope.

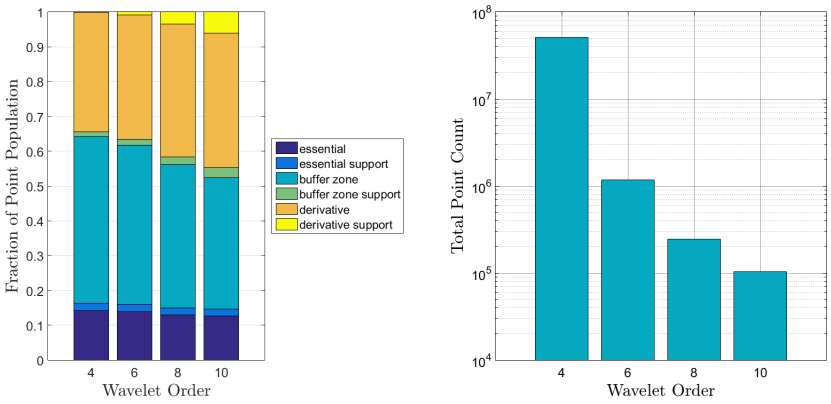

Figure 14: The point-type characteristics required to hold a wavelet tolerance of 10^{-12} by setting ϵ with varying, but equal orders for the wavelets and the finite difference formulations using the test function defined by Equation 20. (Left) The relative point-type population versus wavelet order. This indicates the relative memory cost for the different points. Since an individual point may serve several purposes depending on neighboring points (*e.g.*, being an essential point and a derivative point required by a neighbor), the categorization priority is determined with essential points leading and derivative interpolation support being last. The legend indicates the priority level with the most important at the top and proceeding in descending order. (Right) The total number of points versus the wavelet order.

3.4.2 Shock Tube Problem

A common numerical benchmark for compressible flow solvers is the flow in a shock tube. Figure 15 shows a simple schematic of this experiment consisting of a pipe with a diaphragm in the middle separating a gas with a high pressure and density on the left and the same gas at a low pressure and density on the right.

When the diaphragm is released, a shock front occurs between the high- and low-density gases at the diaphragm location. The shock moves to the right compressing the low-density gas, while a rarefaction wave moves to the left expanding the high-density gas. In between these two waves there is a contact interface between the two original gases.

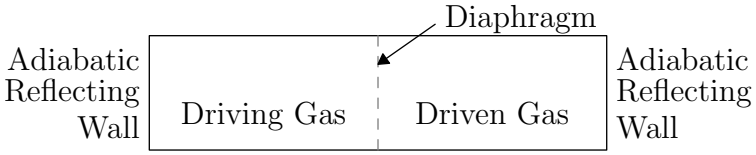

Figure 15: A simplified schematic of a shock tube experiment is shown. The diaphragm is broken (removed) instantaneously at the simulation start time.

For an inviscid, non-conducting fluid, the flow is governed by the compressible Euler equations, which have an analytical solution that was studied in great detail by Sod (1978) [103]. This problem is now commonly referred to as the *Sod Shock* problem. The inviscid solution has the rarefaction wave and true discontinuities for the contact interface and the shock, all moving at specific speeds. The details of the analytical solution can be found in Sod (1978) [103]. For a real gas, with small but finite values for viscosity and thermal conductivity, the location and motion of the characteristic flow features still follow the inviscid model, but the property discontinuities now have a finite thickness.

In the present test case, air is modeled as a two-component ($79N_2$:$21O_2$), calorically perfect ideal gas with the viscosity and thermal conductivity of air at 20 °C. The mass diffusion of each species into the mixture is modeled using a Schmidt number

of 0.7. The flow is one dimensional to compare with the analytical solution and so viscous interactions with the sidewall of the tube are ignored. The tube ends are adiabatic and perfectly reflecting. The initial conditions from Paolucci et al. (2014) [85] are a uniform initial temperature of 300 K, a high pressure of 1000 kPa, and a low pressure of 100 kPa. The initial pressure at the diaphragm position was smoothed using a hyperbolic tangent function with a characteristic dimensionless width of 10^{-3}. The shock tube has a length of 0.001 m, and the simulation was run up to 49.987 μs, which is enough time to clearly identify the rarefraction wave, the contact discontinuity, and the shock front before reaching the end of the shock tube.

All of the results from the 1D wavelet simulation compared very well to the inviscid analytical solution. For example, both density profiles are shown in Figure 16. Note that the wavelet simulation has fully resolved the contact and shock layer thicknesses using 10 levels of grid refinement. The transition region at the contact discontinuity arises from two sources. First, the continuous initial smoothing function had a transition region thickness on the order of 0.005. Second, the presence of viscous and mass diffusion cause further spreading of the transition region over the entire duration of the simulation. At this point in time, the transition region thickness has increased by a factor of 5, to about 0.025. These physical diffusion effects also operate in the shock layer region, except that the shock counteracts the effects of diffusion due to the nonlinear-steepening of the wave [Toro (2013) [112]]. In this example, 193 grid points were required to represent the solution compared to 15,361 points in a uniform grid, resulting in a compression factor of 80.

To fully verify that the wavelet simulation worked correctly in all directions, the same problem was run with the 1D flow aligned in the x-, y-, and z-directions. The results were identical in all cases.

The Sod Shock problem has traditionally been used to measure the performance of numerical methods in capturing the correct locations of shocks fronts and other

features in compressible flows. In particular, it highlights the amount of *artificial viscosity* required by shock-capturing schemes near the shock and contact discontinuities. In the wavelet simulation, these features were fully resolved without the use of any shock capturing schemes or artificial viscosity.

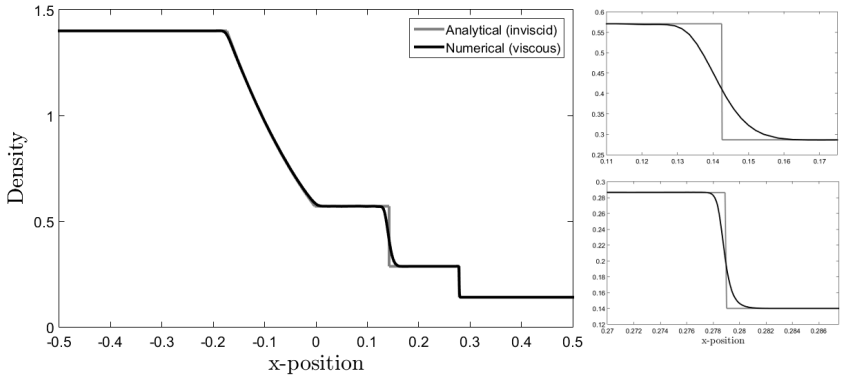

Figure 16: The computed density profile for the 1D flow in a shock tube compared to the analytical result for one instant in time. The contact discontinuity is enlarged in the upper-right plot, and the shock is enlarged in the lower-right plot. Note the difference in length scales in the enlarged plots.

3.4.3 Richtmyer-Meshkov Instability

The Richtmyer-Meshkov instability occurs when a planar shock wave accelerates an interface between two fluids of different densities. When the shock interacts with a bubble of lighter fluid, the instability is directly forced by the baroclinic torques caused by the misalignment of the density and pressure gradients at the interface. The resulting motion of the interface later develops into secondary shear-driven instabilities. This problem is presented here as a qualitative assessment of the wavelet simulation's ability to capture the essential physics of a complex multispecies compressible flow, and its ability to provide significant grid compression during the evolution of the solution.

The physical domain for this 2D simulation is shown in Figure 17. The non-dimensional height and width of the domain is one unit by four units, respectively.

The top and bottom boundaries are symmetry planes (lines), and the left and right boundaries are rigid, adiabatic walls. A helium bubble with a diameter equal to one is located as shown in the figure. It is surrounded by air and the density ratio between the two gases is approximately 8. The instability is driven by a normal shock wave, initiated just before the helium bubble by the bursting of a thin diaphragm. The initial conditions for this flow are the the same as in the previous Sod Shock problem.

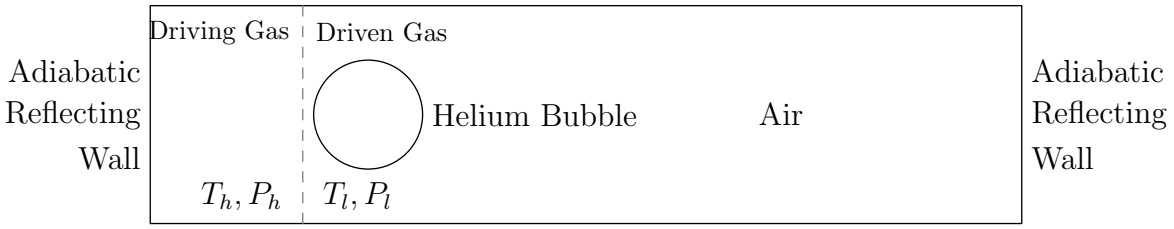

Figure 17: A simplified schematic of the Richtmyer-Meshkov instability problem in a shock tube with a helium bubble surrounded by air. The diaphragm is broken (removed) instantaneously at the simulation start time.

Figure 18 shows a snapshot of the density field a short time after the shock wave has passed the bubble. The bubble itself has been split into two pieces connected by a thin interface. Interfacial motions caused by secondary shear-induced instabilities can also be observed. The structure and evolution of this simulated flow is in qualitative agreement with the helium-air experiment in Giordano and Burtschell (2006) [46].

The wavelet-adapted grid at this time required approximately 820,000 points compared to about one billion for a uniform grid with the finest resolution, for a compression ratio of 1205.

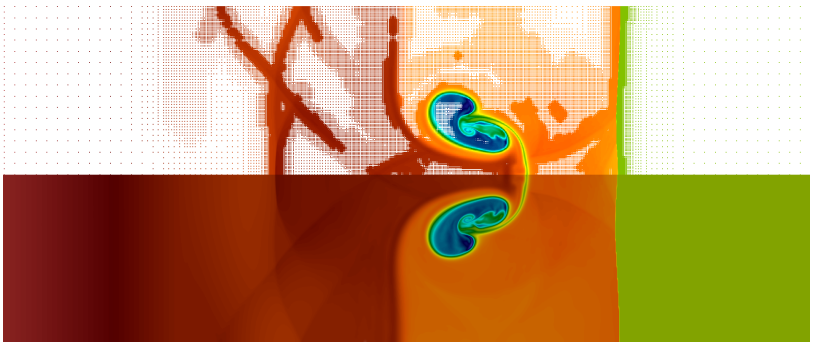

Figure 18: The density field in the Richtmyer-Meshkov instability arising from a 2D, viscous, shock-bubble interaction. At this time, the initially circular helium bubble has been effectively split into two separate bubbles and secondary shear-induced interfacial instabilities have occurred. While solved non-dimensionally, if the domain is scaled to a one meter length, the finest grid spacing would be 15 μm. The top portion of the plot shows the wavelet-adaptive point representation of the density and the bottom shows the reconstructed density field on a uniform grid.

3.4.4 Supersonic Flow Over a 2D Wedge

The supersonic flow of an inviscid, non-conducting, perfect gas over a 2D wedge with a half-angle of δ is the last numerical validation problem. This flow has an analytical solution, which makes it useful for this purpose. The test configuration is shown in Figure 19, in which the flow is from left to right and the wedge half-angle is 15°. The wavelet simulation of this steady flow ignores viscous effects at the top and bottom walls and uses simple inflow and outflow conditions at the left and right walls, respectively. The computational expense of this simulation was decreased by using a first-order upwind stabilization scheme [Swanson and Turkel (1992) [106]]. This scheme introduces numerical dissipation to the solution, which artificially limits the required resolution near the oblique shocks. The strict error-control properties of the wavelet simulation are diminished as a result; but this trade-off between computation expense and resolution shows the flexibility of the simulation to choose between accuracy and larger domain sizes on a problem-by-problem basis.

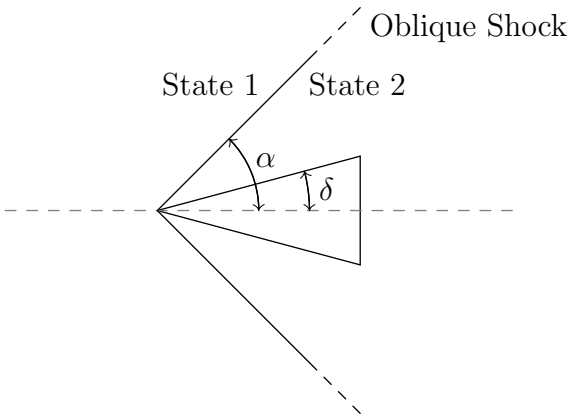

Figure 19: The 2D wedge configuration in a numerical supersonic wind tunnel experiment.

The explicit, analytical oblique shock wave relations for this problem can be found in Zucrow and Hoffman (1976) [132]. For reference, the oblique shock wave angle relations are shown in Figure 20 for a range of wedge half-angles and free stream Mach numbers.

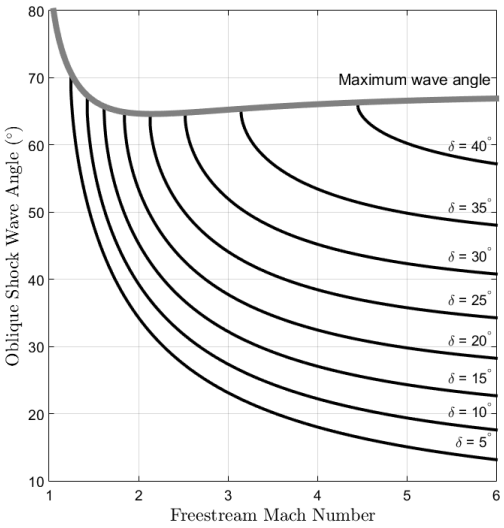

Figure 20: The oblique shock wave angle for a range of free stream Mach numbers and wedge half-angles.

The simulation was run to steady state for a given inlet (free stream) Mach number. The angle α of the attached oblique shock was then measured, and the pressure, density, and temperature ratios across the shock were evaluated. Also, the minimum free stream Mach number leading to the formation of a detached bow shock was determined.

Figure 21 shows a Schlieren plot of the density field for a wavelet simulation with a free stream Mach number of 1.8. The plot shows the magnitude of the density gradient and highlights the shock structure in the flow. The wavelet adapted grid for this case required approximately 5.7 million points compared to about 450 million for a uniform grid with the finest resolution used, for a compression ratio of $O(10^2)$.

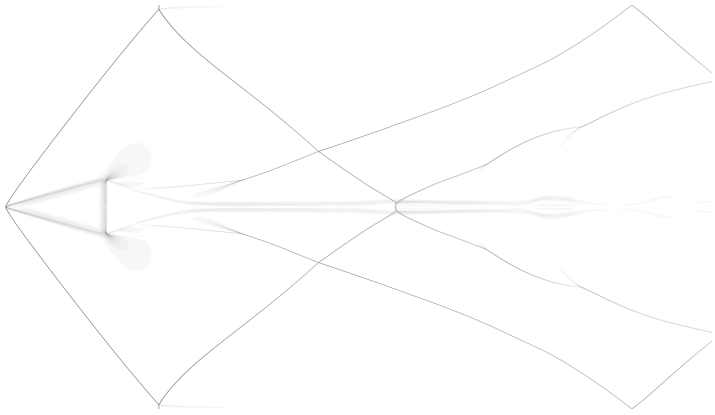

Figure 21: Numerical Schlieren plot for a 2D supersonic flow over a 15° wedge at a free stream Mach number of 1.8.

Results from the wavelet simulation for three different free stream Mach numbers are tabulated in Table 3 and compared to the analytical results. The simulation results are in excellent agreement with the analytical values (in parentheses). As shown in Figure 20, the oblique shock waves detach from the wedge at some minimum Mach number. This detachment and the subsequent formation of a bow shock is predicted by the wavelet simulation near the expected Mach number. Note that small deviations between the quantities of interest are expected since the analytical

solution is for a thermally non-conducting, inviscid, perfect gas, while the numerical solution includes these physical diffusion effects. In particular, viscous effects near the stagnation point at the front of the wedge have a larger range of influence at the lower Mach numbers tested.

Table 3: Tabulated results for the 2D supersonic flow over a wedge with a half-angle of 15° for a range of free stream Mach numbers. The analytical results are in parentheses.

Free Stream Mach Number	Pressure Ratio	Temperature Ratio	Density Ratio	Shock Angle (°)
1.6	-	-	-	detached ✓
1.62	2.280 (2.277)	1.289 (1.285)	1.769 (1.772)	63.1 (63.3)
1.7	2.151 (2.150)	1.262 (1.261)	1.704 (1.705)	56.2 (56.0)
1.8	2.135 (2.138)	1.259 (1.258)	1.696 (1.699)	51.3 (51.3)

3.5 Performance

The performance of the wavelet-adaptive method on the GPU architecture is strongly dependent on optimization for thread-level or Single Instruction Multiple Data (SIMD) parallelism. GPUs work particularly well for uniform grid applications since memory locations of neighboring points can be directly computed and all points within a kernel typically perform the same operations, thus avoiding warp divergence. Adaptive methods lose some of these benefits since neighboring points are not known *a priori*. As a result, special care is needed when implementing these methods in order to minimize warp divergence and maximize memory-access coalescence. GPUs also require a large number of active threads to effectively hide memory latency. Thus, GPU performance has a significant dependence on problem size up to a certain point, at which the performance starts to scale more linearly with total point count.

The problem size performance of the wavelet-adaptive method is measured using the elapsed wall time during one global time step in the evolution of an array of cylindrical shocks (*i.e.*, an extension of the 1D shock tube problem) to systematically vary the problem size with the number of grid refinement levels maintained at 10

additional levels. The performance is measured with all levels of resolutions up to the finest levels containing points. The grid refinement level is held fixed since the wavelet transform requires more work as the number of refinement levels increases, as can be seen in Algorithm 1. Thus, the fixed resolution level allows the problem size dependence in the wavelet method to be isolated. The elapsed wall time for the computation is normalized by the corresponding value for the smallest problem size considered of approximately 75,000 points. Problems smaller than this are relatively simple, such as linear heat diffusion in two dimensions, and are easily (and more quickly) solved using uniform grid methods. Note that above this minimum problem size, the streaming multiprocessors on the GPU are fully occupied to ensure effective utilization of the available GPU resources by hiding memory latency.

Figure 22 shows the performance for the wavelet-adaptive method with varying problem sizes. The most notable result is the super-linear scaling with problem size that remains constant. This trend indicates higher computational efficiency with larger problem sizes on the GPU and efficient use of the GPU cache system.

The normalized wall time performance of the wavelet solver for the 2D Richtmyer-Meshkov instability problem versus the level of grid refinement from a base grid of 65x17 is shown in Figure 23. This is compared to a uniform grid solver on the same problem with equivalent resolution. For this case, the normalization reference point is the value at the intersection of the two performance curves. The GPU-based uniform grid solver used in this comparison has the same code base as the wavelet solver, but all of the wavelet operations (and overhead) are disabled so it is equally well-optimized for the GPU to establish a fair comparison to the wavelet-adaptive method. At problem sizes near the size of the base grid and the first few levels of refinement, the simulation may run up to four times faster on uniform grids than on wavelet-adaptive grids because of the additional overhead required by the

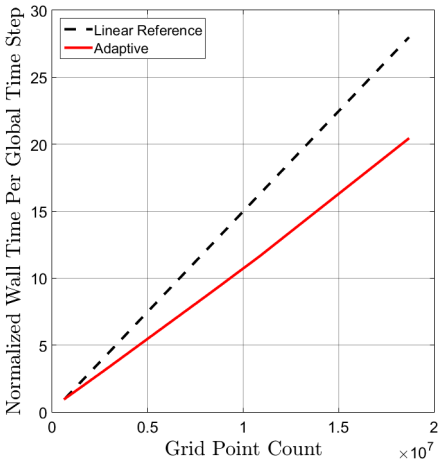

Figure 22: Super-linear scaling for varying problem size at a fixed number of refinement levels. The reference curve has a slope of unity.

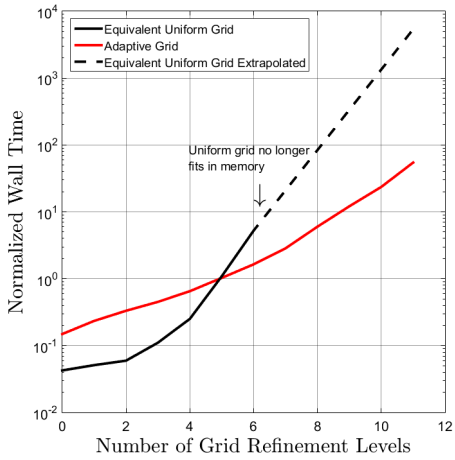

Figure 23: Comparison of the normalized run times for GPU-based wavelet-adaptive grids and uniform grids of equivalent resolution. The test case uses the 2D Richtmyer-Meshkov instability problem with a base grid of 65x17. The dotted line is an extrapolation based on the total point count.

wavelet transforms for adapting the grid. However, the performance for the wavelet-adaptive method is better after five levels of refinement (corresponding to uniform grids with approximately 10^6 points). After six levels of refinement, the uniform grid problem at an equivalent resolution to the wavelet-adaptive grid required too many points to fit in the available GPU memory. The uniform grid wall time was then linearly extrapolated past this point based on the assumption that all streaming multiprocessors on the GPU are fully occupied and that the number of additional compute blocks is proportional to the number of points. At eleven levels of refinement (obtained with a wavelet tolerance of $\epsilon = 10^{-4}$), the wavelet simulation runs about 100 times faster than an equivalent uniform grid simulation (if adequate memory was available to run it). This increased performance at high refinement levels is also indicated in Figure 22 because the grid point count increases with the level of refinement.

For 3D problems, the performance is even better. The grid refinement level necessary to favor wavelet-adaptive grids is smaller since the total point count increases as n^3, where n is the node count along each domain dimension, and the trade-off point (*i.e.*, the intersection of the curves) occurs near the same value for the total point count. A well-resolved, uniform grid problem size in 3D could easily surpass the available GPU memory making such simulations impossible to perform. The wavelet-adaptive method allows these larger problems to be solved on a reasonable amount of computing hardware and at a significantly reduced cost. This makes 3D simulations much more accessible to the average user.

Further performance savings can be realized by increasing the order of accuracy of the wavelet-adaptive method. This is illustrated using the 2D Richtmyer-Meshkov instability problem with higher-order wavelets and finite difference formulations. In the following results, the finite difference order of accuracy is always commensurate to the wavelet order of accuracy. The wall time performance relative to 4^{th}-order

accuracy is shown in Figure 24. The wall time increases by a factor of 2.5 going from 4^{th}- to 10^{th}-order accuracy. However, the total point count decreases by a factor of 500 as shown in Figure 14. The net effect is that higher-order accuracy reduces the overall cost of the wavelet simulation due to the large reduction in the number of points required to represent the solution.

It should be noted that the performance benefit of a higher-order wavelet method running on a single GPU may not translate to multi-GPU implementations. There are additional communication expenses between the multiple GPUs and so the overall performance will largely depend on the ability to hide this communication expense by overlapping the computation time. Due to larger stencils and larger halo region sizes, a higher-order wavelet method may prematurely limit strong scaling performance for multi-GPU implementations. This will be investigated further in future work.

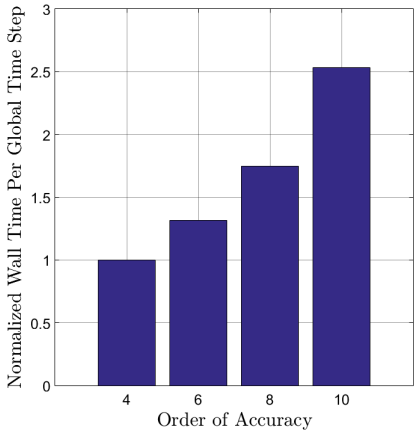

Figure 24: The performance of the wavelet solver on the 2D Richtmyer-Meshkov instability problem versus the order of accuracy of the wavelets and finite-difference stencils. The wavelet and derivative stencil sizes corresponding to higher-order accuracy require additional computational overhead.

3.6 Conclusions

The current work is the first known entirely GPU-based wavelet-adaptive compressible-flow solver that completely avoids data transfers over the PCI-express bus during solving. The first working version of the solver was presented by Forster and Smith (2014) [41]. Other groups [Rossinelli et al. (2011) [92]] investigated the acceleration of wavelet-adaptive-grid frameworks by offloading some of the work to GPUs. However, that approach incurs significant overhead with PCI-express bus transfers between the host CPU and the GPU, although Rossinelli et al. (2011) [92] and Van Rees et al. (2013) [117] hide much of this overhead (˜75%) with concurrent and expensive multiphysics calculations on the CPU.

A data set with localized features can be accurately resolved with a wavelet-adaptive multiresolution representation using a smaller number of points compared to a uniform grid with the same resolution. A 1D example of a step function showed a compression ratio of 118. Similar examples for 2D and 3D showed compression ratios up to $O(10^5)$ and $O(10^8)$, respectively. These large grid compression ratios significantly reduce memory requirements and the number of computations required during a simulation of a set of PDEs. This means that a wavelet-adaptive grid on a desktop workstation with a single GPU can represent highly resolved solutions to complex three dimensional PDE problems that would otherwise be inaccessible.

The numerical accuracy of the wavelet representation of a test function and the finite difference evaluations of its derivatives was verified. The function error scales almost linearly with the user-specified wavelet tolerance and can reach to almost machine precision with wavelets of order 8–10. Another benefit of high-order wavelets is the reduction in the number of points needed to represent the function. This number decreased by a factor of 500 going from 4^{th}- to 10^{th}-order wavelets. The same trends were seen for the derivative error in the function using finite difference formulations with the same order of accuracy as the wavelets. The absolute errors though were

an order of magnitude higher for the first derivative and up to seven orders higher for the second. The sensitivity of the solution with respect to the wavelet tolerance ϵ depends on the physics of the problem, and the error can be controlled by varying this single parameter.

The GPU-based, wavelet-adaptive, compressible-flow solver was shown to perform well on several compressible, high-Mach number flows. These validation cases showed that the solver could resolve all of the relevant flow features to within an accuracy measured by the user-specified wavelet tolerance. An important feature of these problems, in relation to ultimately solving multiphase flow problems, is that they demonstrate the ability to efficiently capture shock waves and contact discontinuities in the solution.

The GPU-based, wavelet-adaptive, compressible-flow solver was compared to a uniform grid finite-difference, compressible-flow solver that also runs entirely on the GPU. With the 2D Richtmyer-Meshkov instability problem as a test case, the time-to-solution for the GPU-based simulation was two orders of magnitude faster that the uniform grid solver, with three orders possible at maximum resolution. While this is a projection due to memory limitations when using uniform grids, part of the motivation behind the wavelet method is to create an optimized grid to reduce memory requirements so that larger problems can be solved with the available memory. Figure 23 shows the extent to which this is possible. By comparing the performance of the traditional uniform grid solver and the wavelet-adaptive-grid solver, the reader can estimate the possible performance gains based on their own particular solver choices and hardware configurations for various applications based on the performance of existing finite-difference simulations.

In the comparison of the GPU-based uniform grid solver and the GPU-based wavelet-adaptive solver, it was observed that the wavelet-adaptive, finite-difference method has an order of magnitude *lower* average throughput of grid points per unit of

time than the uniform grid solver running on the same GPU. The lower performance is caused by the computing overhead from GPU warp divergence and data lookup. This extra work is more than offset by the reduced problem sizes offered by the grid compression from the wavelet-adaptive method. These results show that the wavelet-adaptive solver performs well on a GPU, a modern hardware architecture that relies heavily on SIMD parallelism and use of reduced cache sizes. This is a important step forward in the continued growth of wavelet-adaptive methods for use in advanced HPC systems.

CHAPTER IV

ALL-MACH NUMBER COMPRESSIBLE FLOW

4.1 Introduction

Multiphase flows contain a variety of physical processes, including vaporization, condensation, turbulence, and flow and interfacial instabilities. These complex flows may be nearly incompressible or experience significant compressibility effects as in high-speed flows, sometimes within a single domain. Historically, these two Mach number regimes required the use of separate solvers to efficiently and accurately handle the physical problem. This separation creates difficulties if the two regimes occur within a single domain, which is the case for the problem of nucleate boiling with acoustic forcing of the liquid-vapor interface introduced in Chapter 3 of this thesis, which covers the development of the wavelet-adaptive compressible solver that runs entirely on the GPU architecture.

The present chapter extends the work from Chapter 3 to adapt it to solving the compressible Navier-Stokes accurately at all Mach numbers, which is made possible with the introduction of a dual-time stepping method with preconditioning. This work continues towards the end goal of a GPU-based all-Mach number solver for multiphase flows with phase change that may have physical processes with disparate spatial and temporal scales. The disparity in temporal scales is efficiently handled using a preconditioned dual time-stepping procedure combined with a spatially adaptive flow solver implemented entirely on the Graphics Processing Unit (GPU) architecture using the Wavelet-Adaptive Multiresolution Representation (WAMR). The focus of the present work is on the verification of the solver for flows approaching the incompressible limit. The benefits of dual-time stepping are three-fold: 1) it allows for

the solution of nucleate boiling with and without acoustics using the same solver; 2) it complements the error control of the wavelet-adaptive spatial discretization since it allows for strict error control over the residuals of the equations during the advancement of physical time; and 3) the implicit dual-time stepping procedure can be efficiently implemented on the GPU architecture.

Direct numerical simulation of multiphase flows, particularly low-speed compressible flows, require numerical methods that accurately discretize the governing equations in a stable manner and are minimally dissipative. Additionally, the co-located grid arrangement used in this work suffer from odd-even decoupling in the incompressible limit, which requires special care and is addressed in this chapter. The typical numerical errors present in such simulations consist of truncation and aliasing errors. Aliasing error was first pointed out by Phillips (1959) [87] and was attributed to the non-linear convective terms of the Navier-Stokes equations combining two low-wavenumber modes in a way that generates higher wavenumber modes than the grid can resolve. These high-wavenumber modes are interpreted by the grid as long-wave solution features. Such solution features are completely unphysical and numerical in nature because the underlying grid cannot support wavelengths in the flow that are less than two grid spacings. A carefully designed numerical scheme should either dampen or inherently cancel these discretization errors from the convective term in order to provide a stable and physically correct solution.

The convective term in the Navier-Stokes equations can take on several different forms that are analytically equivalent, including the advective, divergence, skew-symmetric, or rotational form; however differences arise when they are discretized, and they can take on significantly different properties. On uniform grids and with the skew-symmetric convective operator, Kwak et al. (1975) [64] showed that the integral primary and secondary conservation properties (*i.e.*, momentum and kinetic energy) are discretely preserved. Blaisdell (1991) [8] demonstrated the numerical

differences between the advective and skew-symmetric forms and identified aliasing errors that occur in the advective form. Zang (1991) [129] showed that the rotational form also suffers from aliasing errors while the skew-symmetric form provided better performance. Blaisdell et al. (1996) [9] demonstrated that the skew-symmetric form provides some cancellation of aliasing errors through numerical tests and Fourier analysis. The improved numerical properties of the skew-symmetric form are important in constructing a minimally dissipative simulation since they reduce the observed aliasing error and reduce the reliance on dissipative filtering techniques. It was pointed out in Canuto et al. (2012) [17] that aliasing errors are reduced with increasing grid resolution; however, even with wavelet-adaptive grids it is better to prevent the problem to the largest extent possible using more sophisticated discretization techniques rather than relying on brute force to fix the problem through grid adaptation alone, since the latter method can be prohibitively expensive in 3D. In addition, higher-order finite differences exhibit more aliasing as they approach spectral-like behavior [Desjardins et al. (2008) [28]], which reinforces the need to use the skew-symmetric form of the convective term. Thus, the present work addresses the issue of discretization error by solving the skew-symmetric form of the compressible Navier-Stokes equations on a wavelet-adaptive grid.

In addition to reducing aliasing error, it is used to avoid the odd-even decoupling behavior associated with non-staggered grids with a co-located variable arrangement and to reduce aliasing errors with high-order spatial discretization. This avoids the need for explicit filtering [Cook and Cabot (2005) [20]] and/or a Rhie-Chow type interpolation procedure that implicitly adds high-order artificial dissipation [Rhie and Chow (1983) [91], Shen et al. (2001) [99], Date (2003) [24], Zhang et al. (2014) [130]]. The skew-symmetric form allows for the use of less numerical dissipation than the divergence form of the convective term requires, which is important for the study of multiphase flows with interfacial instabilities (*e.g.* nucleate boiling) since artificial

dissipation can prevent the formation or significantly alter the long-time behavior of the flow and interfacial instabilities.

The time integration scheme used in this work is the preconditioned dual-time stepping technique first introduced by Jameson (1991) [54] and improved upon by Buelow (1995) [16] and Oefelein (1997) [82] to solve the compressible Navier-Stokes and Euler equations at low Mach numbers with non-uniform grids and real fluid properties. Since the compressible form of the governing equations permit acoustic waves, solving at low Mach numbers creates stiffness due to the disparity of the convective, diffusive, and acoustic time scales. Depending on the physics of the problem and the quantities of interest, it may be desirable to take the maximum allowable time steps to sufficiently resolve the physics at one of those time scales. This requires the use of an implicit time integration scheme, which can provide a time accurate solution for the physical processes that are temporally fully resolved but neglect the time accuracy of the faster processes. An example of decoupled time scales would be the small effect of acoustics on low-speed flow over an airfoil where only the accuracy of the convective time scale is of interest. In addition to selectively resolving physical time scales, it is also necessary to precondition the system of equations to allow for the accurate solution at very low Mach numbers (*e.g.*, $Ma = 10^{-6}$). This dual-time stepping procedure also requires some modifications to work with the wavelet-adaptive grid and to perform well on the GPU architecture.

The next chapter in this thesis extends the present work to include a level-set model for multiphase flows. The end result will be a temporally and spatially accurate multiphase simulation of the compressible Navier-Stokes equations at all Mach numbers. This simulation will use existing numerical techniques that have been carefully modified and designed to work together to provide accurate solutions and reduced simulation times on modern GPU computing hardware. The objective at the end of this three-part series is to perform high-fidelity simulation of low- and high-speed

compressible, multiphase flows within the same solver, particularly for applications of nucleate boiling flows with acoustic interfacial forcing. Note that for such flows, the compressible form of the Navier-Stokes equations is actually more faithful to the underlying physics of the problem than the incompressible formulation, particularly when the operating conditions may violate the assumptions of the Boussinesq approximation and/or when acoustic effects are significant. The performance of this minimally dissipative, wavelet-adaptive, co-located finite-difference solver on GPU architectures is a crucial step in the path towards exascale computing.

4.2 Governing Equations

In these simulations, the physics for the evolution of a compressible, viscous flow is governed by the following set of conservation equations for mass, momentum, energy, and species transport. These are the same equations as those in Chapter 3 § 3.2, but the skew-symmetric form of the convective terms are used here.

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_j}{\partial x_j} = 0 \tag{21}$$

$$\frac{\partial \rho u_i}{\partial t} + \frac{1}{2}\left(\frac{\partial \rho u_i u_j}{\partial x_j} + u_j \frac{\partial \rho u_i}{\partial x_j} + \rho u_i \frac{\partial u_j}{\partial x_j}\right) = -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j} + \rho g_i \tag{22}$$

$$\frac{\partial \rho E}{\partial t} + \frac{1}{2}\left(\frac{\partial \rho E u_j}{\partial x_j} + u_j \frac{\partial \rho E}{\partial x_j} + \rho E \frac{\partial u_j}{\partial x_j}\right) + \frac{\partial p u_j}{\partial x_j} = -\frac{\partial q_j}{\partial x_j} + \frac{\partial u_i \tau_{ij}}{\partial x_j} + \rho u_i g_i \tag{23}$$

$$\frac{\partial \rho Y_k}{\partial t} + \frac{1}{2}\left(\frac{\partial \rho Y_k u_j}{\partial x_j} + u_j \frac{\partial \rho Y_k}{\partial x_j} + \rho Y_k \frac{\partial u_j}{\partial x_j}\right) = -\frac{\partial J_{j,k}}{\partial x_j} \tag{24}$$

Here, ρ is the density, u_i is the velocity vector, p is the pressure, τ_{ij} is the viscous stress tensor, g_i is the acceleration of gravity vector, E is the total specific energy defined as the sum of the specific internal and kinetic energy ($E = e + \frac{1}{2}u_i u_i$), and q_j is the heat flux. In the last equation for species transport, Y_k is the mass fraction

for the k^{th} species and $J_{j,k}$ is the diffusive flux of species Y_k. Identically, the sum of the mass fractions must be unity. To ensure this $k-1$ species are explicitly tracked, and the final component is calculated as $Y_k = 1 - \sum_{m=1}^{k-1} \rho_m Y_m$. This final component is typically chosen to be an inert component (*e.g.*, nitrogen) in the mixture to minimize the effects from accumulation of numerical error arising during species transport [Poinsot and Veynante (2005) [89]].

These conservation equations are augmented by a set of constitutive equations for the viscous stress, heat flux, and mass diffusion. Thermodynamic properties will use an equation of state appropriate to the material. For example, air could be modeled as a calorically perfect ideal gas.

4.3 Numerical Methods

4.3.1 Wavelet-Adaptive Method

The governing equations 21–24 are discretized using a wavelet-adaptive grid and a centered finite difference scheme is used for the derivatives. The present work uses second-generation wavelets implemented through a *Fast Wavelet Transform* that uses p^{th}-order Lagrange interpolating polynomials to determine the wavelet coefficients. The method is discussed in detail in Chapter 3 of this thesis. The result of that work is a GPU-based wavelet-adaptive solver for high-speed, compressible flows; a traditional, explicit, compressible flow solver.

4.3.2 Dual-time Stepping

To develop a GPU-based wavelet-adaptive solver for *low-speed* compressible and *incompressible* flows, a preconditioned dual-time stepping method is introduced to allow accurate solutions to very low Mach number flows (*e.g.* $10^{-7} \leq Ma \geq 10^{-2}$). This mach number range is not accessible to a traditional explicit compressible solver like the one developed in Chapter 3.

The dual time-stepping approach, also known as pseudo-transient continuation,

allows for implicit integration of the physical-time system of equations while maintaining the simplicity and computational performance characteristics of an explicit scheme [Buelow (1995) [16], Kelley and Leyes (1996) [59], Oefelein (1997) [82]]. This is achieved by introducing a pseudo-time derivative to the system of time-dependent Partial Differential Equations (PDEs). The pseudo-time derivative goes to zero in the pseudo-time, steady-state limit of the modified system, which then recovers the original time-dependent system. Since only the pseudo-steady-state condition of the modified system is important in recovering the original system, the system of equations in pseudo-time can be modified (without loss of accuracy) to optimize the convergence to steady-state in fewer pseudo-time iterations [Jameson (1991) [54], Jameson (2015) [55]]. The accelerated convergence is achieved through a preconditioning step that modifies the characteristics (*i.e.*, propagation speeds) of the pseudo-transient system to alleviate stiffness and improve the condition number of the system to allow for more efficient time marching in pseudo-time. This approach can be a good alternative to more general non-linear implicit solvers since the dual time-stepping method takes into account the underlying structure of the PDEs and can succeed where standard implicit solvers may stagnate around local minima [Kelley and Leyes (1996) [59]]. Additionally, the preconditioner contains the "Mach-squared" factor that is observed in asymptotic analysis of the governing equations in the incompressible limit [Muller (1998) [78], Oefelein (1997) [82]], which allows the solution of the compressible form of the governing equations at all Mach numbers.

The GPU architecture is particularly sensitive to memory access patterns and has limited cache per thread compared to other current HPC architectures [Woolley (2013) [127]]. These characteristics allow GPUs to work well with explicit solvers, but their performance is less than ideal for non-linear implicit solvers because of the sparse matrix and general matrix preconditioning algorithms that are used [Saule et al. (2013) [95], Li and Saad (2013) [69]]. This decline in performance is largely the

result of memory latency problems. In contrast, the dual time-stepping procedure is especially attractive for use with the GPU architecture because it avoids these latency bottlenecks while maintaining the high-performance characteristics (*e.g.*, data access patterns) of explicit methods. Thus, dual time stepping offers the stability of implicit time integration methods together with improved convergence rates in pseudo-time using specialized preconditioning.

At each physical time step, the preconditioned pseudo-time system is integrated to steady-state to recover the original time-dependent equations. The process is repeated at each physical time step until the end of the simulation. This process requires an inner pseudo-time integration loop inside of the physical time integration, and this is illustrated in Figure 25.

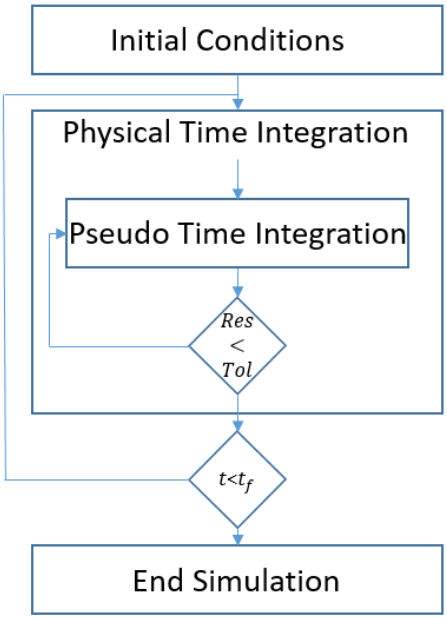

Figure 25: Dual time-stepping schematic. The advancement of physical-time is initiated with the specification of the initial conditions and is advanced through an implicit integration formulation until the final simulation time. Within each physical time step, another pseudo-time system of equations (original equations with a pseudo-time derivative and modified to remove stiffness) is integrated in pseudo-time until the original system of equations is satisfied at the $n+1$ phsyical time. Steady-state of the pseudo-time system is determined by a user-defined tolerance and the pseudo-time integration is terminated once the maximum residual of the original system of equations is decreased below this tolerance.

Equation 25 introduces the pseudo-time derivative, and it is written in flux vector notation and assumes the divergence form of the convective terms for convenience and clarity; however, all of the concepts are immediately applicable to the skew-symmetric form of the equations presented in Equations 21-24. Each of the vectors are defined in Equation 26.

$$\boldsymbol{\Gamma}\frac{\partial \boldsymbol{Q}}{\partial \tau} + \frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{E}}{\partial x_1} + \frac{\partial \boldsymbol{F}}{\partial x_2} + \frac{\partial \boldsymbol{G}}{\partial x_3} = 0 \tag{25}$$

$$\begin{aligned}
\boldsymbol{Q} &= \{\ p,\ u,\ v,\ w,\ T,\ Y_k\ \}^T \\
\boldsymbol{U} &= \{\ \rho,\ \rho u,\ \rho v,\ \rho w,\ \rho E_t,\ \rho Y_k\ \}^T \\
\boldsymbol{E} &= \{\ \rho u,\ \rho uu + p - \tau_{11},\ \rho uv - \tau_{12},\ \rho uw - \tau_{13},\ (\rho E_t + p)u,\ \rho Y_k u\ \}^T \\
\boldsymbol{F} &= \{\ \rho u,\ \rho vu - \tau_{12},\ \rho vv + p - \tau_{22},\ \rho vw - \tau_{23},\ (\rho E_t + p)v,\ \rho Y_k v\ \}^T \\
\boldsymbol{G} &= \{\ \rho u,\ \rho wu - \tau_{13},\ \rho wv - \tau_{23},\ \rho ww + p - \tau_{33},\ (\rho E_t + p)w,\ \rho Y_k w\ \}^T
\end{aligned} \tag{26}$$

Here, $\boldsymbol{Q}$ is the vector of primitive variables, $\boldsymbol{U}$ is the vector of conserved variables, $\boldsymbol{E}$ is the vector of fluxes in the x_1-direction, $\boldsymbol{F}$ is the vector of fluxes in the x_2-direction, $\boldsymbol{G}$ is the vector of fluxes in the x_3-direction, Γ is the preconditioning matrix, τ is the pseudo-time, p is the pressure, T is the temperature, and $\{u,\ v,\ w\}$ are the components of the u_i velocity vector. The remaining variables are consistent with the governing equations defined in Equations 21-24.

The modified system of equations is defined in terms of primitive variables, $\boldsymbol{Q}$, for three reasons. First, the primitive variable formulation in terms of pressure and temperature simplifies fluid property calculations with multiple species since a non-linear iterative solve for temperature from energy can be avoided. The second reason is in the formulation of the preconditioning matrix, where the pressure variable helps isolate and eliminate the pressure singularity problem at low Mach numbers. The final reason has to do with stiff equations of state. For example, in the case of liquid

water it is easier to accurately determine the density from the pressure rather than trying to evaluate the pressure from the density. This is because the large physical stiffness (*i.e.*, $\partial p/\partial \rho$) of liquid water combined with a density-based equation of state will magnify small numerical errors in the density field during the evaluation of the pressure. As a result, large variations in the calculated pressure can be observed.

Equation 25 can be rearranged to place all of the physical terms and the inverse preconditioning matrix on the Right-Hand Side (RHS), as shown in Equation 27. The RHS contains the residual of the original equations and it is driven to zero (within a user-specified tolerance) as the modified system is marched towards steady-state and the pseudo-time derivative approaches zero. This process requires a suitable approximation of the physical time derivative. In this work, the physical time derivative is approximated with the 2^{nd}-order Backwards Difference Formulation (BDF) shown in Equation 28. This particular scheme is chosen for the compromise of low-storage requirements, formal order of accuracy, and its large absolute stability region in the complex domain. Alternative approximations can be made, and this has been investigated by Jameson (2015) [55].

$$\frac{\partial \boldsymbol{Q}}{\partial \tau} = -\boldsymbol{\Gamma}^{-1}\left(\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{E}}{\partial x} + \frac{\partial \boldsymbol{F}}{\partial y} + \frac{\partial \boldsymbol{G}}{\partial z}\right) \tag{27}$$

$$\frac{\partial \boldsymbol{U}}{\partial t} = \frac{1}{\Delta t}\left(\frac{3}{2}\boldsymbol{U}^{n+1} - 2\boldsymbol{U}^{n} + \frac{1}{2}\boldsymbol{U}^{n-1}\right) \tag{28}$$

The physical time derivative approximation requires the storage of the solution at previous time steps, which creates some difficulty for a wavelet-adaptive grid that can change at every time step. This problem is addressed using a sliding window where the grid for the set of stored solutions at the three different times is formed so that all of the grid points at the $n+1$ time step are present in the two previous time steps. This is easily and efficiently achieved by feeding the wavelet coefficients from the stored solutions and the current solution through the thresholding operation

where wavelet interpolation will reconstruct the missing points in the grids of the earlier time steps. This method provides additional control over retaining grid points from previous times that are not present in the grid at the $n+1$ time step through specification of the wavelet tolerance for each grid. The reconstruction of missing grid points at subsequent time steps is illustrated in Figure 26.

The rest of the wavelet-adaptive procedure remains unchanged from Chapter 3 of this thesis. The buffer zone extent remains unchanged even for physical acoustic and convective CFL numbers larger than unity since the grid is adapted at each iteration of the inner loop, which is required to take a pseudo-time step with a modified-CFL number of less than or equal to unity. This work assumes a physical CFL number of less than or equal to unity since the goal is to obtain time-accurate solutions of multiphase flows. It should be noted that there are multiple CFL numbers, both acoustic and convective, depending on the time scale of interest, and the physical time step should be chosen accordingly. Additionally, one needs to consider the von Neumann Number (VNN) to account for the diffusion velocities in a similar manner.

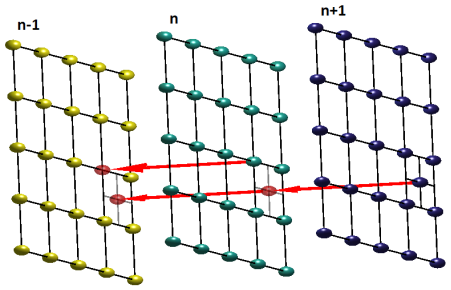

Figure 26: The backwards difference formulation for the time derivative requires grid points to exist at a particular location in space and at each time step used in the time derivative approximation. The grid is adapting at every time step throughout the solution process, and these grid points may or may not be present at previous times. If they are missing, they need to be wavelet-interpolated, within the user-specified tolerance, from neighboring data at the same time step. This figure illustrates the data dependency on previous grids and solutions, and the red points indicate possible missing points in the BDF approximation.

The pseudo-time integration loop, also known as the *inner loop*, uses an explicit

time integrator. Alternative methods for inner loop integration have been investigated by Oefelein (1997) [82], and Jameson (1991) [54], but in the context of GPU-based solvers explicit formulations are preferred over implicit for computational performance as discussed above. A 4^{th}-order Runge-Kutta scheme is preferred for the inner loop due to its compromise of storage requirements, accuracy, and stability for larger step sizes, but lower order Runge-Kutta schemes work as well. The stability region of some Runge-Kutta methods are shown in Figure 27 and the 4^{th}-order Runge-Kutta stages and Butcher coefficients are shown in Equation 29. Note that the stability range of the 4^{th}-order Runge-Kutta scheme allows for CFL numbers of greater than unity, which is acceptable for the inner loop since only the steady-state condition of the pseudo-time system is of interest. It is also possible to increase the buffer region around essential points in the wavelet grid to allow for larger-than-unity CFL pseudo-time steps. While this would help to reduce the number of inner loop iterations, the cost (*i.e.*, memory requirements) of doing so is prohibitively expensive.

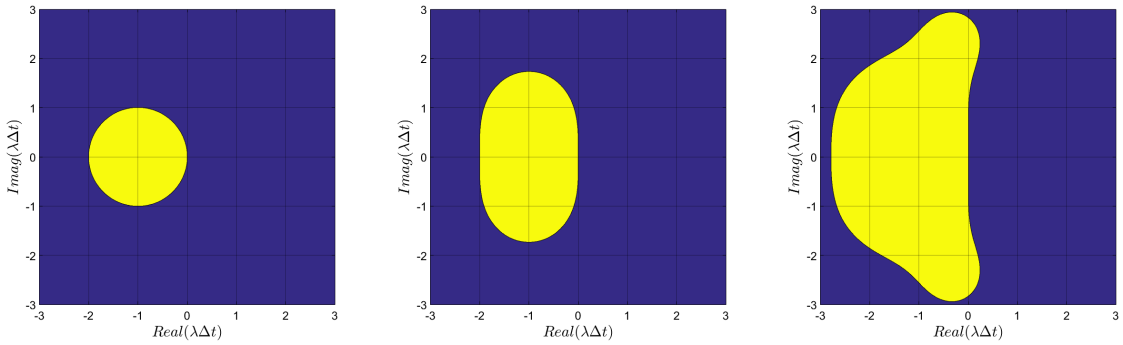

Figure 27: The absolute stability regions are shown for the (left) 1^{st}-order Runge-Kutta method, (middle) 2^{nd}-order Runge-Kutta method, and (right) 4^{th}-order Runge-Kutta method. The stable region is highlighted in yellow. The 4^{th}-order Runge-Kutta scheme provides the largest absolute stability region and the highest accuracy of the methods considered here.

$$
\begin{aligned}
\boldsymbol{Q}_0 &= \boldsymbol{Q}^m \\
\boldsymbol{Q}_1 &= \boldsymbol{Q}_0 + \frac{1}{2}\Delta\tau L(\boldsymbol{Q}_0) \\
\boldsymbol{Q}_2 &= \boldsymbol{Q}_0 + \frac{1}{2}\Delta\tau L(\boldsymbol{Q}_1) \\
\boldsymbol{Q}_3 &= \boldsymbol{Q}_0 + \Delta\tau L(\boldsymbol{Q}_2) \\
\boldsymbol{Q}^{m+1} &= \boldsymbol{Q}_0 + \frac{1}{6}\Delta\tau L(\boldsymbol{Q}_0) + \frac{1}{3}\Delta\tau L(\boldsymbol{Q}_1) + \frac{1}{3}\Delta\tau L(\boldsymbol{Q}_2) + \frac{1}{6}\Delta\tau L(\boldsymbol{Q}_3)
\end{aligned}
\tag{29}
$$

The superscript n is reserved for physical time step indexing (see Equation 28) while the superscript m is reserved for pseudo-time step indexing (see Equation 29). The $\boldsymbol{U}^{n+1}$ solution used in the construction of $\boldsymbol{Q}^m$ at the beginning of each outer loop (*i.e.*, physical time step) is initialized by simply copying the $\boldsymbol{U}^n$ solution. Using a linear extrapolation from the $\boldsymbol{U}^n$ and $\boldsymbol{U}^{n-1}$ solutions was investigated, but it was found that negative densities and mass fractions were sometimes estimated, depending on the solution and the time derivatives of the solution vector. Copying the previous solution proved to be more robust and did not measurably affect the average number of iterations compared to the linear extrapolation.

4.3.3 Preconditioning

Following the work of Oefelein (1997) [82], a single parameter preconditioner is used for real fluids assuming a generalized compressibility model. This allows for optimizing the preconditioning for a variety of simulation conditions and a wide range of working fluids. The preconditioner from Oefelein (1997) [82] is presented in Equation 30. In this work, an analytical inverse preconditioning matrix was derived for an arbitrary number of species, as shown in Equation 32. This explicit form of the inverse preconditioning matrix avoids the need to numerically invert the non-symmetric, non-positive definite, non-banded, preconditioning matrix for general systems with varying numbers of species. The result is an improvement in the accuracy, robustness, and computational efficiency of the algorithm, especially with an increasing number of

species.

The impact of this result is significant for the GPU performance since a generalized matrix inversion routine would require too many registers (cache) to allow full occupancy since each grid point requires a matrix inversion. This can be demonstrated by considering two examples, one with two-component 3-dimensional flow and the other being the reacting 3-dimensional flow of hydrogen and oxygen with 9 species [Oefelein (1997) [82]]. These systems of equations would require a 6x6 matrix inverse (36 elements) and a 13x13 matrix inverse (169 elements), respectively. Each thread on current GPUs can only store 16 double-precision numbers in registers before reduced processor occupancy is required due to resource constraints, which is far less than either example matrix size. With the analytical inverse shown in Equation 32, only the non-zero entries in the matrix need to be calculated in the preconditioning function, and these can be done in stages to further reduce resource requirements for improved GPU performance. Since preconditioning is a completely local operation (*i.e.*, not requiring information from neighboring grid points), the order that the grid points are processed does not matter, so the data access is completely coalesced for maximum performance. The result is that the preconditioning accounts for less than 1% of the total wall time per physical simulation time step, and the computational performance of the explicit time stepping that is used in the inner loop remains unchanged. The overall performance of the method will be discussed in the next section.

$$\mathbf{\Gamma} = \begin{bmatrix} \frac{\gamma}{\beta} & 0 & 0 & 0 & -\frac{\rho}{T}\mathcal{Z}_T & -\rho R_1 & \cdots & -\rho R_{N-1} \\ \frac{\gamma u}{\beta} & \rho & 0 & 0 & -\frac{\rho u}{T}\mathcal{Z}_T & -\rho u R_1 & \cdots & -\rho u R_{N-1} \\ \frac{\gamma v}{\beta} & 0 & \rho & 0 & -\frac{\rho v}{T}\mathcal{Z}_T & -\rho v R_1 & \cdots & -\rho v R_{N-1} \\ \frac{\gamma w}{\beta} & 0 & 0 & \rho & -\frac{\rho w}{T}\mathcal{Z}_T & -\rho w R_1 & \cdots & -\rho w R_{N-1} \\ \frac{\gamma h_t}{\beta} - \mathcal{Z}_T & \rho u & \rho v & \rho w & -\frac{\rho h_t}{T}\mathcal{Z}_T + \rho c_p & -\rho h_t R_1 + \rho H_1 & \cdots & -\rho h_t R_{N-1} + \rho H_{N-1} \\ \frac{\gamma Y_1}{\beta} & 0 & 0 & 0 & -\frac{\rho Y_1}{T}\mathcal{Z}_T & -\rho Y_1 R_1 + \rho & \cdots & -\rho Y_1 R_{N-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{\gamma Y_{N-1}}{\beta} & 0 & 0 & 0 & -\frac{\rho Y_{N-1}}{T}\mathcal{Z}_T & -\rho Y_{N-1} R_1 + \rho & \cdots & -\rho Y_{N-1} R_{N-1} \end{bmatrix} \tag{30}$$

Here, γ is the ratio of specific heats, h_t is the total enthalpy, c_p is the specific heat capacity at constant pressure, Z is the compressibility factor, R is the gas constant, and the remaining variables are defined in Equation 31.

$$\begin{aligned} \beta &= \frac{\gamma \epsilon c^2}{1 + (\gamma - 1)\epsilon} \\ \mathcal{Z}_T &= \left[1 + \frac{T}{Z} \left(\frac{\partial Z}{\partial T} \right)_p \right] \\ \mathcal{R}_k &= \frac{R_k - R_N}{R} \left[1 + \frac{R}{(R_k - R)Z} \left(\frac{\partial Z}{\partial Y_k} \right)_{p,T,Y_j,\ldots(j \neq k)} \right] \\ R_k &= \frac{Z R_u}{W_k} \\ R &= \sum_{i=1}^{N} R_k Y_k \\ H_k &= h_k - h_N \end{aligned} \tag{31}$$

$$\mathbf{\Gamma}^{-1} =$$

$$\begin{bmatrix}
\frac{-h_t+U^2+\frac{c_pT}{\mathcal{Z}_T}+\sum_{i=1}^{n-1}H_iY_i-\frac{c_pT}{\mathcal{Z}_T}\sum_{i=1}^{n-1}R_iY_i}{c_pT-\frac{\beta}{\gamma}\mathcal{Z}_T^2}\frac{\beta}{\gamma}\mathcal{Z}_T & \frac{-u}{c_pT-\frac{\beta}{\gamma}\mathcal{Z}_T^2}\frac{\beta}{\gamma}\mathcal{Z}_T & \frac{-v}{c_pT-\frac{\beta}{\gamma}\mathcal{Z}_T^2}\frac{\beta}{\gamma}\mathcal{Z}_T & \frac{-w}{c_pT-\frac{\beta}{\gamma}\mathcal{Z}_T^2}\frac{\beta}{\gamma}\mathcal{Z}_T & \frac{1}{c_pT-\frac{\beta}{\gamma}\mathcal{Z}_T^2}\frac{\beta}{\gamma}\mathcal{Z}_T & \frac{-H_1+\frac{c_pT}{\mathcal{Z}_T}R_1}{c_pT-\frac{\beta}{\gamma}\mathcal{Z}_T^2}\frac{\beta}{\gamma}\mathcal{Z}_T & \dots & \frac{-H_{n-1}+\frac{c_pT}{\mathcal{Z}_T}R_{n-1}}{c_pT-\frac{\beta}{\gamma}\mathcal{Z}_T^2}\frac{\beta}{\gamma}\mathcal{Z}_T \\
-\frac{u}{\rho} & \frac{1}{\rho} & 0 & 0 & 0 & 0 & \dots & 0 \\
-\frac{v}{\rho} & 0 & \frac{1}{\rho} & 0 & 0 & 0 & \dots & 0 \\
-\frac{w}{\rho} & 0 & 0 & \frac{1}{\rho} & 0 & 0 & \dots & 0 \\
\frac{-h_t+U^2+\frac{\beta}{\gamma}\mathcal{Z}_T+\sum_{i=1}^{n-1}H_iY_i-\frac{\beta}{\gamma}\mathcal{Z}_T\sum_{i=1}^{n-1}R_iY_i}{\rho(c_pT-\frac{\beta}{\gamma}\mathcal{Z}_T^2)}T & \frac{-u}{\rho(c_pT-\frac{\beta}{\gamma}\mathcal{Z}_T^2)}T & \frac{-v}{\rho(c_pT-\frac{\beta}{\gamma}\mathcal{Z}_T^2)}T & \frac{-w}{\rho(c_pT-\frac{\beta}{\gamma}\mathcal{Z}_T^2)}T & \frac{1}{\rho(c_pT-\frac{\beta}{\gamma}\mathcal{Z}_T^2)}T & \frac{-H_1+R_1\frac{\beta}{\gamma}T}{c_pT-\frac{\beta}{\gamma}\mathcal{Z}_T^2}T & \dots & \frac{-H_{n-1}+R_{n-1}\frac{\beta}{\gamma}T}{c_pT-\frac{\beta}{\gamma}\mathcal{Z}_T^2}T \\
-\frac{Y_1}{\rho} & 0 & 0 & 0 & 0 & \frac{1}{\rho} & \dots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
-\frac{Y_{n-1}}{\rho} & 0 & 0 & 0 & 0 & 0 & \cdots & \frac{1}{\rho}
\end{bmatrix} \tag{32}$$

The term β/γ that appears in Equation 30 and Equation 32 allows for the modified propagation speed of acoustic waves in the pseudo-time system to reduce the stiffness at low Mach numbers. The term ϵ is varied based on local conditions to minimize the disparities in the characteristics of the governing equations and bring the condition number of the modified Jacobian matrix as close to unity as possible. The stiffness is considered to be completely removed (*i.e.*, all characteristics propagate at the same speed) when the condition number is unity, which means a single global time step is sufficient for all of the characteristics. In the case of $Ma = 1$ flow, ϵ evaluates to unity, and the preconditioning matrix simplifies to the unmodified transformation matrix that simply converts between primitive and conserved variables. In the case of low Mach number flows and a physical CFL time that is large with respect to the acoustic time scale, ϵ is based on a few criteria relating to the local Mach number, the unsteadiness of the problem, and the relative importance of viscous effects (*i.e.*, the resulting diffusion velocities relative to the acoustic and convective propagation speeds). The choice of ϵ requires a careful analysis of the eigenvalues of the system of governing equations to design the preconditioning factors, shown in Equation 33, to optimize the condition number at all flow conditions. This work has been carried out in detail by Buelow (1995) [16] and Oefelein (1997) [82]. The choice of ϵ in this work is based on the results in Oefelein (1997) [82] and summarized in Equation 33.

$$
\begin{aligned}
\epsilon_{inviscid} &= \begin{cases} \epsilon^2 & : \quad M \leq \epsilon_{min} \\ 2M^2 & : \quad \epsilon_{min} < M < 1 \\ 1 & : \quad M \geq 1 \end{cases} \\
\epsilon_{unsteady} &= \frac{1}{c^2}\left[\left(\frac{L_{x1}}{\Delta t \pi}\right)^2 + \left(\frac{L_{x2}}{\Delta t \pi}\right)^2\right] + M^2 \\
\epsilon_{viscous} &= max\left(\frac{u^2\delta_{x1}(\delta_{x1}-1)}{u^2(\delta_{x1}-1)+c^2},\ \frac{v^2\delta_{x2}(\delta_{x2}-1)}{v^2(\delta_{x2}-1)+c^2}\right) \\
\epsilon &= min\left(1.0,\ max\left(\epsilon_{inviscid},\ max\left(\epsilon_{unsteady},\ \epsilon_{viscous}\right)\right)\right)
\end{aligned}
\tag{33}
$$

$$
\begin{aligned}
\delta_{x1} &= max\left(\nu, \frac{\nu}{Pr}, \frac{\nu}{Sc_k}\right)\frac{1}{u}\frac{CFL}{VNN} \\
\delta_{x2} &= max\left(\nu, \frac{\nu}{Pr}, \frac{\nu}{Sc_k}\right)\frac{1}{v}\frac{CFL}{VNN}
\end{aligned}
$$

Here, ν is the kinematic viscosity, Pr is the Prandtl number relating heat and momentum diffusion, Sc_k is the Schmidt number for the k^{th}-species that relates the mass diffusion to the momentum diffusion. ϵ_{min} is typically restricted to a minimum of $10^{-8} \leq \epsilon_{min} \leq 10^{-5}$ to prevent dividing by zero near stagnation points. This choice does not affect the physical solution, but it may have an influence on the convergence rate of the inner loop at the slowest moving parts of the flow field. These preconditioning factor criteria account for the effects of momentum, energy, and mass diffusion processes on the overall convergence rate.

4.3.4 Performance

The overall performance of the preconditioned dual-time stepping method is strongly dependent on the problem being solved, the quantities of interest, and the required tolerance on the residuals. For time-accurate simulations, it has been the author's experience that the reduction in physical time steps through larger CFL numbers is generally accompanied by an increase in the number of inner-loop iterations required

for convergence of the residuals to some extent. The improvement in time-to-solution generally increases as the convective speeds decrease. It should be noted that physical diffusive effects are relatively small in the applications considered here, and the preconditioning can have a large effect in reacting flow where very large gradients in species concentration (*i.e.*, large diffusion velocities) can be generated. The tolerances held for high accuracy even in the very low-speed regions of the flow in this work have an impact on the number of inner-loop iterations that may be larger than other use cases for dual-time stepping seen in the literature. Like other non-linear solvers, it is expected that the iteration count for convergence will increase as tolerances are made more stringent. The preconditioned dual-time stepping method is always much faster in finding steady-state solutions than the explicit compressible solver. Additionally, the direct comparison of the implicit dual-time stepping method with an explicit integration method is an "apples-to-oranges" comparison since the preconditioned dual-time stepping method can obtain highly accurate solutions to even the lowest speed flows (*e.g.*, $Ma\ 10^{-8}$). Such solutions are simply not possible to find using an explicit time integrator for the compressible form of the Navier-Stokes equations due to the pressure singularity problem as the Mach number approaches zero.

The physical time derivative calculation and the preconditioning step accounts for less than 1% of the total wall time per physical simulation time step, and the computational performance of the explicit time stepping that is used in the inner loop has remained unchanged. Thus, the overall computational expense of the preconditioned dual-time stepping method is competitive with the explicit schemes introduced in Chapter 3 of this thesis. This removes the bottleneck from the computing architecture that is commonly seen with implicit methods on parallel computing architectures. The overall computational expense of the implicit dual-time stepping method relative to an explicit integration is effectively a function of the problem-dependent number of inner-loop iterations required for convergence. With this in mind, the cost of

the preconditioned dual-time stepping method relative to explicit integrators can be characterized by Equation 34.

$$\Delta t_{wall,implicit} = \Delta t_{wall,explicit} \left(\frac{CFL_{explicit}}{CFL_{implicit}} \right) N_{inner} \tag{34}$$

Here, $\Delta t_{wall,implicit}$ is the wall time required to solve a given problem to the final simulation time by the implicit, preconditioned dual-time stepping method, $\Delta t_{wall,explicit}$ is the wall time required to solve a given problem to the final simulation time by an explicit time integrator, $CFL_{explicit}$ is the maximum acoustic CFL number allowed by the explicit time integrator stability, $CFL_{implicit}$ is the CFL number chosen to be unity for the time scale of interest (*e.g.*, acoustic, convective, or diffusive scales) for time-dependent problem or very large (*e.g.*, $CFL \geq 1000$) for steady-state problems, and N_{inner} is the number of inner loop iterations required to reach convergence.

4.4 Verification and Validation

Chapter 3 of this thesis provided verification of the accuracy of the wavelet reconstruction of a function at any point in the domain to the user-prescribed tolerance, convergence of the derivative operators, and the solution of high-speed, viscous, multispecies compressible flow test cases. This chapter provides verification for the solution of the compressible Navier-Stokes equations in the low-Mach number regime using dual-time stepping and preconditioning. This builds towards the full verification and validation of an all-Mach number compressible flow solver that will be applied to low-speed multiphase flows and presented in Chapter 5 of this thesis.

The lid-driven cavity flow demonstrates that the compressible Navier-Stokes equations can be solved accurately at very low Mach numbers (*i.e.*, nearly incompressible flow), the isentropic vortex test case highlights the dissipation properties of the numerical method since there is no physical dissipation, and the Taylor-Green vortex demonstrates that the solver can be applied to fully resolve complex flow problems

that are sensitive to the accuracy and dissipation properties of the method.

The isentropic vortex test case highlights the dissipation properties of the numerical method since there is no physical dissipation, the lid-driven cavity flow demonstrates that the compressible Navier-Stokes equations can be solved accurately at very low Mach numbers (*i.e.*, nearly incompressible flow), and the Taylor-Green vortex demonstrates that the solver can be applied to fully resolve complex flow problems that are sensitive to the accuracy and dissipation properties of the method.

4.4.1 Lid-Driven Cavity

A common numerical benchmark to demonstrate the accuracy of an incompressible flow solver is the flow inside a rectangular cavity with a sliding lid. A simplified schematic of this problem configuration is shown in Figure 28. While this test problem has been solved and published by numerous authors, the present work will be compared to the original finite difference computations of Ghia et al. (1982) [44] and the more accurate spectral results from Botella and Peyret [11]. Note that Ghia et al. (1982) [44] and Botella and Peyret (1998) [11] solved the incompressible form of the Navier-Stokes equations, while the present work solves the fully compressible form of the equations with a lid speed that corresponds to a Mach number of 0.001 in order to limit the effects of compressibility. The equations of motion for the flow are discretized using 4^{th}-order accurate finite differences over the interior and at the boundaries for all operators. The finest grid resolution is limited to 287^2 to prevent excessive refinement near the corner singularities and the relative wavelet tolerance is held to 10^{-8}.

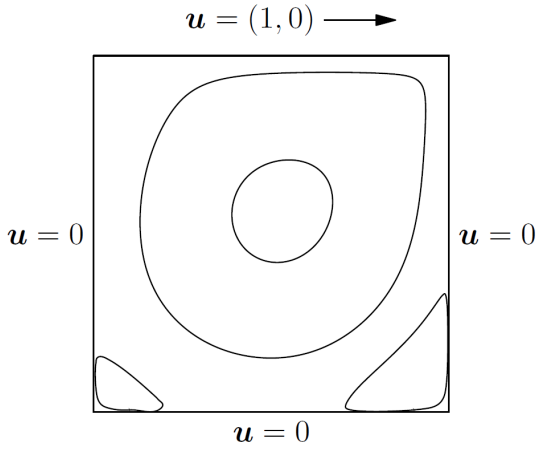

Figure 28: An idealized configuration of a 2D lid-driven cavity flow. The velocity conditions are provided in the diagram, and all of the walls are treated as adiabatic.

The present results for contours of velocity magnitude with a lid Reynolds number $Re = 1000$ are shown in Figure 29. The expected flow pattern with a primary clockwise vortex and the two secondary corner vortices are displayed. The Mach number of the flow throughout the domain is in the range $0 \leq Ma \leq 0.001$, which demonstrates the ability of the present preconditioned dual-time stepping method to accurately solve for very low-Mach number flows.

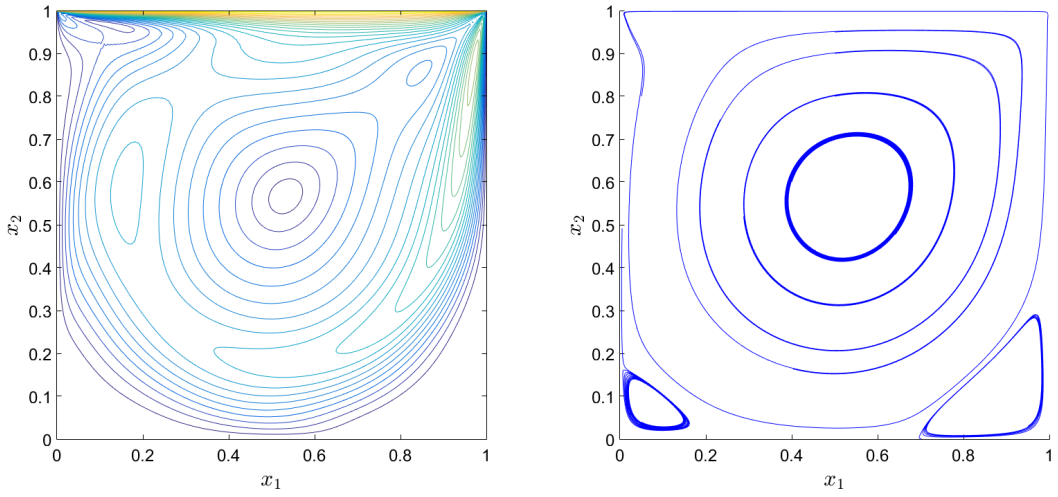

Figure 29: (Left) Velocity magnitude contours for $Re = 1000$ and $Ma = 0.001$. (Right) Selected streamlines to show primary and secondary vortices.

The results for $Re = 1000$ from Ghia et al. (1982) [44], Botella and Peyret (1998) [11], and the present work are shown in Figure 30 and tabulated in Tables 4 and 5. All of the solutions are in close agreement, although there are some small differences between the two benchmark solutions and the present work. The average relative difference in the horizontal and vertical velocities along the two centerlines is about 3%, with a maximum difference of 9% near the bottom wall, and 6% near the left and right side walls. Differences of this size are expected due to the different numerical methods used, the special treatment of the singularities in Botella and Peyret (1998) [11], and mostly because the present work solves the *compressible* Navier-Stokes equations. A compressible flow will behave slightly differently compared to an incompressible flow near the corner lid singularities. In particular, near the upper-right corner where the moving lid meets the stationary side wall the fluid density will increase and the resulting wall jet will behave differently compared to the incompressible case. This slight difference will also influence the resulting flow field in the main body of the cavity, especially near the side and bottom walls as seen here.

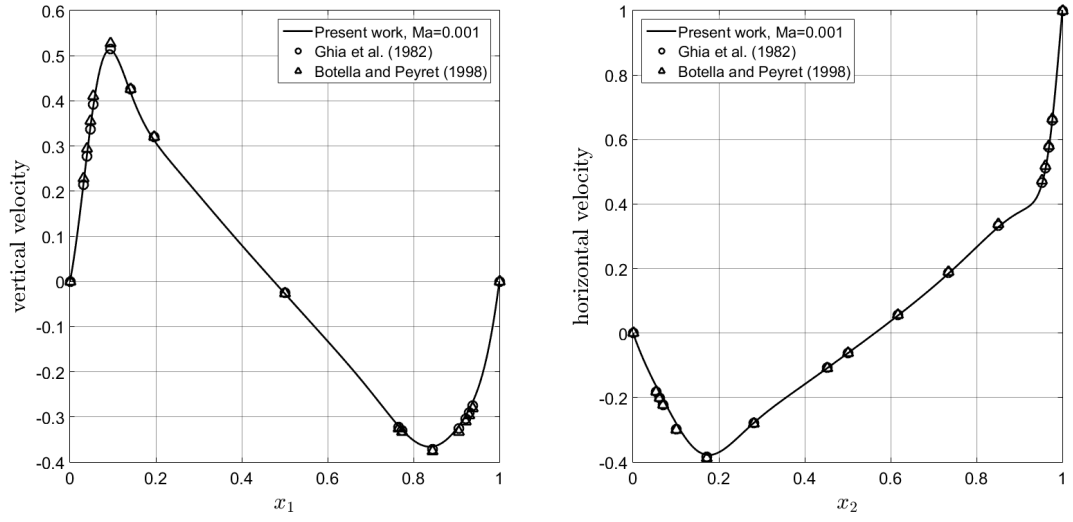

Figure 30: Velocities in a 2D lid-driven cavity with $Re = 1000$ and $Ma = 0.001$ based on the lid velocity. (Left) Vertical velocity along the horizontal centerline. (Right) Horizontal velocity along the vertical centerline.

Table 4: Tabulated vertical velocity along the cavity's horizontal centerline for a a 2D lid-driven cavity with $Re = 1000$ and $Ma = 0.001$ based on the lid velocity.

x_1	v_{Ghia}	$v_{Botella}$	$v_{wavelet}$
0	0	0	0
0.0312	0.21388	0.2279225	0.2148416
0.0391	0.27669	0.2936869	0.2777784
0.0469	0.33714	0.3553213	0.3373287
0.0547	0.39188	0.4103754	0.3911639
0.0937	0.51550	0.5264392	0.5124237
0.1406	0.42665	0.4264545	0.4198815
0.1953	0.31966	0.3202137	0.3134421
0.5000	-0.02526	-0.0257995	-0.0265117
0.7656	-0.32235	-0.3253592	-0.3205919
0.7734	-0.33075	-0.3339924	-0.3287515
0.8437	-0.37095	-0.3769189	-0.3657670
0.9062	-0.32627	-0.3330442	-0.3191815
0.9219	-0.30353	-0.3099097	-0.2962649
0.9297	-0.29012	-0.2962703	-0.2828520
0.9375	-0.27485	-0.2807056	-0.2676048
1	0	0	0

Table 5: Tabulated horizontal velocity along the cavity's vertical centerline for a a 2D lid-driven cavity with $Re = 1000$ and $Ma = 0.001$ based on the lid velocity.

x_2	u_{Ghia}	$u_{Botella}$	$u_{wavelet}$
1	1	1	1
0.9766	0.65928	0.6644227	0.6602701
0.9688	0.57492	0.5808359	0.5756969
0.9609	0.51117	0.5169277	0.5108458
0.9531	0.46604	0.4723329	0.4653779
0.8516	0.33304	0.3372212	0.3294523
0.7344	0.18719	0.1886747	0.1842142
0.6172	0.05702	0.0570178	0.0546180
0.5000	-0.06080	-0.0620561	-0.0625550
0.4531	-0.10648	-0.1081999	-0.1079387
0.2813	-0.27805	-0.2803696	-0.2786154
0.1719	-0.38289	-0.3885691	-0.3777638
0.1016	-0.29730	-0.3004561	-0.2821754
0.0703	-0.22220	-0.2228955	-0.2064348
0.0625	-0.20196	-0.2023300	-0.1867353
0.0547	-0.18109	-0.1812881	-0.1667220
0	0	0	0

The lid-driven cavity was chosen as a test problem because it is a popular benchmark flow and it clearly demonstrates the ability of the preconditioned dual-time stepping method to solve for an incompressible flow. However, the advantage of having a wavelet adaptive grid was negligible for this particular problem. This is largely due to the ill-posed velocity boundary conditions on the lid, and secondarily due to the relatively small grid dimensions that were required to obtain an accurate solution;

a grid size that did not leave much room for grid coarsening. The ill-posed boundary conditions in this problem produce singularities in each of the upper corners where the moving lid and the stationary wall velocity conditions are in conflict. When the lid is impulsively started from rest at $t = 0$, relatively large pressure pulses emanate from the lid and the top corners. The physical CFL number was chosen to be large to reach steady-state more quickly, and the dissipative properties of the BDF time integration scheme helped to eliminate these high-wavenumber initial transients. However, before that occurred these initial transient pressure pulses were reflected across the cavity many times and temporarily required grid refinement nearly everywhere in the domain. This behavior is typically not an issue for compressible flow solvers in well-posed problems with consistent boundary and initial conditions; however, it can be a problem in some cases (like the lid-driven cavity problem) because unlike the incompressible Navier-Stokes equations, acoustics are permitted in these compressible flow solutions.

The next two test cases in this work will clearly demonstrate both the effectiveness of wavelet-adaptive grids combined with the preconditioned dual-time stepping method and the accuracy of the flow solver.

4.4.2 Isentropic Vortex Transport by a Uniform Flow

The problem of a translating isentropic vortex in an *inviscid* fluid is a convenient test case to measure the artificial dissipation present in a numerical scheme. The vortex remains unchanged during the flow, aside from a pure translation, and so the final vortex can be directly compared to the initial vortex. A survey of several vortex test configurations can be found in Spiegel et al. (2015) [105]. The test case used here is from Wang et al. (2013) [123]. This particular case uses the lowest Mach number out of all the surveyed cases, thus it is the most demanding test for an all-Mach number compressible flow solver. The flow configuration is a rectangular domain, as shown

in the schematic in Figure 31. The fluid is air.

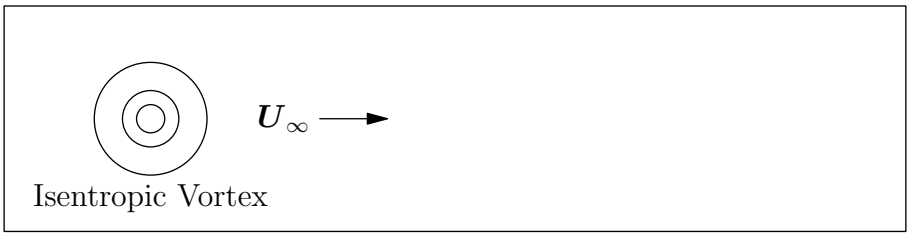

Figure 31: Vortex configuration in a uniform flow field of air.

The initial conditions for the flow are provided in Equation 35.

$$
\begin{aligned}
u &= U_\infty + \delta u \\
v &= \delta v \\
T &= T_\infty - \delta T \\
P &= \rho R_{gas} T \\
\rho &= \rho_\infty \left(\frac{T}{T_\infty} \right)^{1/(\gamma-1)} \\
M_\infty &= \frac{U_\infty}{\sqrt{\gamma R_{gas} T_\infty}} \\
\delta u &= -(U_\infty \beta) \frac{x_2 - x_{2c}}{R} \exp\left(\frac{-r^2}{2} \right) \\
\delta v &= (U_\infty \beta) \frac{x_1 - x_{1c}}{R} \exp\left(\frac{-r^2}{2} \right) \\
\delta T &= \frac{1}{2c_p} (U_\infty \beta)^2 \exp(-r^2) \\
r &= \frac{\sqrt{(x_1 - x_{1c})^2 + (x_2 - x_{2c})^2}}{R}
\end{aligned}
\tag{35}
$$

Here, u is the local velocity in the x_1-direction, v is the local velocity in the x_2-direction, U_∞ is the freestream velocity in the x_1-direction, T is the temperature, P is the pressure, ρ is the density, γ is the ratio of specific heats, and c_p is the specific heat capacity at constant pressure. Quantities with an infinity subscript are free stream quantities and M_∞ is the freestream Mach number. The quantities δu, δv, and δT are the velocities and temperature related to the isentropic vortex flow, x_{1c} is the vortex center in the x_1-direction, x_{2c} is the vortex center in the x_2-direction, r is the radial

position from the vortex center, R is the characteristic length scale of the vortex, and β is the vortex strength.

Air is modeled as an ideal gas with $\gamma = 1.4$, $R_{gas} = 287.15$ J/kg-K, and $c_p = \gamma/(\gamma-1)R_{gas}$. The rectangular computational domain has $(x_1, x_2) \in [0,\ 16L] \times [0,\ L]$, with $L = 0.1$. The vortex is located at $(x_{1c}, x_{2c}) = (0.05,\ 0.05)$ with a size $R = 0.005$ and strength $\beta = 1/50$. The freestream flow conditions are set at $P_\infty = 105$ N/m^2, $T_\infty = 300$ K, and $M_\infty = 0.05$.

The work reported by Wang et al. (2013) [123] used periodic boundary conditions from the left to right walls and from the top to bottom walls. In the present work, the top and bottom walls are rigid, the left boundary is an inlet with the freestream conditions, and the right is prescribed as an outflow boundary. The domain size is very large compared to the size of the vortex and this change in boundary conditions does not affect the flow. The equations are discretized using 4^{th}-order accurate finite differences over the interior and at the boundaries for all operators. The relative wavelet tolerance is held to 10^{-8}.

The time period flow this flow as defined by Wang et al. (2013) [123] is $t_s = L/U_\infty$. The test case was run for a time of $73.5t_s$. The duration of the simulation was simply limited by the available computing resources; otherwise, the simulation could have been run for a longer time period. The global kinetic energy conservation error increased linearly over the duration of the simulation for a maximum relative error of 1.1×10^{-3}. The solution in the present work is inviscid, and there is no artificial dissipation added to the central difference scheme and the time step was chosen with a physical CFL of unity that also provides almost no numerical diffusion in this case. This combined with very slight indications of the long-term instability described in Wang et al. (2013) [123] may explain the very small increase in the global kinetic energy over the long duration of the simulation. The vortex shape is well-preserved, even at late times. The contour plots of the kinetic energy at the initial and final

times are shown side-by-side in Figure 32. The error levels are low in comparison with the high-order methods in the literature surveyed by Spiegel et al. (2015) [105].

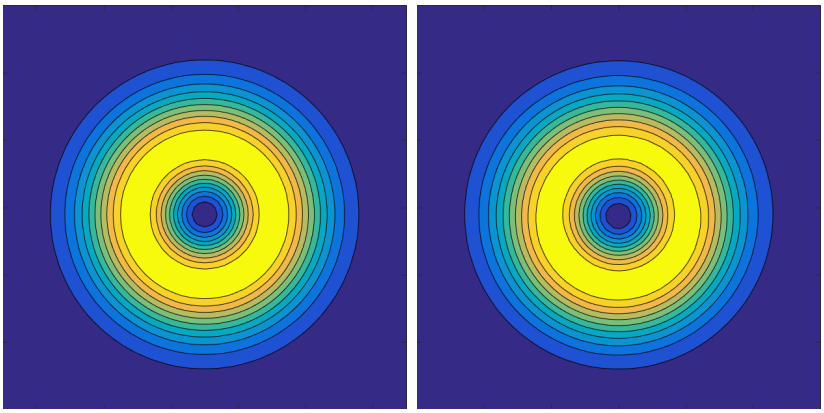

Figure 32: The kinetic energy contours of the initial (left) and final (right) states are shown with the freestream velocity component removed from the calculations to show the vortex structure more clearly. There is a very small visible difference between the two, which appears as a slight misalignment of the contours in the vertical direction.

4.4.3 Taylor-Green Vortex

The nearly incompressible Taylor-Green vortex is a popular turbulent cascade model that tests a simulation's ability to capture vortex dynamics, vortex decay, and a transition to turbulence [Taylor and Green (1937) [110], Boyd (2001) [12], DeBonis (2013) [27]]. The problem is initialized with a periodic set of large vortices in the domain that contain a prescribed amount of kinetic energy. Since there is no external forcing on the system the flow will eventually dissipate all of the initial kinetic energy to reach a final quiescent steady state. The flow field evolves with the larger vortices interacting, rolling up, stretching, and finally breaking down into turbulence. The size and structure of the vortices depends on the Reynolds number and tends to create smaller scales as the Reynolds number is increased.

The present test case is run at a Reynolds number of 400 and Mach number of 0.1 with the initial conditions shown in Equation 36. The adaptive grid requires only approximately 9% of the points that would be required with a uniform grid with an

equivalent peak resolution of 481^3. The relative wavelet tolerance is prescribed at 10^{-4}. The symmetry of the problem is utilized to reduce the computational expense by simulating one octant of the domain, for a domain size of $[0, \pi]^3$. The equations were discretized using 4^{th}-order accurate finite differences over the interior and at the boundaries for all operators.

$$\begin{aligned} u(\boldsymbol{x}, t_0) &= \sin\left(\frac{x_1}{L}\right)\cos\left(\frac{x_2}{L}\right)\cos\left(\frac{x_3}{L}\right) \\ v(\boldsymbol{x}, t_0) &= -\cos\left(\frac{x_1}{L}\right)\sin\left(\frac{x_2}{L}\right)\cos\left(\frac{x_3}{L}\right) \\ w(\boldsymbol{x}, t_0) &= 0 \\ p(\boldsymbol{x}, t_0) &= p_0 + \frac{\rho}{16}\left[\cos\left(2\frac{x_1}{L}\right) + \cos\left(2\frac{x_2}{L}\right)\right]\left[\cos\left(2\frac{x_3}{L}\right) + 2\right] \end{aligned} \tag{36}$$

The kinetic energy dissipation rate and iso-surfaces of vorticity magnitude have been provided in previous studies by other researchers at a Reynolds number of 400 [Brachet et al. (1983) [15], Yang and Pullin (2011) [128]]. Results have been reported from both compressible and incompressible simulations, and they compare well due to the low Mach number of $Ma = 0.1$. The results from Yang and Pullin (2011) [128] were found using a pseudo-spectral simulation on a uniform grid with a 16^{th}-order exponential filter to dealias the solution. This numerical approach was applied to solving a Lagrangian-like formulation of the flow to evolve a Vortex-Surface Field (VSF). Owing to the fast convergence rates of the pseudo-spectral method, the selective (high-order) dealiasing filter, and the VSF formulation, their results provide a highly accurate, fully resolved baseline that is considered the reference solution for this work.

The kinetic energy dissipation rate versus time and a volume rendering of kinetic energy values near the peak dissipation rate are shown in Figure 33. The results are in close agreement with those of Yang and Pullin (2011) [128]. As pointed out by Shu et al. (2005) [101], it is important to realize that agreement in integrated global quantities can be misleading since these quantities do not assess local error and may

appear to be close even though the flow is under-resolved. This point is addressed in the present work because the wavelet transform quantifies the local error at every time step, thus ensuring that local accuracy is maintained in addition to the global accuracy assessed by the time trace of the kinetic energy dissipation rate shown in Figure 33.

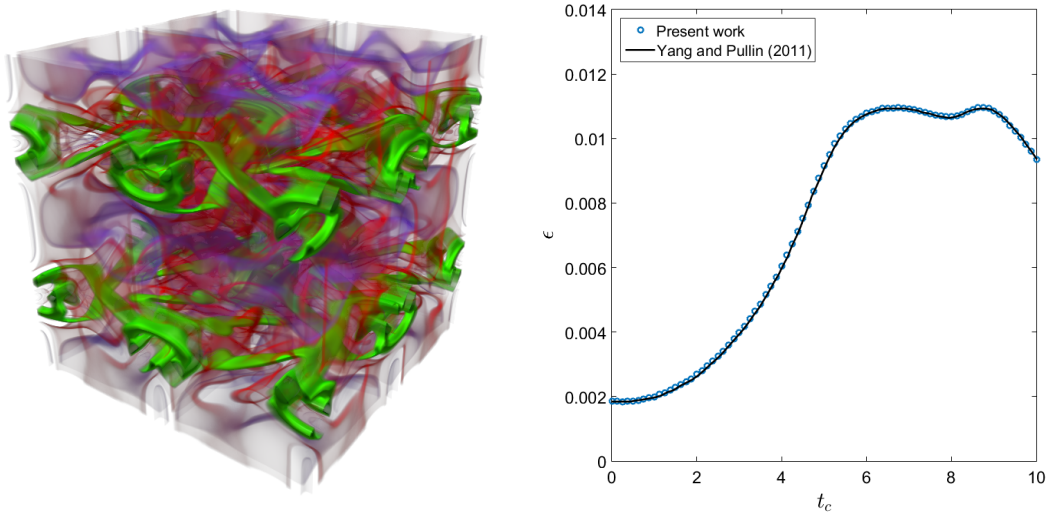

Figure 33: The Taylor-Green vortex at $Re = 400$ and $Ma = 0.1$. (Left) The kinetic energy dissipation rate ϵ as a function of time scaled on the convective time scale t. (Right) The three-dimensional volumetric field of kinetic energy near the peak kinetic energy dissipation rate ($t = 7$). The Voreen framework is used to generate the volume renderings presented in this work [Meyer-Spradow et al. (2009) [75]].

The evolution of the vorticity magnitude iso-surfaces are reported in Figure 34. The instances in time and iso-surface values are chosen for direct comparison to Yang and Pullin (2011) [128]. Both the kinetic energy dissipation rate and the vorticity magnitude iso-surfaces agree closely with the results from Yang and Pullin (2011) [128].

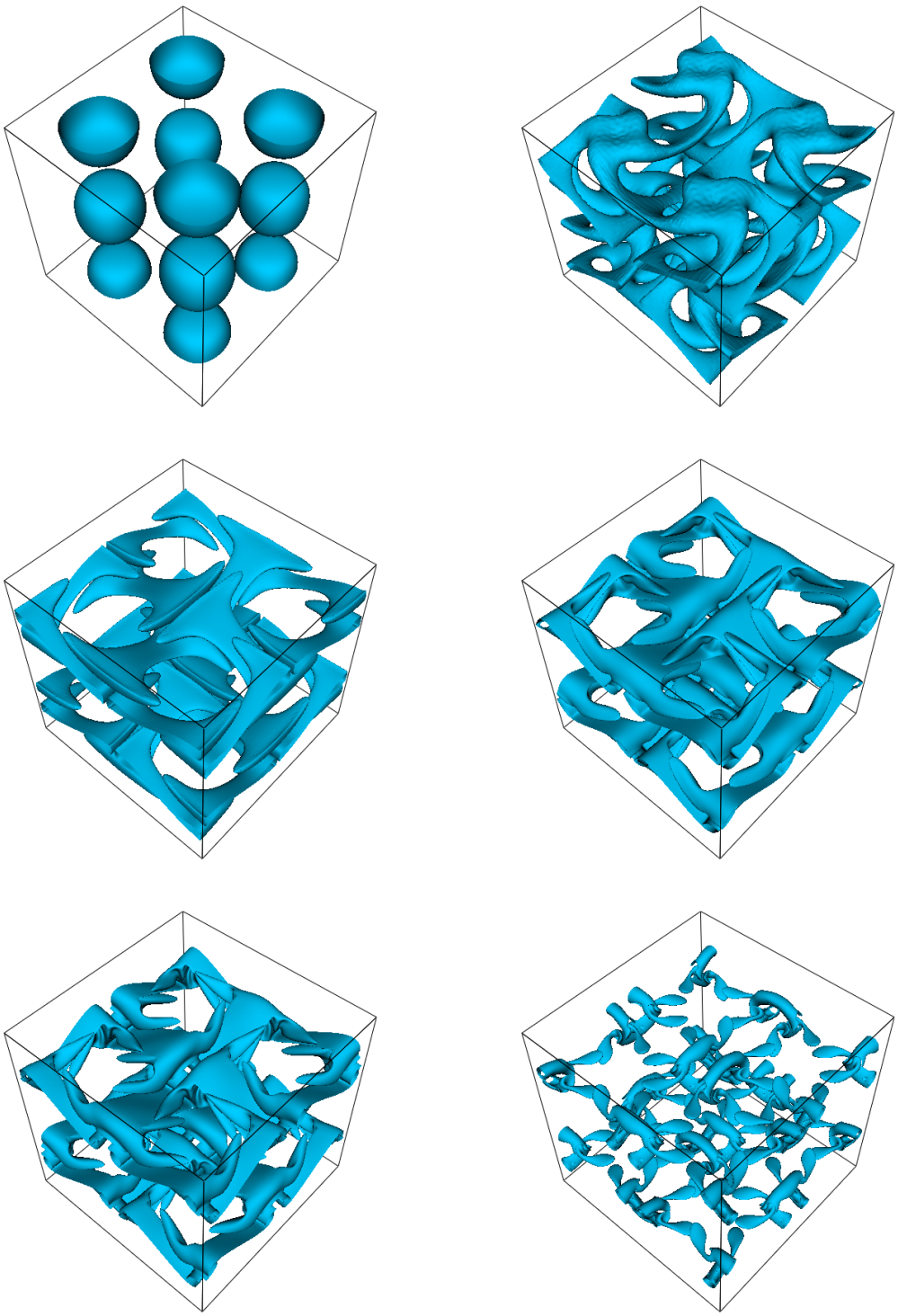

Figure 34: The evolution of the iso-surfaces of the vorticity magnitude $|\omega|$ in the Taylor-Green vortex at $Re = 400$. The iso-surface locations have been reported at specific values and times for comparison to the work of Yang and Pullin (2011) [128]. (a) t = 0, $\hat{\omega} = 0.6$, (b) t = 1.5, $\hat{\omega} = 0.9$, (c) t = 3, $\hat{\omega} = 0.6$, (d) t = 4.5, $\hat{\omega} = 0.2$, (e) t = 5, $\hat{\omega} = 0.2$, (f) t = 7, $\hat{\omega} = 0.6$. $\hat{\omega}$ is the vorticity magnitude scaled by the maximum value in the domain at each instant in time.

4.5 Conclusions

The present work is the first to demonstrate an error-controlled, wavelet-adaptive, all-Mach number solver that has been designed from the ground-up to run effectively on the GPU architecture. The numerical methods were carefully chosen and developed to achieve both spatial and temporal error control while still maintaining high-performance for both compressible and nearly incompressible flows. HPC hardware is evolving to promote more thread-level parallelism, and the need for tightly coupled development of the numerical methods and their implementation on a particular hardware platform is becoming more important since even the most accurate solver is less useful if it cannot perform well on modern computing platforms. Likewise, even the fastest solver is less than useful if it does not provide sufficiently accurate results for a particular application. Modern simulation development requires careful design from the applied mathematics, physics, and computer science perspectives.

The three verification test cases presented in this chapter demonstrated the low-dissipation properties and high accuracy of the wavelet-adaptive, preconditioned, dual-time stepping flow solver in the low-Mach number regime. Several modifications to existing techniques were implemented in order to accomplish this. First, the quasi-skew-symmetric form of the convection terms in the governing equations was used to prevent the odd-even decoupling seen in other co-located grid solvers. The wavelet-adaptive grid helps to protect against a pitfall of using the skew-symmetric form on under-resolved grids in which a significant amount of energy and other conserved quantities is aliased to higher wavenumbers and subsequently dissipated [Kennedy and Gruber (2008) [61]]. With the resolution provided by the wavelet-adaptive grid, the dissipation of these aliasing errors had minimal impact on the solutions. Secondly, the physical time derivative approximation for the dual-time stepping procedure requires storage of previous solutions that each have a different adaptive grid configuration from the current solution. This required modifications to the wavelet-adaptive grid

strategy to ensure that all of the grid points are present to perform the point-wise time derivative calculations. Finally, an analytical inverse preconditioning matrix was derived for an arbitrary number of species in order to make the preconditioning efficient on the GPU architecture with its limited amount of cache per thread. Prior to this development, the preconditioning matrix required a numerical inverse to be performed at each grid point, which is very resource intensive. The savings in computing time from using the analytical inverse increases as the number of transport equations (*e.g.*, number of species) increases.

Chapter 5 of this thesis will describe the final steps in the development of the wavelet method for use in a nucleate boiling simulation with acoustic forcing. It will describe the addition of a level-set formulation to model a multiphase flow and the use of general equations of state to properly enforce the mass, momentum, and energy balances on fluid-fluid interface. This final combination of numerical techniques is a novel approach in the development of a direct numerical simulation of nucleate boiling with acoustic forcing. It offers high resolution and fidelity to the underlying physics of the simulation and significantly reduced compute times for the simulation of complex, 3D, multiphase flows on a single workstation.

CHAPTER V

LOW-SPEED MULTIPHASE COMPRESSIBLE FLOW

5.1 Introduction

Multiphase flows arise in a number of applications, such as shock-droplet (bubble) interactions, droplet impact, droplet atomization, film boiling, and nucleate boiling. In most of these applications, the calculation of the effect of surface tension on the motion of the interface is required in order to accurately represent the physics of the system. To do this, numerical methods must be implemented to compute and track the location of the interface, to determine the normal-vector orientation, and to compute the curvature of the interface.

Interface tracking techniques generally fall into two categories: 1) explicit front-tracking where the grid points follow the interface in a Lagrangian fashion, and 2) implicit interface tracking where the location of the interface is computed directly on an Eulerian grid. The level-set method is one form of implicit interfacing tracking that uses a higher-dimensional representation of the interface in the form of a signed-distance field. The main advantage of this method is that topological changes of the interface are handled automatically. In contrast, explicit methods require remeshing or mesh modification around topological changes, such as bubble merging or pinch-off. The choice of which method to use in the present work must also consider the computational performance of its implementation on the GPU architecture and its compatibility with the wavelet-adaptive finite-difference method developed in the two previous chapters.

There have been many successful implementations of explicit front tracking techniques, such as the work by Unverdi and Tryggvason (1992) [116] for isothermal flows

and Juric and Tryggvason (1998) [58] for boiling flows, and they have achieved very high accuracy. Their work treats the fluid-fluid interface with a Lagrangian moving mesh moving with the local flow velocity to avoid mass loss due to numerical diffusion. They define a small transition region for fluid properties about the explicitly tracked interface. This is in contrast to the level-set method, where the entire domain is treated as one continuous fluid with a rapid transition in properties across the interface that is advected in an Eulerian framework. An additional benefit of Tryggvason's implementation (2010) [113] is that subgrid (*e.g.*, lubrication approximation) models can be used in place of fully resolving small gaps between bubbles and/or walls with reasonable accuracy for large problems. Despite the success of the front-tracking method, this approach is not used in the present work due to the incompatibility with the structured, Cartesian wavelet-adaptive grid and the goal of efficiently performing simulations on the GPU architecture. Instead, the level-set method is used to continue the development of a GPU-based, wavelet-adaptive, multiphase flow solver.

The level-set method as introduced by Osher and Sethian (1988) [83] uses the zero level set of a field of signed-distance values (defined as the distance to the nearest interface) to represent the interface. The motion and possible deformation of the interface is performed by the flow advecting the signed-distance field. Because of the form of this simple advection process, many numerical techniques developed for the discretization of hyperbolic conservation laws can be used.

As the signed-distance field is advected by the flow it remains reasonably accurate through periods of small deformation. However, over time the signed-distance field becomes inaccurate, especially for large deformations. This occurs because the signed-distance is not a conserved quantity. As the signed-distance field deteriorates over time, the accuracy of normal vector and curvature calculations based on this field also deteriorates. This problem can be remedied by periodically *reinitializing* the signed-distance field. Reinitialization can be done in a number of ways, including

fast-marching methods [Sethian (1996) [98]] or PDE-based reinitialization [Peng et al. (1998) [86]]. These methods reconstruct the signed-distance field in a narrow band about the interface while keeping the interface stationary during reinitialization. The reason for only reinitializing in a narrow band about the interface is because this is the only region where normal vectors and curvature need to be calculated (and are well-defined). Thus, the method can be made more computationally efficient by not recomputing the signed-distance everywhere in the domain. PDE-based reinitialization is chosen in the present work due to the ease of parallel implementation and the simplicity of the approach.

Once the location of the interface is defined it must be modeled in the equations of motion. The sharp interface model includes the interface as a 2D surface in the flow with boundary conditions across the surface used to connect the two fluids and to account for surface tension and phase change. This model creates a singularity in the flow domain, which must be addressed in a numerical simulation. Sharp interfaces have been approximated by many researchers. One successful method introduced by Fedkiw et al. [40] is the *Ghost Fluid Method*. This method was designed for compressible two-phase flow to eliminate the oscillations across the interface caused by the discontinuous nature of the sharp transition. This method also uses a constant extrapolation across the interface, which is a concern with the higher-order, error-controlled framework developed in the present work.

The diffuse interface model avoids this singularity by approximating the interface as a narrow, finite, and continuous transition between the two fluids. This model creates some ambiguity in where to apply the surface tension forces since there is no longer a sharp boundary. Thus, surface tension is applied in a narrow band about the interface as determined by an approximate delta function. If a numerical method can fully resolve this transition region, any problems with the discontinuity that are seen in a sharp interface model will not appear. Since a wavelet-adaptive grid can

provide this level of resolution, the diffuse interface model will be used in the present work.

The implementation of the diffuse interface model is based on a simple idea; create an interfacial transition region that is fully resolvable by the wavelet-adaptive grid. The width of the transition region is chosen by the user as a compromise between interface thickness and the computational expense incurred by the local level of resolution required about the interface. The order of accuracy (related to the number of moments preserved) of the delta function used to include surface tension forces is commensurate with the overall order of accuracy of the rest of the discretization methods used (*i.e.*, wavelet, finite-difference derivative, and boundary condition orders of accuracy). Since the wavelet-adaptive method in the present work allows for a very large amount of local refinement at the interface while maintaining reasonable computational expense, the use of the diffuse interface model will maintain the high overall order of accuracy of the framework developed in the present work.

Finally, the properties of the fluids on each side of the interface are determined by separate equations of state. The property transitions across the interface are prescribed as a mass-weighted average, where the mass fraction is approximated by a hyperbolic tangent function to provide a smooth transition through the diffuse interface that is fully resolvable by the wavelet-adaptive grid. The compressible formulation of the Navier-Stokes equations allows a single equation of state to be used, but at the present time there is not a known fluid model that is accurate enough to handle the dense liquid, transition, and superheated vapor regions and is computationally suitable for the large number of evaluations that are required throughout the course of a simulation, particularly for water. Thus, when using separate equations of state for each fluid, source terms arising from the liquid-vapor phase change must be added to the governing equations.

The incorporation of the level-set method for diffuse interfaces, surface tension,

and phase-change source terms completes the development of the GPU-based, error-controlled wavelet-adaptive finite-difference multiphase flow simulation started in Chapter 3. In this development, numerical techniques were chosen, improved, and implemented to work together to create a novel high-performance simulation that runs entirely on the GPU architecture, an essential element in the current path to predictive exascale computing [Ahern et al. (2011) [1]].

5.2 Governing Equations

In these simulations, the physics for the evolution of a compressible, viscous flow is governed by the same equations as those in Chapter 4 § 4.2, including the quasi-skew-symmetric form of the convective terms. To allow for a multiphase flow, the momentum equation is modified to include a surface-tension force and the level-set advection equation is included in the set, as shown.

$$\frac{\partial \rho u_i}{\partial t}+\frac{1}{2}\left(\frac{\partial \rho u_i u_j}{\partial x_j}+u_j\frac{\partial \rho u_i}{\partial x_j}+\rho u_i\frac{\partial u_j}{\partial x_j}\right)=-\frac{\partial p}{\partial x_i}+\frac{\partial \tau_{ij}}{\partial x_j}+\rho g_i+\sigma\kappa\delta(\phi)\hat{n}_i+\left(\delta_{ij}-\hat{n}_i\hat{n}_j\right)\delta_{ij}\frac{\partial \sigma}{\partial x_i}\delta(\phi) \tag{37}$$

$$\frac{\partial \rho E}{\partial t}+\frac{1}{2}\left(\frac{\partial \rho E u_j}{\partial x_j}+u_j\frac{\partial \rho E}{\partial x_j}+\rho E\frac{\partial u_j}{\partial x_j}\right)+\frac{\partial p u_j}{\partial x_j}=-\frac{\partial q_j}{\partial x_j}+\frac{\partial u_i\tau_{ij}}{\partial x_j}+\rho u_i g_i-\dot{m}\delta\left(h_{fg}+(c_l-c_g)T_{sat}\right) \tag{38}$$

$$\frac{\partial \phi}{\partial t}+\left(u_j+\frac{\dot{m}}{\rho}\hat{n}_j\right)\frac{\partial \phi}{\partial x_j}=0 \tag{39}$$

Here, ρ is the density, u_i is the velocity vector, p is the pressure, τ_{ij} is the viscous stress tensor, g_i is the acceleration of gravity vector, σ is surface tension, κ is the interface curvature, ϕ is the signed-distance field, $\delta(\phi)$ is the approximate delta function for applying surface tension forces in the diffuse interface model, $\dot{m}$ is the mass flux across the interface, h_{fg} is the latent heat of vaporization, and c is the specific heat

capacity at constant pressure, and $\hat{n}_i$ is the normal vector to the interface. Equation 39 is used to advect the level-set field ϕ, and ϕ is used in the normal-vector and curvature calculations to include the surface tension force in the momentum equations 37.

These conservation equations are augmented by a set of constitutive equations for the viscous stress, heat flux, and mass diffusion. Thermodynamic properties will use an equation of state appropriate to the material. For example, air could be modeled as a calorically perfect ideal gas.

The smooth property transition across the interface is defined by Equation 40.

$$H = \frac{1}{2}\left(\tanh\frac{2x}{\delta_i} + 1\right) \tag{40}$$

Here, H is an approximate, smooth Heaviside function, and δ_i is the interface width.

The approximate delta function is defined by Equation 41.

$$\delta(\xi) = \begin{cases} 1 - \frac{1}{2}|\xi| - |\xi|^2 + \frac{1}{2}|\xi|^3 & : \quad 0 \leq |\xi| \leq 1, \\ 1 - \frac{11}{6}|\xi| + |\xi|^2 - \frac{1}{6}|\xi|^3 & : \quad 1 \leq |\xi| \leq 2 \end{cases} \tag{41}$$

Here, ξ is the normalized signed distance. The width of the approximate delta function and the property (interface) transition width are chosen to be the same size.

5.3 Numerical Methods

5.3.1 Wavelet-Adaptive Method

The governing equations 21–24, along with the surface tension modifications in 37 and the level-set equation 39 are discretized using a wavelet-adaptive grid and a centered finite difference scheme is used for the derivatives. The present work uses second-generation wavelets implemented through a *Fast Wavelet Transform* that uses p^{th}-order Lagrange interpolating polynomials to determine the wavelet coefficients. The

method is discussed in detail in Chapter 3 of this thesis. The result of that work and its further development in Chapter 4 is a GPU-based wavelet-adaptive flow solver for compressible and incompressible flows at all Mach numbers.

To develop a GPU-based wavelet-adaptive solver for low-speed, compressible, *multiphase* flows, the level-set method is described in the next section.

5.3.2 Level-set Method

The level-set method captures the interface implicitly using a signed-distance field, with the interface defined by the zero level set. The evolution of the interface shape is handled by advecting the signed-distance field using Equation 39. The interface bounds a region $\Omega \subset R^3$ (*e.g.*, a droplet or bubble), and the level-set function has the following properties, which are defined in Equation 42.

$$\begin{aligned} \phi(\boldsymbol{x}, t) &< 0 \quad for\ \boldsymbol{x} \in \Omega \\ \phi(\boldsymbol{x}, t) &\geq 0 \quad for\ \boldsymbol{x} \notin \Omega \end{aligned} \tag{42}$$

As the signed-distance field is advected by the flow it gradually loses its signed-distance property and needs to be reinitialized. This is achieved by solving Equation 43.

$$\frac{\partial \phi}{\partial \tau} + \operatorname{sgn}(\phi_0)(\nabla \phi - 1) = 0 \tag{43}$$

Here, τ is a fictitious time for propagating the distance calculation away from the interface, and $\operatorname{sgn}(\phi_0)$ is a one-dimensional smeared sign function that is approximated numerically as shown in Equation 44.

$$\operatorname{sgn}(\phi_0) = \frac{\phi_0}{\sqrt{\phi_0^2 + (\nabla x)^2}} \tag{44}$$

As Equation 43 is solved towards steady-state in the fictitious time, the level-set value is restored throughout the domain along the normal direction away from the

interface location. This occurs because Equation 43 assumes a unit velocity in the direction away from the interface, and the distance is simply the amount of time it takes for the propagated signed-distance to reach each grid point. Periodic reinitialization of the signed-distance field allows for continued accurate normal vector and curvature calculations for implementing surface tension forces. Rather than solving all the way to steady-state in fictitious time, the reinitialization is solved over a shorter duration that allows the signed-distance to propagate over a narrow band about the interface, only where normal vector and curvature calculations are needed. This requires fewer iterations and reduces computational expense.

In the diffuse interface model surface tension appears in the momentum Equation 37. Here an approximate delta function spreads the surface tension forcing over the width of the diffuse interface. The singular surface tension force term can be approximated to an order of accuracy (*i.e.*, the number of discrete moments preserved) that matches the spatial discretization of the governing equations [Waldon (1999) [122], Tornberg and Engquist (2004) [111]]. This is important to ensure the surface tension force is accurately distributed about the interface as a source term to regain the same integral force as with a sharp interface method and the surface tension error converges at the correct rate with grid refinement.

The level-set advection equation is discretized using a 5^{th}-order Weighted Essentially Non-Oscillatory (WENO) scheme [Jiang and Shu (1995) [57]] to handle the discontinuous signed-distance function. This is needed since the level-set advection equation has a hyperbolic form with no dissipation that allows true discontinuities to develop in the signed-distance field. Only the level-set advection term is discretized using the 5^{th}-order WENO scheme. The rest of the governing equations are discretized using high-order central differences as previously discussed.

5.4 Verification

Chapter 4 of this thesis provided verification of the accuracy of the wavelet-adaptive, compressible flow solver combined with the preconditioned dual-time stepping method to solve for very low-Mach number flows; solutions not attainable using explicit time integration on the compressible Navier-Stokes equations. This section demonstrates the accuracy of the all-Mach-number flow solver combined with the level-set method and a diffuse interface model to implement surface tension and phase change for the solution of multiphase flows.

Three verification problems were chosen to demonstrate the various aspects of this multiphase implementation. The 3D level-set advection test demonstrates that the interface is accurately tracked through large deformations without excessive mass loss or tearing of the interface. The free droplet oscillation problem tests the ability of the method to capture interfacial dynamics. Lastly, the problem of 3D bubble growth in a quiescent superheated fluid highlights the accuracy of the phase-change implementation. The test problems from this chapter together with the previous two chapters completes the verification of all components in the wavelet-adaptive compressible flow simulation for multiphase flows.

5.4.1 3D Level-set Advection Test

The test problem developed for level-set methods by LeVeque (1996) [68] is used here. The flow field given by Equation 45 is prescribed in a unit domain. This vortical flow causes deformations in the $x_1 - x_2$ plane and the $x_1 - x_3$ plane. A spherical interface of radius 0.15 is placed in the flow at (0.35, 0.35, 0.35) and the simulation is run until time $t = 3$. The prescribed vortices cause the sphere to be stretched, flattened, and curled over on itself as it translates.

$$
\begin{aligned}
u(\boldsymbol{x}) &= 2\sin^2(\pi x_1)\sin(2\pi x_2)\sin(2\pi x_3) \\
v(\boldsymbol{x}) &= -\sin^2(\pi x_2)\sin(2\pi x_1)\sin(2\pi x_3) \\
w(\boldsymbol{x}) &= -\sin^2(\pi x_3)\sin(2\pi x_1)\sin(2\pi x_2)
\end{aligned}
\tag{45}
$$

Enright and Fedkiw (2002) [38] used this test problem with a 100^3 uniform grid with and without Lagrangian particle corrections for Eulerian advection errors. At the maximum time, the deformation causes the thinnest region to be approximately 1 grid spacing on the uniform grid. The standard level-set method (with no particle corrections) fails severely with a large mass loss and many holes present in the deformed shape. With particle corrections, there are still holes in the shape, but the error is maintained at a much lower level. At the maximum deformation, $t = 3$, the reported errors for mass loss are 49% and 1.9% for the standard level-set and particle level-set methods, respectively.

The grid refinement for this test problem in the present work was limited to six levels of refinement on top of a base grid of 16^3, for a maximum resolution equivalent to 961^3. This was chosen to limit computational expense, additional levels of resolution would improve accuracy at an increased cost. The initial and final state are shown in Figure 35. At the maximum time there are no holes in the interface and the mass loss is 1.5%. The wavelet-adaptive grid uses approximately 2% of the possible 961^3 at the finest uniform grid (for a compression ratio of approximately 50) while holding a wavelet tolerance of 10^{-3}.

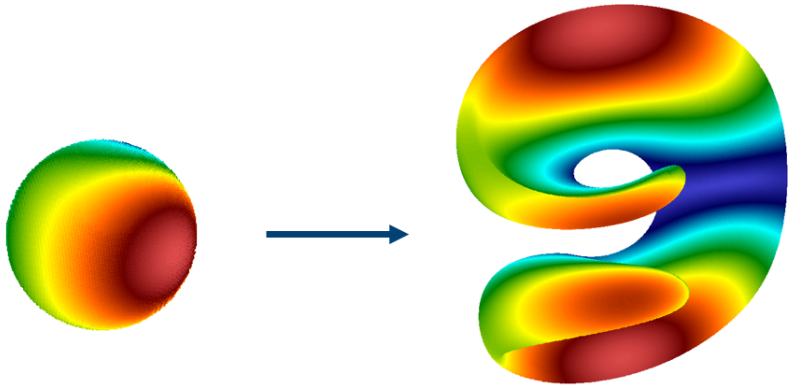

Figure 35: The 3D Level-set deformation test demonstrates the ability of the method to capture large deformations and narrow regions without excessive mass loss that is common in under-resolved simulations. The color of the zero-level-set iso-surface indicates the velocity magnitude to provide some depth and help distinguish overlapping regions of the plot.

5.4.2 Free Oscillating Droplet

Analytical solutions for the small-amplitude oscillations of droplets and bubbles were developed by Lamb (1881) [65] for the inviscid case and Prosperetti (1980) [90] for the viscous case. The initial configuration of the droplet in this test problem is shown in Figure 36. In this test case, a liquid droplet with a density of 1000 kg/m^3 and dynamic viscosity of 10^{-3} Pa-s is surrounded by an ideal gas with a density of 1 kg/m^3 and dynamic viscosity 10^{-5} Pa-s, resulting in a density ratio of 1000:1 and viscosity ratio of 100:1. The speed of sound in the liquid and gas is approximately 1500 m/s and 347 m/s, respectively. The liquid is described by a stiffened gas equation of state [Glaister (1988) [47]]. The mean droplet diameter is 0.002109 m (determined from the volume of the initial deformed shape), and the surface tension coefficient is 0.07 N/m. The oscillation frequency of the droplet is 110.0 Hz, as predicted by Prosperetti (1980) [90] and given by Equation 46.

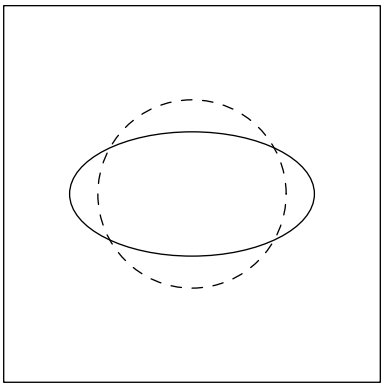

Figure 36: A schematic of the cross section of the initial deformed droplet (solid) and the final spherical equilibrium state (dashed).

$$\begin{aligned} \omega_n^2 &= n(n-1)(n+2)\frac{\sigma}{\rho R^3} \\ \beta_n &= (n-1)(2n+1)\frac{\nu}{R^2} \\ f_n &= \frac{1}{2\pi}\sqrt{w_n^2 - \beta_n^2} \end{aligned} \tag{46}$$

The oscillation frequency in the wavelet simulation was determined by finding the times corresponding to the maximum deformation amplitude along a principle axis of the droplet and then measuring the period. The observed oscillation frequency was 109.5 Hz, which is within 0.45% of the predicted frequency. When the droplet reached steady state, the pressure jump due to surface tension of the spherical drop was measured to be within 0.1% of the value predicted by Equation 47.

$$\Delta P = \frac{2\sigma}{R} \tag{47}$$

5.4.3 Bubble Growth in a Quiescent Superheated Liquid

The growth of a bubble in a superheated liquid has an approximate analytical result provided by the work of Mikic et al. (1970) [76], which makes this problem a useful verification test case for the heat transfer and phase-change aspects of the simulation.

In this problem, a bubble with an initial diameter of 0.002 m is seeded in a uniformly superheated liquid under quiescent conditions. The system is approximately

liquid water and water vapor at a pressure of 1.66 bar. The density ratio is approximately 1000:1. The latent heat of vaporization is prescribed as 2257 kJ/kg. The liquid is superheated by 30 K, and the surface tension is neglected to more closely match the assumptions of the analytical solution.

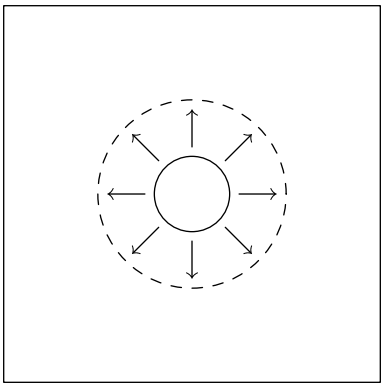

Figure 37: A cross section of the initial (solid) and final (dashed) configurations of a spherical bubble in a superheated liquid in a 3D domain. The arrows indicate the bubble growth, and the bubble remains spherical as it grows.

For reference, the analytical result from Mikic et al. (1970) [76] is given in Equation 48.

$$
\begin{aligned}
R^+ &= \frac{2}{3}\left[\left(t^+ + 1\right)^{3/2} - \left(t^+\right)^{3/2} - 1\right] \\
R^+ &= \frac{R}{B^2/A} \\
t^+ &= \frac{t}{B^2/A^2} \\
A &= \sqrt{\frac{2}{3}\frac{(T - T_{sat})h_{fg}\rho_v}{T_{sat}\rho_l}} \\
B &= \sqrt{\frac{12}{\pi}Ja^2\alpha_l}
\end{aligned}
\tag{48}
$$

Here, R^+ is the dimensionless bubble radius, t^+ is the dimensionless time, T_{sat} is the saturation temperature at the pressure far from the bubble, T_∞ is the temperature of the liquid far from the bubble, ρ is the density, c is the specific heat capacity of the liquid, h_{fg} is the latent heat of vaporization, and α is the thermal diffusivity. The

subscripts l, v, and sat indicate liquid, vapor, and saturation properties, respectively. The Jakob Number is defined by Equation 49.

$$Ja = \frac{\rho_l c_l (T_\infty - T_{sat})}{\rho_v h_{fg}} \tag{49}$$

The results from the present simulation closely match the analytical result of Mikic et al. (1970) [76], as shown in Figure 38. Some differences stem from the initial conditions, since the simulation starts with a finite-sized bubble and the analytical result starts from a zero radius, and the initial temperature near the liquid-vapor interface was approximated with a hyperbolic tangent function for a smooth radial profile.

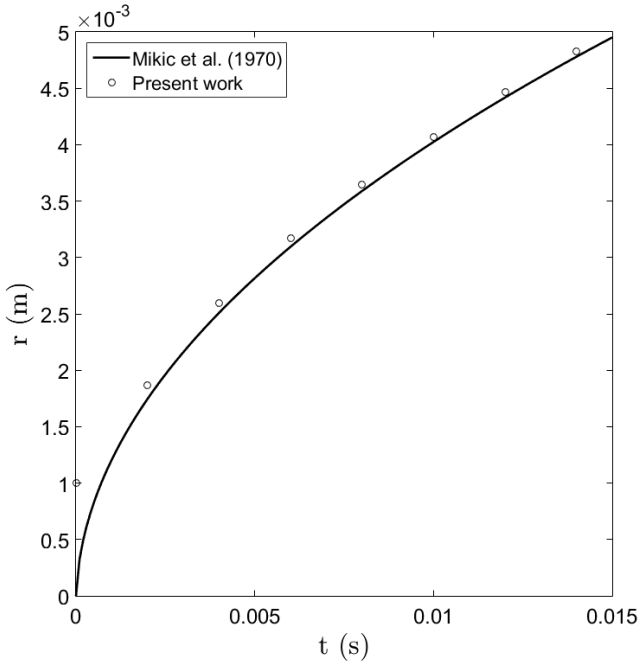

Figure 38: Bubble-radius growth over time for $Ja = 55.8$ and the properties matching the simulation parameters listed above.

5.5 Single-Bubble Nucleate Boiling

The wavelet-adaptive multiphase flow solver was designed to address problems in boiling. This capability is demonstrated in this section using the problem of nucleate

boiling of a single bubble over a flat plate with a prescribed wall superheat. The working fluid is water at a pressure of 1.66 bar, the liquid region is modeled using the stiffened gas equation of state, and the vapor region is modeled as an ideal gas. The density ratio is approximately 1000:1 and the viscosity ratio is 100:1. The surface tension coefficient is 0.07 N/m, gravity is 9.81 m/s^2, and the wall superheat is 8 K. The results of this demonstration are shown in Figure 39.

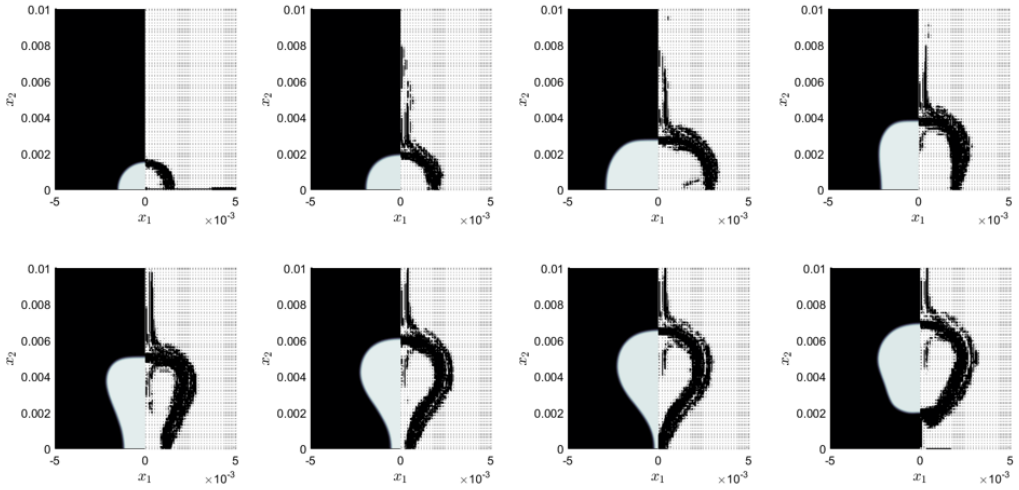

Figure 39: Evolution of single-bubble nucleate boiling with a contact angle of 90°. The left half of each plot shows the mass fraction with the liquid in black and vapor in grey. The right half of each plot shows the adapted grid.

5.6 Conclusions

The present work is the first known demonstration of a GPU-based, wavelet-adaptive, all-Mach number, compressible, multiphase flow solver. The solver uses a diffuse interface model with surface tension and phase change, while interfacial motion is computed using the level-set method. The accuracy and performance of the single-phase, compressible flow solver was addressed in Chapter 3 and Chapter 4.

For the multiphase, compressible flow solver, the accuracy of the level-set method was assessed using a common deformation test from the level-set literature. The results show very good mass conservation while the wavelet-adaptive grid reduced

the grid-point population to 2% of that required by an equivalent uniform grid at the finest resolution of the wavelet grid. The maximum mass loss is 1.5%, which is slightly better than the results from the particle level-set method used in the comparison.

The droplet oscillation problem demonstrates the ability of the simulation to accurately capture droplet interface dynamics. The resulting oscillation frequency is within 0.45% of the analytical solution for small-amplitude oscillations, and the steady-state Young-Laplace pressure jump is within 0.1% of the predicted value for a spherical droplet. The diffuse interface approach, in conjunction with a wavelet-adaptive grid and a preconditioned, dual-time stepping method, allows for a stable solution of these multiphase flows at large density ratios of $O(10^3)$; ratios typical of water and air.

The problem of bubble growth in a superheated liquid demonstrates the accuracy of the method for phase-change applications. The results of the simulation are reasonably close to the analytical result of Mikic et al. (1970) [76] when considering the differences between the two models.

Each component of the multiphase flow solver has now been verified, and the simulation has the capability to solve for multiphase flows at all Mach numbers with and without phase change. This capability will allow simulations of nucleate boiling in the presence of acoustic fields to be performed. In turn, the will enable the study of enhanced heat transfer mechanisms and the optimization of new, enhanced, heat exchanger designs.

CHAPTER VI

CONCLUDING REMARKS

The present work is the first known wavelet-adaptive, all Mach-number, multiphase flow solver designed and built to run exclusively on a single GPU. The design completely avoids data transfers over the PCI-express bus during solving, which results in greatly improved execution and memory performance. The single-phase version of this flow solver was presented earlier by Forster and Smith (2014) [41]. This solver is a significant improvement over the work of other groups [Rossinelli et al. (2011) [92]] that investigated the acceleration of wavelet-adaptive-grid frameworks by offloading some of the work to GPUs. However, that approach incurs significant overhead with PCI-express bus transfers between the host CPU and the GPU, although Rossinelli et al. (2011) [92] and Van Rees et al. (2013) [117] hide much of this overhead (˜75%) with concurrent and expensive multiphysics calculations on the CPU.

Chapter 3 demonstrated that a data set with localized features can be accurately resolved with a wavelet-adaptive multiresolution representation using a smaller number of points compared to an equivalent uniform grid at the highest resolution. A 1D example of a step function showed a compression ratio of 118. Similar examples for 2D and 3D showed compression ratios up to $O(10^5)$ and $O(10^8)$, respectively. These large grid compression ratios significantly reduce memory requirements and the number of computations required during a simulation of a set of PDEs. This means that a wavelet-adaptive grid on a desktop workstation with a single GPU can represent highly resolved solutions to complex three dimensional PDE problems that would otherwise be inaccessible.

The numerical accuracy of the wavelet representation of a test function and the

finite difference evaluations of its derivatives was verified. The function error scales almost linearly with the user-specified wavelet tolerance and can reach to almost machine precision with wavelets of order 8–10. Another benefit of high-order wavelets is the reduction in the number of points needed to represent the function. This number decreases by a factor of 500 going from 4^{th}- to 10^{th}-order wavelets. The same trends were seen for the derivative error in the function using finite difference formulations with the same order of accuracy as the wavelets. However, the absolute errors were an order of magnitude higher for the first derivative and up to seven orders higher for the second. The sensitivity of the solution with respect to the wavelet tolerance ϵ depends on the physics of the problem, and the error can be controlled by varying this single parameter.

The GPU-based, wavelet-adaptive, compressible flow solver was compared to a uniform-grid, finite-difference, compressible flow solver that also runs entirely on the GPU. With the 2D Richtmyer-Meshkov instability problem as a test case, the time-to-solution for the GPU-based simulation was two orders of magnitude faster that the uniform-grid solver, with three orders possible at maximum resolution. While this is a projection due to memory limitations when using uniform grids, part of the motivation behind the wavelet method is to create an optimized grid to reduce memory requirements so that larger problems can be solved with the available memory. Figure 23 shows the extent to which this is possible. By comparing the performance of the traditional uniform-grid solver and the wavelet-adaptive solver, the reader can estimate the possible performance gains based on their own particular solver choices and hardware configurations for various applications based on the performance of their existing finite-difference simulations.

In the comparison of the GPU-based, uniform-grid solver and the GPU-based, wavelet-adaptive solver, it was observed that the wavelet-adaptive, finite-difference method has an order of magnitude *lower* average throughput of grid points per unit of

time than the uniform-grid solver running on the same GPU. The lower performance is caused by the computing overhead from GPU warp divergence and data lookup. This extra work is more than offset by the reduced problem sizes offered by the grid compression from the wavelet-adaptive method. These results show that the wavelet-adaptive solver performs well on a GPU, a modern hardware architecture that relies heavily on SIMD parallelism and use of reduced cache sizes. This is a important step forward in the continued growth of wavelet-adaptive methods for use in advanced HPC systems.

The three verification test cases presented in Chapter 4 demonstrated the low-dissipation properties and high accuracy of the wavelet-adaptive, preconditioned, dual-time stepping flow solver in the low-Mach number regime. Several modifications to existing techniques were implemented in order to accomplish this. First, the quasi-skew-symmetric form of the convection terms in the governing equations was used to prevent the odd-even decoupling seen in other co-located grid solvers. The wavelet-adaptive grid helps to protect against a pitfall of using the skew-symmetric form on under-resolved grids in which a significant amount of energy and other conserved quantities is aliased to higher wavenumbers and subsequently dissipated [Kennedy and Gruber (2008) [61]]. With the resolution provided by the wavelet-adaptive grid, the dissipation of these aliasing errors had minimal impact on the solutions. Secondly, the physical time derivative approximation for the dual-time stepping procedure requires storage of previous solutions that each have a different adaptive grid configuration from the current solution. This required modifications to the wavelet-adaptive grid strategy to ensure that all of the grid points are present to perform the point-wise time derivative calculations. Finally, an analytical inverse preconditioning matrix was derived for an arbitrary number of species in order to make the preconditioning efficient on the GPU architecture with its limited amount of cache per thread. Prior to this development, the preconditioning matrix required a numerical inverse to be

performed at each grid point, which is very resource intensive. The savings in computing time from using the analytical inverse increases as the number of transport equations (*e.g.*, number of species) increases.

For the multiphase, compressible flow solver presented in Chapter 5, the accuracy of the level-set method was assessed using a common deformation test from the level-set literature. The results show very good mass conservation while the wavelet-adaptive grid reduced the grid-point population to 2% of that required by an equivalent uniform grid at the finest resolution of the wavelet grid. The maximum mass loss is 1.5%, which is slightly better than the results from the particle level-set method used in the comparison.

The droplet oscillation problem demonstrates the ability of the simulation to accurately capture droplet interface dynamics. The resulting oscillation frequency is within 0.45% of the analytical solution for small-amplitude oscillations, and the steady-state Young-Laplace pressure jump is within 0.1% of the predicted value for a spherical droplet. The diffuse interface approach, in conjunction with a wavelet-adaptive grid and a preconditioned, dual-time stepping method, allows for a stable solution of these multiphase flows at large density ratios of $O(10^3)$; ratios typical of water and air.

The problem of bubble growth in a superheated liquid demonstrates the accuracy of the method for phase-change applications. The results of the simulation are reasonably close to the analytical result of Mikic et al. (1970) [76] when considering the differences between the two models.

Extensive verification was performed throughout the development of the wavelet-adaptive, multiphase flow solver to verify the mathematical correctness of the wavelet-adaptive grid, derivative operations, time integration, compressible flow formulation for high-Mach number flows, preconditioned dual-time stepping method for low-Mach number compressible flows, and low-Mach number compressible multiphase flows with

surface tension and phase change. The flow solver demonstrated good error-control properties and low dissipation errors while providing significant grid compression for reduced computational expense. This makes the simulation well-suited for predicting the behavior of nucleate boiling near the CHF, as well as a general purpose simulation for many other multiphase flows with or without interfacial instabilities.

The present work represents several significant advances in the design and use of simulations for high-performance computing.

1. An improved wavelet implementation that is efficient and runs entirely on the GPU architecture to avoid the PCI-express bottleneck for maximum performance.

2. Modifications to the wavelet-grid adaptation to allow the incorporation of the dual-time stepping method.

3. The use of the dual-time stepping method to augment the error control provided by wavelets to obtain rigorous error control in both space and time dimensions.

4. A new derivation of an analytical inverse preconditioning matrix for generalized equation of states using the compressibility factor and the transport of an arbitrary number of species. This inverse avoids costly, large numerical matrix inversions at every grid point and makes preconditioning efficient on the GPU architecture with limited register and shared memory resources.

5. A diffuse interface representation for compressible flows that uses a carefully selected combination of existing methods to maintain the high overall accuracy of the error-controlled framework developed for the wavelet-adaptive grid.

6. A simulation development through a tightly coupled, multidisciplinary approach that investigated the problem from the physics, applied mathematics, and computer science perspectives and allowed for the creation of a high-performance,

error-controlled, simulation framework for a wide variety of multiphysics flow problems with high orders of accuracy.

6.1 Recommendations for Future Work

The next obvious step is to modify the simulation developed in this work that allows very large problems to be solved on a single GPU and make it work with many GPUs for even larger problems and higher resolution (*i.e.*, weak scaling) or solving the same problem size more quickly (*i.e.*, strong scaling). The main challenge here, and with any adaptive grid simulation, will be *load balancing* to prevent an imbalance in work that may leave some compute nodes or GPUs idle while others are still finishing their partition of the work.

At the outset of the present work, GPUs were the only accelerator or large-core count device available as an alternative to the CPU; however, now there are competing computing hardware platforms, such as the many-core Intel Xeon Phi architecture. Future work should include a comparative study to determine which architecture is the most efficient architecture for sparse, adaptive grids. Both GPUs and many-core architectures are being incorporated into upcoming supercomputers and are in the roadmap for exascale computing [Binkley (2016) [7]]. HPC hardware architectures are evolving quickly, and it is likely best to update simulation codes to be portable to run on more than one architecture since they will both be prominent in supercomputing for many years to come. One such recent interface for hardware portability is the Kokkos library [Edwards et al. (2014) [36]].

Advanced equations of state could further improve the fidelity of multiphase flow simulations. GPUs offer the possibility of implementing equations of state that may have previously been too costly to compute in a simulation previously. This is due to the large compute intensity (*i.e.*, the number of floating point operations per load from global memory) required to keep the floating point processing units busy. The extra

computation cost may be hidden by potentially reducing the number of otherwise wasted clock cycles while the processors wait for new data from global memory.

The dual-time stepping method may benefit from implicit time integration schemes in the inner loop for multiphase flows with surface tension. Large surface tension coefficients can create stiffness and increase the number of iterations required for convergence at each physical time step. The implicit integration of pseudo-time may reduce the overall work and time-to-solution by reaching steady-state in the modified system of equations in significantly fewer iterations. For low-speed flows without acoustics, it would be beneficial to use a strictly low-Mach number formulation of the Navier-Stokes in place of the all-Mach number formulation for further reduced computational expense.

REFERENCES

[1] Ahern, S., Shoshani, A., Ma, K.-L., Choudhary, A., Critchlow, T., Klasky, S., Pascucci, V., Ahrens, J., Bethel, E. W., Childs, H., and others, "Scientific discovery at the exascale. report from the doe ascr 2011 workshop on exascale data management," *Analysis, and Visualization*, vol. 2, 2011.

[2] Aktinol, E. and Dhir, V. K., "Numerical simulation of nucleate boiling phenomenon coupled with thermal response of the solid," *Microgravity Science and Technology*, vol. 24, no. 4, pp. 255–265, 2012.

[3] AMD, "Amd corporation." http://www.amd.com/, 2016. Accessed: 2016-02-25.

[4] Bell, J. B., Colella, P., and Glaz, H. M., "A second-order projection method for the incompressible navier-stokes equations," *Journal of Computational Physics*, vol. 85, no. 2, pp. 257–283, 1989.

[5] Berenson, P. J., *Transition boiling heat transfer from a horizontal surface*. Massachusetts Institute of Technology, Division of Industrial Cooperation, 1960.

[6] Bertoluzza, S., Maday, Y., and Ravel, J., "A dynamically adaptive wavelet method for solving partial differential equations," *... methods in applied mechanics and ...*, vol. 7825, no. 93, 1994.

[7] Binkley, S., "Doe office of advanced scientific computing research," 2016.

[8] Blaisdell, G. A., *Numerical simulation of compressible homogeneous turbulence*. PhD thesis, Stanford Univ., CA., 1991.

[9] Blaisdell, G. A., Spyropoulos, E. T., and Qin, J. H., "The effect of the formulation of nonlinear terms on aliasing errors in spectral methods," *Applied Numerical Mathematics*, vol. 21, no. 3, pp. 207–219, 1996.

[10] Bonilla, C. F. and Perry, C. W., "Heat transmission to boiling binary liquid mixtures," in *Chemical Engineering Progresses Symposium Series*, vol. 37, pp. 685–705, 1941.

[11] Botella, O. and Peyret, R., "Benchmark spectral results on the lid-driven cavity flow," *Computers and Fluids*, vol. 27, no. 4, pp. 421–433, 1998.

[12] Boyd, J. P., *Chebyshev and Fourier spectral methods*. Courier Corporation, 2001.

[13] Boziuk, T., Smith, M., and Glezer, A., "Enhanced boiling heat transfer on micromachined surfaces using acoustic actuation," 2010.

[14] Boziuk, T. and Glezer, A. private communication, 2013.

[15] Brachet, M. E., Meiron, D. I., Orszag, S. a., Nickel, B. G., Morf, R. H., and Frisch, U., "Small-scale structure of the TaylorGreen vortex," *Journal of Fluid Mechanics*, vol. 130, p. 411, 1983.

[16] Buelow, P. E.-O., *Convergence Enhancement of Euler and Navier-Stokes Algorithms*. PhD thesis, Pennsylvania State University, 1995.

[17] Canuto, C., Hussaini, M. Y., Quarteroni, A. M., Thomas Jr, A., and others, *Spectral methods in fluid dynamics*. Springer Science & Business Media, 2012.

[18] Capozzoli, A. and Primiceri, G., "Cooling systems in data centers: State of art and emerging technologies," *Energy Procedia*, vol. 83, pp. 484–493, 2015.

[19] Chang, Y. P., "A theoretical analysis of heat transfer in natural convection and in boiling," *Trans. ASME*, vol. 79, no. 7, p. 1501, 1957.

[20] Cook, A. W. and Cabot, W. H., "Hyperviscosity for shock-turbulence interactions," *Journal of Computational Physics*, vol. 203, no. 2, pp. 379–385, 2005.

[21] Cooke, D. and Kandlikar, S. G., "Pool Boiling Heat Transfer and Bubble Dynamics Over Plain and Enhanced Microchannels," *Journal of Heat Transfer*, vol. 133, no. 5, p. 052902, 2011.

[22] Cooper, M. and Lloyd, A., "The microlayer in nucleate pool boiling," *International Journal of Heat and Mass Transfer*, vol. 12, no. 8, pp. 895–913, 1969.

[23] Das, A., Das, P., and Saha, P., "Performance of different structured surfaces in nucleate pool boiling," *Applied Thermal Engineering*, vol. 29, no. 17, pp. 3643–3653, 2009.

[24] Date, A. W., "Fluid dynamical view of pressure checkerboarding problem and smoothing pressure correction on meshes with colocated variables," *International Journal of Heat and Mass Transfer*, vol. 46, pp. 4885–4898, 2003.

[25] Daubechies, I., *Ten lectures on wavelets*, vol. 61. SIAM, 1992.

[26] Daubechies, I., "Where do wavelets come from? a personal point of view," *Proceedings of the IEEE*, vol. 84, no. 4, pp. 510–513, 1996.

[27] DeBonis, J., "Solutions of the Taylor-Green Vortex Problem Using High-Resolution Explicit Finite Difference Methods," *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, no. February, pp. 1–9, 2013.

[28] Desjardins, O., Blanquart, G., Balarac, G., and Pitsch, H., "High order conservative finite difference scheme for variable density low Mach number turbulent flows," *Journal of Computational Physics*, vol. 227, pp. 7125–7159, jul 2008.

[29] Deslauriers, G. and Dubuc, S., "Symmetric iterative interpolation processes," in *Constructive approximation*, pp. 49–68, Springer, 1989.

[30] Dhir, V. K., Warrier, G. R., and Aktinol, E., "Numerical simulation of pool boiling: a review," *Journal of Heat Transfer*, vol. 135, no. 6, p. 061502, 2013.

[31] Dhir, V. K., Warrier, G. R., Aktinol, E., Chao, D., Eggers, J., Sheredy, W., and Booth, W., "Nucleate pool boiling experiments (npbx) on the international space station," *Microgravity Science and Technology*, vol. 24, no. 5, pp. 307–325, 2012.

[32] Dhir, V., "Mechanistic Prediction of Nucleate Boiling Heat TransferAchievable or a Hopeless Task?," *Journal of Heat Transfer*, vol. 128, no. 1, p. 1, 2006.

[33] Donoho, D. L. and Yu, T. P.-Y., "Deslauriers-dubuc: ten years after," *Spline functions and the theory of wavelets (Montreal, PQ, 1996)*, vol. 18, pp. 355–370, 1999.

[34] Douglas, Z., Boziuk, T. R., Smith, M. K., and Glezer, A., "Acoustically enhanced boiling heat transfer," *Physics of Fluids*, vol. 24, no. 5, p. 052105, 2012.

[35] Drew, T. B. and Mueller, A. C., "Boiling," *Trans. AIChE*, vol. 33, pp. 449–473, 1937.

[36] Edwards, H. C., Trott, C. R., and Sunderland, D., "Kokkos: Enabling manycore performance portability through polymorphic memory access patterns," *Journal of Parallel and Distributed Computing*, vol. 74, no. 12, pp. 3202–3216, 2014.

[37] El-Genk, M. S. and Ali, A. F., "Enhanced nucleate boiling on copper microporous surfaces," *International Journal of Multiphase Flow*, vol. 36, pp. 780–792, oct 2010.

[38] Enright, D. P., *Use of the particle level set method for enhanced resolution of free surface flows*. PhD thesis, stanford university, 2002.

[39] Farber, E. and Scorah, R., *Heat transfer to water boiling under pressure*. University of Missouri, 1948.

[40] Fedkiw, R. P., Aslam, T., Merriman, B., and Osher, S., "A non-oscillatory eulerian approach to interfaces in multimaterial flows (the ghost fluid method)," *Journal of computational physics*, vol. 152, no. 2, pp. 457–492, 1999.

[41] Forster, C. J. and Smith, M. K., "Gpu-based multi-resolution direct numerical simulation of multiphase flows with phase change," *Bulletin of the American Physical Society*, vol. 59, 2014.

[42] Forster, C. J. and Smith, M. K., "A GPU-based Wavelet-Adaptive Multiphase Flow Simulation for All Mach Numbers. Part 1: Verification of High-speed Compressible Flow," *Journal of Computational Physics*, 2016.

[43] Garcia-Rojo, R., Herrmann, H., and McNamara, S., "Powders and grains 2005," 2005.

[44] Ghia, U., Ghia, K., and Shin, C., "High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method," *Journal of computational physics*, vol. 411, pp. 387–411, 1982.

[45] Giangaspero, G., van der Weide, E., Svärd, M., Carpenter, M. H., and Mattsson, K., "Case c3. 3: Taylor-green vortex," 2015.

[46] Giordano, J. and Burtschell, Y., "Richtmyer-meshkov instability induced by shock-bubble interaction: Numerical and analytical studies with experimental validation," *Physics of Fluids (1994-present)*, vol. 18, no. 3, p. 036102, 2006.

[47] Glaister, P., "An approximate linearised riemann solver for the euler equations for real gases," *Journal of Computational Physics*, vol. 74, no. 2, pp. 382–408, 1988.

[48] Gottlieb, S. and Shu, C.-W., "Total variation diminishing runge-kutta schemes," *Mathematics of computation of the American Mathematical Society*, vol. 67, no. 221, pp. 73–85, 1998.

[49] Griebel, M. and Koster, F., *Adaptive wavelet solvers for the unsteady incompressible Navier-Stokes equations*. Springer, 2000.

[50] Haar, A., "Zur theorie der orthogonalen funktionensysteme," *Mathematische Annalen*, vol. 69, no. 3, pp. 331–371, 1910.

[51] Holmström, M., "Solving hyperbolic PDEs using interpolating wavelets," *SIAM Journal on Scientific Computing*, vol. 21, no. 2, pp. 405–420, 1999.

[52] Incropera, F. P. and Dewitt, D. P., "Introduction to heat transfer (1996)," *John WHey & Sons. New York. NY.*

[53] Intel, "Intel corporation." http://www.intel.com/, 2016. Accessed: 2016-02-25.

[54] Jameson, A., "Time dependent calculations using multigrid, with applications to unsteady flows past airfoils and wings," 1991.

[55] Jameson, A., "Application of Dual Time Stepping to Fully Implicit Runge-Kutta Schemes for Unsteady Flow Calculations," pp. 1–17, 2015.

[56] Jameson, A., Schmidt, W., Turkel, E., and others, "Numerical solutions of the euler equations by finite volume methods using runge-kutta time-stepping schemes," *AIAA paper*, vol. 1259, p. 1981, 1981.

[57] Jiang, G.-S. and Shu, C.-W., "Efficient implementation of weighted eno schemes.," tech. rep., DTIC Document, 1995.

[58] Juric, D. and Tryggvason, G., "Computations of boiling flows," *International Journal of Multiphase Flow*, vol. 24, no. 3, pp. 387–410, 1998.

[59] Kelley, C. T. and Leyes, D. E., "Convergence analysis of pseudo-transient continuation," vol. 35, pp. 508–523, 1996.

[60] Kennedy, C. A., Carpenter, M. H., and Lewis, R. M., "Low-storage, explicit runge–kutta schemes for the compressible navier–stokes equations," *Applied numerical mathematics*, vol. 35, no. 3, pp. 177–219, 2000.

[61] Kennedy, C. A. and Gruber, A., "Reduced aliasing formulations of the convective terms within the Navier-Stokes equations for a compressible fluid," *Journal of Computational Physics*, vol. 227, no. 3, pp. 1676–1700, 2008.

[62] Kravchenko, A. G. and Moin, P., "On the effect of numerical errors in large eddy simulations of turbulent flows," *Journal of Computational Physics*, vol. 131, pp. 310–322, 1997.

[63] Kuhn, T. S., "The structure of scientific revolutions, international encyclopedia of unified science, vol. 2, no. 2," 1970.

[64] Kwak, D., Reynolds, W. C., and Ferziger, J. H., *Three-Dimensional Time Dependent Computation of Turbulent Flow*. PhD thesis, 1975.

[65] Lamb, H., "On the oscillations of a viscous spheroid," *Proceedings of the London Mathematical Society*, vol. 1, no. 1, pp. 51–70, 1881.

[66] Lee, R. and Nydahl, J., "Numerical calculation of bubble growth in nucleate boiling from inception through departure," *Journal of Heat Transfer*, vol. 111, no. 2, pp. 474–479, 1989.

[67] Leidenfrost, J. G., *De aquae communis nonnullis qualitatibus tractatus*. Ovenius, 1756.

[68] LeVeque, R. J., "High-resolution conservative algorithms for advection in incompressible flow," *SIAM Journal on Numerical Analysis*, vol. 33, no. 2, pp. 627–665, 1996.

[69] Li, R. and Saad, Y., "GPU-accelerated preconditioned iterative linear solvers," *Journal of Supercomputing*, vol. 63, no. 2, pp. 443–466, 2013.

[70] Lienhard, J. H. and Witte, L. C., "An historical review of the hydrodynamic theory of boiling," *Reviews in Chemical Engineering*, vol. 3, no. 3-4, pp. 187–280, 1985.

[71] Liu, X., Xia, Y., and Luo, H., "Assessment of the rdgflo3d for a delta wing at low reynolds number (c2. 2) and dns of the taylor-green vortex (c3. 3)," 2014.

[72] Lorenz, E. N., "Deterministic nonperiodic flow," *Journal of the atmospheric sciences*, vol. 20, no. 2, pp. 130–141, 1963.

[73] Mallat, S., *A wavelet tour of signal processing: the sparse way*. Academic press, 2008.

[74] Mastellone, A., Capuano, F., Di Benedetto, S., Cutrone, L., and Aerospaziali, C. I. R., "C3. 3: Direct numerical simulation of the taylor-green vortex at re= 1600," 2015.

[75] Meyer-Spradow, J., Ropinski, T., Mensmann, J., and Hinrichs, K. H., "Voreen: A rapid-prototyping environment for ray-casting-based volume visualizations," *IEEE Computer Graphics and Applications (Applications Department)*, vol. 29, pp. 6–13, Nov./Dec. 2009.

[76] Mikic, B., Rohsenow, W., and Griffith, P., "On bubble growth rates," *International Journal of Heat and Mass Transfer*, vol. 13, no. 4, pp. 657–666, 1970.

[77] Morinishi, Y., "Skew-symmetric form of convective terms and fully conservative finite difference schemes for variable density low-Mach number flows," *Journal of Computational Physics*, vol. 229, pp. 276–300, 2010.

[78] Müller, B., "Low-mach-number asymptotics of the navier-stokes equations," in *Floating, Flowing, Flying*, pp. 97–109, Springer, 1998.

[79] Nejadmalayeri, A., Vezolainen, A., Brown-Dymkoski, E., and Vasilyev, O. V., "Parallel adaptive wavelet collocation method for PDEs," *Journal of Computational Physics*, vol. 298, pp. 237–253, 2015.

[80] Nukiyama, S., "Film boiling water on thin wires," *Society of Mechanical Engineering*, vol. 37, 1934.

[81] NVIDIA, "Nvidia corporation." http://docs.nvidia.com/, 2016. Accessed: 2016-02-25.

[82] Oefelein, J., *Simulation and analysis of turbulent multiphase combustion processes at high pressures*. PhD thesis, Pennsylvania State University, 1997.

[83] Osher, S. and Sethian, J. A., "Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulations," *Journal of computational physics*, vol. 79, no. 1, pp. 12–49, 1988.

[84] Paolucci, S., "On the filtering of sound from the navier-stokes equations," 1982.

[85] Paolucci, S., Zikoski, Z. J., Grenga, T., and Wirasaet, D., "WAMR: An adaptive wavelet method for the simulation of compressible reacting flow. Part I. Accuracy and efficiency of algorithm," *Journal of Computational Physics*, vol. 272, pp. 814–841, Sept. 2014.

[86] Peng, D., Merriman, B., Osher, S., Zhao, H., and Kang, M., "A pde-based fast local level set method," *Journal of computational physics*, vol. 155, no. 2, pp. 410–438, 1999.

[87] Phillips, N. A., "An example of non-linear computational instability," *The Atmosphere and the Sea in motion*, vol. 501, pp. 501–504, 1959.

[88] Pletcher, R. H., Tannehill, J. C., and Anderson, D., *Computational fluid mechanics and heat transfer*. CRC Press, 2012.

[89] Poinsot, T. and Veynante, D., *Theoretical and numerical combustion*. RT Edwards, Inc., 2005.

[90] Prosperetti, A., "Free oscillations of drops and bubbles: the initial-value problem," *Journal of Fluid Mechanics*, vol. 100, no. 02, pp. 333–347, 1980.

[91] Rhie, C. M. and Chow, W. L., "Numerical study of the turbulent flow past an airfoil with trailing edge separation," *3rd Joint Thermophysics, Fluids, Plasma and Heat Transfer Conference*, vol. 21, pp. 1525—-1532, 1983.

[92] Rossinelli, D., Hejazialhosseini, B., Spampinato, D. G., and Koumoutsakos, P., "Multicore/multi-gpu accelerated simulations of multiphase compressible flows using wavelet adapted grids," *SIAM Journal on Scientific Computing*, vol. 33, no. 2, pp. 512–540, 2011.

[93] Sakurai, A., Shiotsu, M., and Hata, K., "A general correlation for pool film boiling heat transfer from a horizontal cylinder to subcooled liquid: part 2experimental data for various liquids and its correlation," *Journal of heat transfer*, vol. 112, no. 2, pp. 441–450, 1990.

[94] Sakurai, A. t. and Shiotsu, M., "Temperature-controlled pool-boiling heat transfer," in *Heat transfer, 1974. Vol. 4*, 1974.

[95] Saule, E., Kaya, K., and Catalyurek, U. V., "Performance Evaluation of Sparse Matrix Multiplication Kernels on Intel Xeon Phi," pp. 1–19, 2013.

[96] Schneider, K. and Vasilyev, O. V., "Wavelet methods in computational fluid dynamics*," *Annual Review of Fluid Mechanics*, vol. 42, pp. 473–503, jan 2010.

[97] Scriven, L., "On the dynamics of phase growth," *Chemical engineering science*, vol. 10, no. 1-2, pp. 1–13, 1959.

[98] Sethian, J. A., "A fast marching level set method for monotonically advancing fronts," *Proceedings of the National Academy of Sciences*, vol. 93, no. 4, pp. 1591–1595, 1996.

[99] Shen, W. Z., Michelsen, J. a., and Sorensen, J. N., "Improved Rhie-Chow interpolation for unsteady flow computations," *AIAA Journal*, vol. 39, pp. 2406–2409, 2001.

[100] Shiro, N., "The maximum and minimum values of the heat q transmitted from metal to boiling water under atmospheric pressure," *International Journal of Heat and Mass Transfer*, vol. 27, no. 7, pp. 959–970, 1984.

[101] Shu, C. W., Don, W. S., Gottlieb, D., Schilling, O., and Jameson, L., "Numerical convergence study of nearly incompressible, inviscid Taylor-Green vortex flow," *Journal of Scientific Computing*, vol. 24, no. 1, pp. 569–595, 2005.

[102] Skodras, A., Christopoulos, C., and Ebrahimi, T., "The jpeg 2000 still image compression standard," *Signal Processing Magazine, IEEE*, vol. 18, no. 5, pp. 36–58, 2001.

[103] Sod, G., "A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws," *Journal of Computational Physics*, 1978.

[104] Son, G., Dhir, V., and Ramanujapu, N., "Dynamics and heat transfer associated with a single bubble during nucleate boiling on a horizontal surface," *Journal of . . .*, 1999.

[105] Spiegel, S. C., Huynh, H. T., Debonis, J. R., and Glenn, N., "A Survey of the Isentropic Euler Vortex Problem using High-Order Methods," no. June, pp. 1–21, 2015.

[106] Swanson, R. C. and Turkel, E., "On central-difference and upwind schemes," *Journal of computational physics*, vol. 101, no. 2, pp. 292–306, 1992.

[107] Sweldens, W., "The lifting scheme: A construction of second generation wavelets," *SIAM Journal on Mathematical Analysis*, no. November, 1998.

[108] Sweldens, W., "The Lifting Scheme: A Custom-Design Construction of Biorthogonal Wavelets," *Applied and Computational Harmonic Analysis*, vol. 3, pp. 186–200, Apr. 1996.

[109] Sweldens, W. and Schröder, P., "Building your own wavelets at home," *Wavelets in the Geosciences*, 2000.

[110] Taylor, G. I. and Green, A. E., "Mechanism of the Production of Small Eddies from Large Ones," 1937.

[111] Tornberg, A. K. and Engquist, B., "Numerical approximations of singular source terms in differential equations," *Journal of Computational Physics*, vol. 200, no. 2, pp. 462–488, 2004.

[112] Toro, E. F., *Riemann solvers and numerical methods for fluid dynamics: a practical introduction*. Springer Science & Business Media, 2013.

[113] Tryggvason, G., Thomas, S., Lu, J., and Aboulhasanzadeh, B., "Multiscale issues in DNS of multiphase flows," *Acta Mathematica Scientia*, vol. 30, no. 2, pp. 551–562, 2010.

[114] Tryggvason, G., Thomas, S., and Lu, J., "Direct numerical simulations of nucleate boiling," in *ASME 2008 International Mechanical Engineering Congress and Exposition*, pp. 1825–1826, American Society of Mechanical Engineers, 2008.

[115] Tymczak, C., Niklasson, A. M., and Röder, H., "Separable and Non-separable Multiwavelets in Multiple Dimensions," *Journal of Computational Physics*, vol. 175, 2002.

[116] Unverdi, S. O. and Tryggvason, G., "A front-tracking method for viscous, incompressible, multi-fluid flows," *Journal of computational physics*, vol. 100, no. 1, pp. 25–37, 1992.

[117] Van Rees, W. M., Rossinelli, D., Hadjidoukas, P. E., and Koumoutsakos, P., "High performance cpu/gpu multiresolution poisson solver.," in *PARCO*, pp. 481–490, 2013.

[118] Vasilyev, O. V., "Solving Multi-dimensional Evolution Problems with Localized Structures using Second Generation Wavelets," *International Journal of Computational Fluid Dynamics*, vol. 17, pp. 151–168, Mar. 2003.

[119] Vasilyev, O. V. and Bowman, C., "Second-Generation Wavelet Collocation Method for the Solution of Partial Differential Equations," *Journal of Computational Physics*, vol. 165, pp. 660–693, Dec. 2000.

[120] Vermeire, B. C., Witherden, F. D., and Vincent, P. E., "Problem c3. 3 report: Direct numerical simulation of the taylor green vortex," 2014.

[121] Vermeire, B. C., Zwanenburg, P., and Nadarajah, S., "Higher order workshop 3: Problem c3. 3 direct numerical simulation of the taylor-green vortex," 2014.

[122] Waldon, J., "On the approximation of singular source terms in differential equations," *Numerical Methods for Partial Differential Equations*, vol. 15, no. 4, pp. 503–520, 1999.

[123] Wang, Z., Fidkowski, K., Abgrall, R., Bassi, F., Caraeni, D., Cary, A., Deconinck, H., Hartmann, R., Hillewaert, K., Huynh, H., and others, "High-order cfd methods: current status and perspective," *International Journal for Numerical Methods in Fluids*, vol. 72, no. 8, pp. 811–845, 2013.

[124] Welch, S. W., "Direct simulation of vapor bubble growth," *International Journal of Heat and Mass Transfer*, vol. 41, no. 12, pp. 1655–1666, 1998.

[125] Wells, J., "Preparing Scientific Software for Exascale DOE s Office of Science Computation User Facilities," tech. rep., U.S. Department of Energy, Oak Ridge National Laboratory, 2015.

[126] Wirasaet, D., *Numerical solutions of multi-dimensional partial differential equations using an adaptive wavelet method*. PhD thesis, University of Notre Dame, 2007.

[127] Woolley, C., "GPU Optimization Fundamentals," tech. rep., NVIDIA, 2013.

[128] Yang, Y. and Pullin, D. I., "Evolution of vortex-surface fields in viscous TaylorGreen and KidaPelz flows," *J. Fluid Mech*, vol. 685, pp. 146–164, 2011.

[129] Zang, T. A., "On the rotation and skew-symmetric forms for incompressible flow simulations," *Applied Numerical Mathematics*, vol. 7, no. 1, pp. 27–40, 1991.

[130] Zhang, S., Zhao, X., and Bayyuk, S., "Generalized formulations for the rhie-chow interpolation," *Journal of Computational Physics*, vol. 258, pp. 880–914, 2014.

[131] Zuber, N., "Hydrodynamic aspects of boiling heat transfer (thesis)," tech. rep., California. Univ., Los Angeles; and Ramo-Wooldridge Corp., Los Angeles, 1959.

[132] Zucrow, M. J. and Hoffman, J. D., "Gas dynamics, vol. i," *John Wiley and Sons, New York*, pp. 112–115, 1976.

VITA

Christopher J. Forster received his B.S.M.E. and M.S.M.E. degrees from California Polytechnic State University, San Luis Obispo, where he studied turbomachinery, propulsion systems, and computational fluid dynamics and heat transfer. In 2009, he started in the Fluid Mechanics Research Laboratory at the Georgia Institute of Technology to conduct research in the areas nucleate boiling, heat exchanger optimization, and numerical analysis of multiphase flows. He is currently a Department of Energy Office of Science fellow and visiting researcher at the Combustion Research Facility at Sandia National Laboratories in Livermore, California.