Stony Brook University

The Explosive Possibilities of Little Dwarfs: Low-Mach Number Modeling of Thin Helium Shells on Sub-Chandrasekhar Mass White Dwarfs

A Dissertation Presented

by

Adam Michael Jacobs

to

The Graduate School

in Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

in

Physics

Stony Brook University

August 2016

Stony Brook University

The Graduate School

Adam Michael Jacobs

We, the dissertation committee for the above candidate for the Doctor of Philosophy degree, hereby recommend acceptance of this dissertation.

Michael Zingale – Dissertation Advisor
Associate Professor, Department of Physics and Astronomy

Frederick M. Walter – Chairperson of Defense
Professor, Department of Physics and Astronomy

Marivi Fernández-Serra
Associate Professor, Department of Physics and Astronomy

Andrew MacFadyen
Associate Professor, Department of Physics
New York University

This dissertation is accepted by the Graduate School.

Nancy Goroff
Interim Dean of the Graduate School

Abstract of the Dissertation

The Explosive Possibilities of Little Dwarfs: Low-Mach Number Modeling of Thin Helium Shells on Sub-Chandrasekhar Mass White Dwarfs

by

Adam Michael Jacobs

Doctor of Philosophy

in

Physics

Stony Brook University

2016

The classic model of type Ia supernovae still taught in many textbooks describes a white dwarf primarily composed of carbon and oxygen accreting from a companion until it nears the critical Chandrasekhar mass, contracts, ignites carbon fusion and explodes. The research community, however, is seeing whatever consensus that may have existed on this model as the dominant channel to normal type Ia's erode in the face of both observational and theoretical challenges. In my dissertation I present the largest ever suite of three-dimensional models of an alternative type Ia progenitor model: the double detonation model. This model evades the requirement for a near-Chandrasekhar mass white dwarf, making it much easier to satisfy observational and theoretical constraints. The sub-Chandrasekhar systems I investigate are also relevant to a

variety of other possible explosive outcomes such as helium novae, ".Ia" events, atypical/sub-luminous type Ia's, and shell deflagrations. I have deployed and further developed the low-Mach number astrophysical fluid dynamics code Maestro to carry out my study. Most saliently, I have developed Maestro's nuclear reaction modules to target GPU accelerators in leadership supercomputers. I find that the double-detonation model is promising and warrants continued study by providing the broadest and most detailed characterization to date of the pre-explosive three-dimensional evolution. I also comment on what my models suggest about other explosive possibilities.

For Mom, whose independence, sense of adventure, and sacrifice made this work possible.

For Dad, whose discipline, methodical care, and intense need to understand pervade all that I do.

For Aaron and Alex, without whom there is no Adam.

For Mrs. Steaurt and Mrs. Wootan, who provided a firm foundation of mathematics and science upon which I still stand.

For Mr. Barton and Mr. Foster, who added a foundation of computer science.

For Dr. Dunn, who gave me my first taste of research.

For Erin, my partner, my dearest friend, whose scholarly acumen I can only aspire to, whose love and companionship sustain me.

Finally, perhaps most importantly, for Vince.

Contents

List of Figures

List of Tables

Acknowledgements

Though I am listed as the author, the work described here is the product of broad collaboration. Many people and institutions made this possible, and I want to acknowledge some here.

The path toward a Ph.D. is rarely, if ever, a painless one. Walking this path has been immensely rewarding and has presented me with the greatest challenges of my life. Perhaps most students feel this way, but it has felt particularly difficult for me. I would not be at this point without the incredible support and advocacy of Stony Brook's Astronomy group. The group has cultivated a sense of community, respect, and collegiality that is rare among research groups in my experience. Even as a brand new member to the group, I have always been treated as a junior colleague. I was given the independence to learn, to explore, and to grow. This has been a delightful environment to work in. I'm grateful to all of the group's members for facilitating this atmosphere. In particular, I am deeply grateful to my advisor Mike Zingale. He has gone above and beyond the call of a Ph.D. advisor. He has been my resolute advocate throughout our time working together and has worked diligently toward my professional success. I also thank Alan Calder, Jim Lattimer, Stan Metchev, Michal Simon, and Doug Swesty for the myriad forms of mentorship they have offer me throughout my time here.

My fellow students have provided companionship, commiseration, and guidance. I got through courses and the initial phases of the Ph.D. program with the help of Josh Ilany, Morgan Lynch, Andrea Massari, Rahul Patel, Omer Rahman, Jeremy Reeves, and Sam Towers to name a few. Many in the classes above me guided me through courses and into research, including Sara Callori, Sarah Campbell, Regina Caputo, Aaron Jackson, Marija Kotur, Yeunhwan Lim, Chris Malone, Francis Paraan, Shawn Pollard, David Puldon, Josh Schlieder, and Bob Towers. Rahul went on to become a colleague in the astronomy research group, along with colleagues and friends Lupe Barrios, Melissa Hoffman, Max Katz, Missy Louie, Mat Syriac, Don Wilcox, and Tianqi Zhao.

I'm thankful for the mentorship and collegiality of those in the broader astronomy and physics community outside Stony Brook. For making me a bet-

ter scientist and professional, I thank Ann Almgren, John Bell, Caitlin Casey, Kelle Cruz, Dimitri Dounas-Frazer, Saavik Ford, Jedidah Isler, John Asher Johnson, Jessica Kirkpatrick, Laura Lopez, Elisabeth Mills, Jorge Moreno, Nick Murphy, Andy Nonaka, Chanda Prescod-Weinstein, Ken Shen, Kartik Sheth, Johanna Teske, Sarah Tuttle, and Weiqun Zhang.

To those I have surely neglected to mention, I offer my thanks and my apologies for a poor memory.

Much of Ch. 5 was done in collaboration with Oak Ridge National Lab staff and affiliated experts. In particular, I thank Mat Colgrove, Fernanda Foertter, Oscar Hernandez, Jeff Larkin, Bronson Messer, and Frank Winkler.

Some content in this work is adapted from an article in the *Astrophysical Journal* (Jacobs et al., 2016), in particular the first three chapters. The content of Ch. 4 contains draft content for a paper to be submitted to the *Astrophysical Journal*, while content from Ch. 5 contains draft content for another paper to be submitted.

I thank Frank Timmes for making his equation of state publicly available and Stan Woosley for making available data from his published work. The work at Stony Brook was supported by DOE/Office of Nuclear Physics grant DE-FG02-87ER40317 to Stony Brook. The work at LBNL was supported by the Applied Mathematics Program of the DOE Office of Advance Scientific Computing Research under U.S. Department of Energy under contract No. DE-AC02-05CH11231. This research used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231. An award of computer time was provided by the Innovative and Novel Computational Impact on Theory and Experiment (INCITE) program. This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725. This research is part of the "Type Ia Supernovae" PRAC allocation of the National Science Foundation (award number OCI-1036199) and the Blue Waters sustained-petascale computing project, which is supported by the National Science Foundation (award number OCI 07-25070) and the state of Illinois. Blue Waters is a joint effort of the University of Illinois at Urbana-Champagne and its National Center for Supercomputing Applications.

Chapter 1

Introduction & Context

The nature of Type Ia supernovae (SNe Ia) progenitors has been the subject of intense scientific inquiry for decades (for a recent review see Hillebrandt et al. (2013)). This inquiry is motivated in large part by the central role SNe Ia play in measuring the Universe's cosmic expansion history, revealing the apparent existence of dark energy (Riess et al., 1998; Perlmutter et al., 1999). Yet, a robust theory of SNe Ia progenitors eludes us.

The situation is further complicated by the existence of a dominant population of "normal" SNe Ia alongside peculiar populations. Normal SNe Ia are characterized in part by prominent Si II and Ca II absorption features in their maximum light spectra, absolute B-band peak magnitudes of about -18.5, and lightcurves obeying an empirical width-luminosity relation (see e.g. Hillebrandt et al. (2013); Branch et al. (1993); Phillips (1993)). About 70% of observed SNe Ia fit in this category (Li et al., 2011). There are multiple classes of peculiar SNe Ia which are defined by the ways in which they deviate from normal SNe Ia in their spectra, peak brightness, and/or lightcurve shape.

Early on, the single-degenerate, Chandrasekhar-mass progenitor model emerged as a promising solution to the question of SNe Ia origins and has been the most studied model. The requirement of a critical mass could explain why observed SNe Ia are so near-uniform, empirically obeying a width-luminosity relation. However, as instruments became more sensitive and a wealth of SNe Ia surveys were carried out, many peculiarities and deviations were discovered in SNe Ia and SNe Ia-like events (see e.g. Li et al. (2011); Livio (2000)). This called into question just how homogeneous SNe Ia are.

In addition, the one-dimensional (1D) pure deflagration Chandrasekhar-mass models of Nomoto et al. (1984) that produced nucleosynthetic structures in good agreement with observations (in particular, the W7 model) failed to reproduce such success in more realistic multi-dimensional studies. In a pure deflagration, central ignition of carbon burning proceeds as a sub-sonic flame

front. Sub-sonic fronts permit thermodynamic interactions between the front and the material it is expanding into that are not possible with trans-sonic fronts, i.e. detonations. This results in substantially different nucleosynthetic yields which in turn will result in different observables. With ideally chosen ignition configurations, three-dimensional (3D) pure deflagration models can achieve reasonable agreement with events from the weaker side of the normal SNe Ia spectrum, but cannot easily account for the full range of normal SNe Ia (Fink et al., 2014).

The single-degenerate, Chandrasekhar-mass progenitor model is by no means rendered unworkable. Many viable Chandrasekhar-mass progenitor models are being actively researched, such as delayed-detonation (e.g. Khokhlov (1991); Röpke and Niemeyer (2007)) and gravitationally confined detonation (e.g. Jordan et al. (2008); Plewa et al. (2004)). In a delayed-detonation model, central ignition begins as a deflagration that then transitions into a detonation at some critical density. This model was proposed to address the failure of 3D pure deflagration or pure detonation (central ignition proceeds immediately as a detonation) models to reproduce normal SNe Ia observables. While delayed-detonation models can produce some reasonable observables, they suffer from a lack of a physical understanding of the transition density. In practice, this density is a model parameter that requires some tuning to yield good observables — not a desirable feature in a robust progenitor model. It remains unclear exactly what value for this critical density is realized in Nature, or if there is a physical justification for the value being consistently what is needed to generate SNe Ia events like those we[1] observe. A gravitationally confined detonation begins as a central deflagration that generates a buoyant bubble of hot material. The bubble breaks out at the surface of the WD, triggering a pressure wave that laterally accelerates fuel over the surface. The accelerated surface material is confined by gravity, travels the circumference of the star, and collides in on itself opposite the site of breakout. This collision then triggers an off-center detonation, leading to a SNe Ia.

Theoretical challenges and observational diversity have not ruled out this historically favored single-degenerate model, but have resulted in focus shifting from finding "the" progenitor of SNe Ia to studying a variety of progenitors. The goal has become to determine which models might be the dominant channel(s) leading to normal SNe Ia as well as which might lead to peculiar, sub-, and super-luminous SNe Ia. A heterogeneous progenitor population may also

[1]In this dissertation "we" is used over "I," despite this being a single-author work. In part, this is because some chapters, as discussed in the Acknowledgements, are adapted from a published paper with multiple authors. Using "we" throughout provides some consistency. In addition, this convention is not uncommon in the field. Many single-authored works on the arXiv make use of "we."

help explain apparent correlations of SNe Ia observables with host galaxy type and cosmological redshift (e.g. Calder et al. (2013), especially §1.3, and references therein).

Recently, a particular progenitor has seen a resurgence of interest: the double detonation sub-Chandrasekhar-mass (sub-M_{Ch}) progenitor model (Nomoto, 1980; Woosley et al., 1980; Nomoto, 1982a; Livne, 1990; Livne and Glasner, 1990, 1991; Woosley and Weaver, 1994). The model consists of a sub-M_{Ch} carbon/oxygen white dwarf (CO WD) with a surface layer of helium accreted from a helium-rich companion such as the helium-burning core remnant of an evolved companion star. Over time enough helium is accreted to establish ignition conditions. To proceed as a viable type Ia progenitor this helium ignition must develop as a detonation that then triggers a second detonation in the CO core, hence double detonation. The secondary detonation of the CO core is proposed to occur as either "edge-lit" or central (e.g. Livne and Glasner (1991); Livne and Arnett (1995)). In the edge-lit case the helium detonation transitions into a carbon detonation at the core/shell interface. In the central case the carbon detonation is ignited near the center of the CO core by radially propagating shockwaves generated by the helium detonation's burning front.

This progenitor system addresses a long-standing shortcoming of the conventional Chandrasekhar-mass single-degenerate model. Population synthesis models and observational data suggest there are not sufficient CO WDs near the Chandrasekhar mass in binaries to account for the observed SNe Ia rate, nor can they easily account for the expected delay time distribution (see §3 of van Kerkwijk et al. (2010) and references therein). Ruiter et al. (2011) have argued sub-M_{Ch} and double-degenerate merger progenitors allow for rates and delay times consistent with observation if they can indeed lead to a normal SNe Ia.

As well as being promising normal SNe Ia progenitor candidates, sub-M_{Ch} systems can yield a variety of other explosive events. Helium deflagrations are one possible outcome of surface nuclear burning. A reasonable number of helium shell deflagrations may account for the observed abundance of ^{44}Ca, which is difficult or impossible to produce in major sites of nucleosynthesis like Chandrasekhar-mass SNe Ia models and core-collapse SNe (§4.3 and 6.1 of Woosley and Kasen (2011)). A new class of faint transient, ".Ia's," may be produced by some system configurations (Bildsten et al., 2007), with work to date favoring systems with core masses $\lesssim 0.8\ M_{\odot}$. Lower mass systems may also explain sub-luminous classes of SNe Ia's, such as "Iax's" (Wang et al., 2013). Finally, lower mass systems may yield helium novae (Woosley and Kasen, 2011). In fact, a nova has been observed with no evidence of hydrogen: V445 Puppis (Kato et al., 2000; Ashok and Banerjee, 2003; Iijima

and Nakanishi, 2008). This event should be modeled by theorists to see if sub-M_{Ch} models yielding helium novae can produce observables similar to V445 Pup. It is not well understood yet what aspects of lower mass systems will determine which type of transient manifests, but likely culprits include the composition and mass of the helium-rich shell as well as the thermal structure of the core (especially the temperature near the surface).

Sub-M_{Ch} models with thick helium shells ($\gtrsim 0.1 M_{\odot}$) have been extensively investigated. The introduction of Woosley and Kasen (2011) lays out the history of this work. These investigations led to the sub-M_{Ch} model falling out of favor as a SNe Ia progenitor candidate. The synthetic spectra generated by these models did not match observed spectra of normal SNe Ia, largely due to helium shell effects (Hoeflich and Khokhlov, 1996; Nugent et al., 1997).

The recent resurgence in interest seems to have been sparked by a 2007 letter's (Bildsten et al., 2007) study of AM CVn binaries and proposal of faint ".Ia" events[2]. The proposed .Ia progenitor consists of a CO WD accreting from a helium-rich companion. A prominent example of such a system are AM Canum Venaticorum (AM CVn) binaries (Warner, 1995; Nelemans, 2005). In their letter, Bildsten et al. (2007) calculate that under the right conditions the thermonuclear timescale in an AM CVn's helium envelope can approach the dynamical timescale, possibly establishing conditions for a detonation which consumes the envelope but leaves the WD core intact. This yields a relatively faint transient a tenth the brightness of a normal SNe Ia. A unique aspect of these calculations is the unprecedentedly low ignition pressures, which is related to the unprecedentedly low masses of the helium envelopes considered. Previous work considering similar systems in the context of double detonations assumed higher shell masses (Nomoto, 1982a; Livne, 1990; Livne and Glasner, 1990, 1991; Woosley and Weaver, 1994; Woosley et al., 1986; García-Senz et al., 1999)[3]. The letter, however, presents calculations suggesting minimal mass helium shells, down to $M_{sh} \approx 0.004\ M_{\odot}$, could achieve runaway conditions.

As suggested by the authors of the letter, many took on the task of a detailed reexamination of these systems with lower mass helium shells. A particularly broad and detailed reexamination was carried out by Woosley and Kasen (2011). As they demonstrate, sub-Chandrasekhar mass (sub-M_{Ch}) CO WDs with low-mass helium shells can yield a variety of explosive phenomena, including helium novae, double detonations, and deflagrations that consumed the envelope, leaving behind a hot core. The potential to produce such a variety of transient events motivates extensive theoretical inquiry, especially as

[2]the decimal point is meant to indicate the events are about a tenth the brightness for a tenth the time of a normal type Ia supernova

[3]excepting a data point in Nomoto (1982b) and artificial detonations in Livne and Arnett (1995)

we approach first light for the Large Synoptic Survey Telescope (Ivezic et al., 2008). A great deal of this inquiry has been carried out, with tantalizing results.

Much of the focus has been on these systems as double detonation SNe Ia progenitors. Detonation of the CO core appears to be very robustly triggered by compression waves if detonation occurs in the helium shell (Woosley and Kasen, 2011; Fink et al., 2007, 2010; Shen and Bildsten, 2014), even in the case of asynchronous, asymmetric ignition points (Moll and Woosley, 2013). This makes sub-M_{Ch} promising candidates as SNe Ia progenitors. Thin helium shells have been shown to be capable of carrying sustained detonations, and may even contribute to features found in SNe Ia observations (Townsley et al., 2012). Synthetic spectra and light curves indicate that if the CO core detonates and dominates over helium shell effects in the observables, many sub-M_{Ch} progenitor systems are promising candidates for normal SNe Ia (Woosley and Kasen, 2011; Fink et al., 2010; Sim et al., 2010; Kromer et al., 2010). In particular, work to date favors CO cores that are near 1.0 $M_{\odot}$ and hot if they are to produce normal SNe Ia. The core-only (no He shell) explosions of Sim et al. (2010) agree best with normal SNe Ia properties for CO WDs near 1.0 $M_{\odot}$. The 1D models and subsequent synthetic observables of Woosley and Kasen (2011) agree best with normal SNe Ia's for their "hot" models in which the core relaxed to a luminosity of 1.0 $L_{\odot}$ as compared to their "cool" models, relaxing to 0.01 $L_{\odot}$ before helium accretion was modeled.

Further, delay time distribution calculations based on binary population synthesis find distributions and rates for sub-M_{Ch} SNe Ia progenitor models consistent with being at least one plausible dominant channel for reproducing distributions and rates based on observations (Ruiter et al., 2011). Similar calculations focusing on a subset of sub-M_{Ch} progenitors find they may be the progenitors of SNe Iax (Wang et al., 2013) (though see also Liu et al. (2015a,b)). Geier et al. (2013) present observational evidence for both a helium-accreting sub-M_{Ch} progenitor system and a high velocity helium-rich star that matches the expected properties of the unbound companion star following a sub-M_{Ch} SNe Ia. Brown et al. (2011) analyze a sample of WD binary systems including extremely low mass WDs in the context of AM CVn binaries and sub-luminous SNe Ia. They calculate merger rates that are comparable to the observed rates of sub-luminous SNe Ia. Drout et al. (2013) compare observations of SN 2005ek with many possible models, including sub-M_{Ch} systems. In particular they argue that if SN 2005ek did have a sub-M_{Ch} progenitor, an edge-lit detonation would be the most viable model. In the edge-lit scenario, the detonation in the helium layer propagates directly into the core, setting off a carbon detonation at the core/shell interface.

We caution that theoretical studies (and this dissertation) have limits and make simplifying assumptions. The importance of realistic compositions and convective mixing have been made clear (Kromer et al., 2010; Shen and Moore, 2014; Piro, 2015). In addition we note that it is currently computationally impossible in any model of the full core/shell system to fully resolve ignition of core detonation, which occurs on 0.01 – 1 cm scales for densities $\rho = 10^7$–10^8 g cm^{-3}. Instead, such work must report the critical conditions achieved in a given computational cell or group of cells and argue the likelihood of them achieving ignition of detonation. This challenge is in part addressed in Shen and Bildsten (2014), who carry out small-scale, fully resolved calculations of detonation ignition in regimes relevant to the CO core of sub-M_{Ch} systems. They argue that conditions reported in multi-dimensional studies of the full core/shell system are sufficient for ignition in many cases, though lower mass (roughly, below 0.8 $M_\odot$) or O/Ne cores are less likely to experience ignition.

A significant uncertainty remains. Only one-dimensional (1D) models have demonstrated development of a detonation in these lower mass helium shells. Multi-dimensional work has focused on assuming ignition of detonation and exploring the consequences. In this dissertation, we hope to begin to fill this gap by modeling the development of ignition and elucidating the detailed 3D properties of the system leading up to and at the moment of such an ignition. In addition, we report on our development of accelerated algorithms for integrating nuclear reaction networks. These algorithms will enable investigation of new physics by making it feasible to model more isotopes and reactions *in-situ* in 3D.

Previous work details our initial methodology, carries out numerical experiments, and demonstrates the development of a localized runaway in a model with a 1.0 $M_\odot$ core and 0.05 $M_\odot$ shell (Zingale et al. (2013), hereafter Paper I). This dissertation applies and expands our methodology to a large number of models at higher resolution, carries out a new numerical experiment, and develops new analyses, diagnostics, and conclusions.

With an eye toward carrying out more complex models with larger reaction networks, we develop an accelerated version of Maestro's nuclear reaction modules targeting GPUs (graphical processing units) via the open standard OpenACC. Calculating reaction rates can easily take 10-30% of each timestep, even when modeling a small number of isotopes. By accelerating this algorithm, we will make it computationally feasible to increase the number of isotopes modeled.

In this work we hope to

- expand our methodology to a much larger suite of models,
- explore what broad outcomes and trends we find for simple initial models,

- characterize the bulk properties of these models, including global 3D structure, 1D averages of the 3D state, and peak global properties such as the properties of the hottest cell in the domain,
- develop a methodology for investigating localized runaways that may lead to detonation in the helium shell, and to
- enable the exploration of new physics with the development of accelerated nuclear burning algorithms.

The methodology and model set are described In Ch. 2. Ch. 3 analyzes the bulk properties of our models, including time-series evolution, an overview of outcomes, and the global 3D character. We follow in Ch. 4 with a more detailed focus on characterizing localized runaway events, which would be the site of any potential helium shell detonation. Ch. 5 details our experience accelerating our nuclear reaction algorithms. We conclude in Ch. 6.

Chapter 2

Methodology & Models

2.1 Maestro

Our simulations are performed using Maestro, a finite-volume, adaptive mesh stellar hydrodynamics code suitable for flows where the fluid speed is less than the sound speed, i.e. low Mach number flows. The code's history is one of increasing scale, from small-scale models of astrophysical flames and fluid instabilities to models of entire stars approaching a violent thermonuclear death.

If one must choose a starting point for this history, a reasonable one is the development of an algorithm for modeling low Mach number reacting fluid flows in the context of terrestrial combustion (Day and Bell, 2000). This methodology was generalized to utilize arbitrary equations of state (Bell et al., 2004a), such as those required to model degenerate matter found in many astrophysical systems such as WDs. This generalized code enabled novel explorations of small-scale astrophysical flames and related instabilities. In Chandrasekhar-mass progenitor models of SNe Ia, it is typically assumed carbon-burning is triggered near the center of the WD. This includes a simmering phase during which burning occurs but does not yet manifest as a runaway event disrupting the entire star. This burning yields sub-sonic flame fronts. Such fronts propagating in a fluid medium can experience many instabilities and their evolution will establish the initial conditions under which any runaway event will develop. Thus, they are essential to understanding this progenitor model. Instabilities have complex and sometimes contradictory impacts on the propagation and evolution of a flame. For example, an instability may accelerate a flame's propagation while also wrinkling it, leading to deceleration. Such complicated, nonlinear evolution of sub-sonic flames demands the sort of computational modeling enabled by the generalized low Mach method-

ology. Such modeling was carried out to investigate Landau-Darrieus instabilities (Bell et al., 2004b), Rayleigh-Taylor instabilities (Zingale et al., 2005), and flame-turbulence interactions (Aspden et al., 2008) under conditions relevant to central burning in WDs.

Why develop this low Mach methodology? The motivation comes from fundamental numerical limitations on explicit methods[1] for solving partial differential equations like the reactive Euler fluid equations modeled here. In their conventional form as commonly implemented in astrophysical codes, the compressible Euler fluid equations capture the evolution of pressure, meaning they capture the evolution of sound waves. Numerically, this limits the largest timestep (dt) that can be evolved depending on the soundspeed (c_s). If the system of interest involves sub-sonic flows, this is often a devastating cost. Though we are primarily interested in a flow moving well below c_s, we must nonetheless evolve with dt's restricted by c_s.

The low Mach method works around this by decomposing the pressure into a background pressure and a perturbative pressure, the ratio of which is on the order of M^2, where M is the Mach number. This decomposed pressure is plugged into the Euler equations, all terms up to M^2 are retained, and an asymptotic analysis is applied where $M \rightarrow 0$. From this a low Mach formulation of the Euler equations are derived. Effectively, this represents filtering out sound waves from the system. Another way to think of it is that all pressure waves are instantaneous. Now our dt is limited by the fluid flow speed instead of the soundspeed, allowing for much larger dt's. These large timesteps make it computationally feasible to model low Mach flows over relatively long timescales.

To be specific, for a simulation in which the peak Mach number is M_{peak}, the low Mach algorithm can take a timestep $1/M_{\text{peak}}$ larger than a compressible algorithm if we assume the location of the peak Mach number coincides with the location of the peak soundspeed. In reality, this is rarely the case, so we can treat this as a conservative bound. Peak Mach numbers in the simulations to be reported in this dissertation vary quite a bit, but roughly they range from about 0.01 to 0.2 for most of the evolution, tending toward the lower value at the beginning of a simulation and toward the larger value as the simulation approaches a runaway event. Thus we can take timesteps from at least 10 up to 100 times larger than a compressible code could.

The success of applying the low Mach algorithm to small-scale astrophysical flames inspired the development of what is now Maestro. Though the

[1] in this context, explicit methods refer to those which approximate derivatives using the system's current state to solve for a new state after some time dt, in contrast to implicit methods which are formulated in terms the new state

core set of equations were in place, much development was needed to make models of entire stars possible. This development is described in a series of papers in conjunction with an investigation of the initial science target for large-scale models: understanding the simmering phase preceding runaway in Chandrasekhar-mass SNe Ia progenitor models. Almgren et al. (2006a) initiate the series by developing the hydrodynamics, neglecting for the moment the complexities of thermal conduction and nuclear reactions. The background pressure of Bell et al. (2004a) is reformulated as a 1D stratified state satisfying hydrostatic equilibrium. The methodology is compared with the results of compressible codes as well as fully incompressible methods in regimes where each is valid. Agreement is verified up to a Mach number of 0.2. A key aspect of the low Mach equations developed is allowing for large density and temperature fluctuations. Incompressible methods exist that also filter out sound waves, such as anelastic approximations. However, these methods are derived with an assumption of small density and temperature fluctuations, which would not allow for modeling the vigorous nuclear burning found in systems like SNe Ia progenitors. Further, formulating the equations in terms of a stratified base state allows the method to satisfy hydrostatic equilibrium which can respond to the evolution of the star. Almgren et al. (2006b) introduces heat release and a time-varying background state (allowing for hydrostatic adjustments as a star evolves). Almgren et al. (2008) incorporate nuclear reactions via Strang operator splitting. Finally, Zingale et al. (2009a) introduce spherical geometry to enable modeling a full star, whereas the previous work assumed a plane-parallel geometry. These establish the essential elements of Maestro. However, the code is under active development. Nonaka et al. (2010) is the most recent and comprehensive in the series, and introduces adaptive mesh refinement (AMR). AMR is a numerical technique in which sub-domains of the full domain being modeled are refined to higher resolutions, allowing one to focus limited computational resources on the most interesting regions. The developers of Maestro are committed to open science. All of the source code, including all the code needed to run the models reported here, are available in a public code repository[2]. The repository includes extensive documentation noting the latest developments.

2.2 Initial Methodology

This work builds on and expands a methodology developed by Zingale et al. (2013) (Paper I). Much of this is reviewed in subsequent sections. Here we

[2]https://github.com/BoxLib-Codes/MAESTRO

will briefly overview the key results and tests of this prior work that are not treated in other sections.

This methodology represents the first time Maestro has been applied to modeling the convection of a shell as opposed to core convection (e.g. Nonaka et al. (2012) and references therein). Paper I assesses the robustness of Maestro in this configuration. Convergence is demonstrated at the base resolution of 256^3, the same used in this work for all models except for the full star model (see § 2.5), though the effective resolution of the octants in the full star model is 256^3. The impact of varying model parameters such as boundary parameters is examined (see § 2.6). The algorithm for building initial models is detailed (§ 2.4), including the tanh-smoothing used to transition from the WD core to the He shell. Initialization of a random convective velocity field in the shell is described. This is necessary to avoid unphysical runaway events due to convection not being established, but the initial field will have no imprint on the final field that develops from the self-consistent evolution. The paper demonstrates the evolution of the model is driven by nuclear burning by comparing with a simulation in which burning is turned off. Finally, some initial results are demonstrated, including a runaway event. All of this is done for a single set of core and shell masses: 1.0 $M_\odot$ and 0.05 $M_\odot$.

As will be elucidated in sections to come, the work detailed in this dissertation not only applies this methodology to an uncommonly large suite of models for the field of 3D astrophysical simulation, but expands the methodology. Two levels of refinement are added to further resolve the thin shells, which is especially important for core masses greater than 1.0 $M_\odot$, which have smaller spatial extents in accordance with the WD equation of state. This is the first time Maestro has been used with this amount of refinement in spherical geometry. A new parameter variation is also carried out to explore the impact of the chosen transition width, δ (see § 2.4).

2.3 Microphysics

We utilize a general, publicly available stellar equation of state (Timmes and Swesty, 2000; Timmes, 2008). Ions, radiation, degenerate and relativistic electrons, and Coulomb corrections are all incorporated.

Our nuclear reaction network is quite simple for the sake of computational efficiency, enabling a broad sampling of parameter space (see §2.7). It is important to note that Woosley and Kasen (2011) emphasize two crucial reactions for exploring sub-M_{Ch} systems: $^{14}N(e^-,\nu)^{14}C(\alpha,\gamma)^{18}O$ (NCO) and $^{12}C(p,\gamma)^{13}N(\alpha,p)^{16}O$ (CagO-bypass). Additionally, Shen and Bildsten (2009) demonstrate the importance of $^{14}N(\alpha,\gamma)^{18}F$ (NagF). While we agree these re-

actions are crucial to understanding sub-M_{Ch} models, we can neglect them for the purposes of exploring the dominant energetics in the pre-ignition burning. We employ a simple network consisting of the isotopes ^{12}C, ^{4}He, and ^{16}O and the rates for 3 ^{4}He $\rightarrow^{12}$C (triple-alpha) and ^{12}C(α,γ)^{16}O (CagO). CagO is included because it can allow for the tracing of ^{16}O production which in turn traces the sites of vigorous burning and how polluted the shell becomes with burning products.

Our baseline reaction rates come from Caughlan and Fowler (1988), with screening as in Graboske et al. (1973); Weaver et al. (1978); Alastuey and Jancovici (1978); Itoh et al. (1979). The CagO reaction rate is scaled by a factor of 1.7, as recommended in Weaver and Woosley (1993); Garnett (1997). Thermodynamic derivatives are held constant over a single timestep as described in Almgren et al. (2008).

2.4 Initial Models

Maestro evolves both a 1D hydrostatic base state and a 3D hydrodynamic state. For spherical problems, such as the sub-M_{Ch} system, this base state is radial. To set the initial conditions for our 3D problem we initialize the base state and map that state onto the 3D grid. Our sub-M_{Ch} initial models are defined by five parameters: the mass of the WD core, M_{WD}, the isentropic helium shell's mass, M_{He}, the temperature at the base of the helium shell, T_{base}, the core's isothermal temperature, T_{core}, and a characteristic scale δ setting the width of the transition from core to shell material. At the interface between the core and the shell there is a composition and temperature gradient following the prescription described in §2.2 of Paper I, which defines this δ. We generate our own initial models using an iterative scheme that enforces hydrostatic equilibrium and the values of T_{core} and T_{base} while converging on the given (M_{WD}, M_{He}). Figure 2.1 demonstrates a representative initial model.

We expand upon Paper I by adding a new parameter test. Most initial models for the simulations reported here use the same transition width parameter δ as in Paper I: $\delta = 50$ km. However, we carried out a supplemental suite of simulations for model 11030 (see §2.7) in which all parameters are the same save for a δ one-fourth, one-half, and twice the original magnitude of 50 km. The quadrupled resolution on a side in this paper allows us to resolve sharper transitions than in Paper I. This tanh-smoothing is necessary for the problem to be well-posed. A sharp discontinuity in grid-based hydrodynamics makes it impossible to demonstrate convergence as it offers no resolvable solution to converge to (see Paper I for convergence tests). The 50 km value for δ in the lower resolution models of Paper I provided roughly 10 cells of radial

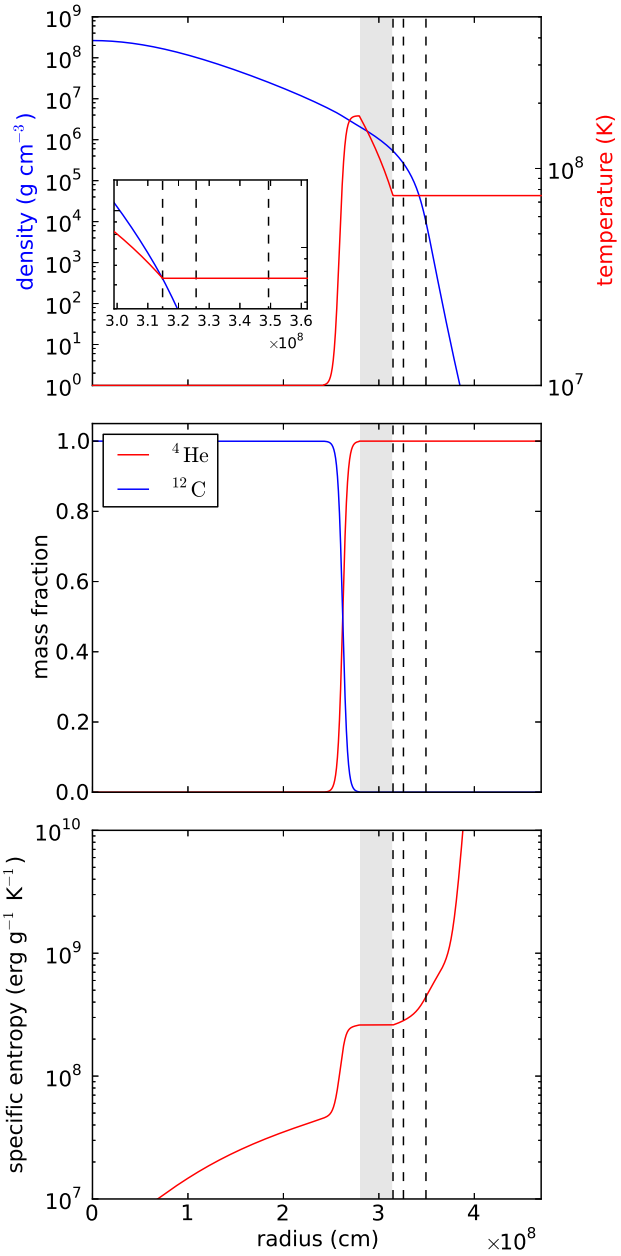


Figure 2.1 A representative initial model with $M_{\rm WD} = 1.2\ M_\odot$, $M_{\rm He} = 0.05\ M_\odot$, $T_{\rm core} = 10^7$ K, $T_{\rm base} = 1.75 \times 10^8$ K. The shaded region is the convection zone. The dashed lines from left to right are: the start of the sponge, the anelastic cutoff, and the base cutoff density (see §2.6). We let $r = 0$ be the center of the star. **Top:** Temperature (red) and density (blue) profiles. The inset zooms in on the sponge and cutoff radii. **Middle:** Mass fraction profiles of carbon (blue) and helium (red). **Bottom:** Specific entropy profile.

resolution over which to resolve the transition. The lowest δ examined in this paper, 12.5 km, offers similar resolution of the transition. In addition, it is not well-known exactly how transitions from core to shell are realized in nature. Thus, this parameter study has both a numerical and physical motivation. See § 3.5 for more discussion and figures.

2.5 Grid Structure

The 3D grid is Cartesian. For all models except one (see §2.7) an octant of the sub-M_{Ch} WD is modeled, allowing us to capture 3D effects yet achieve much greater computational efficiency and explore a large number of models. The impact of simulating an octant instead of the full star is investigated in Paper I. As we discuss in §3.1, the higher resolutions and larger model set presented here introduce complications at the boundaries for octant runs with localized runaway.

The grid is adaptively refined to focus resolution and computational power on the regions of greatest interest. To study the dynamics of the convection and nuclear burning in the helium shell, we refine zones in which $X_{He} > 0.01$ at a density greater than ρ_{cutoff} (see §2.6). To better resolve the shells, in this study we further refine cells with temperatures $T > 125$ MK. We are satisfied with two levels of refinement for models with a 0.8 $M_{\odot}$ core. The mass-radius relationship for WDs means these models will have the largest radius and consequently the thickest shells spatially. For models having $M_{core} \geq 1.0\ M_{\odot}$, the spatial extent of the shell is greatly reduced, which is exacerbated by the fact that more massive cores have lower mass helium shells. Thus, for all such models we add an extra level of refinement for a total of four levels (the base grid, which we label level one, and three additional, further refined levels).

Figure 2.2 illustrates the grid we have described. This figure outlines grid patches[3] for each of the levels in the initial grid for model 10040H (for details about our adaptive mesh algorithm and the definition of grid patches, see §5 of Nonaka et al. (2010)). At the coarsest (base) level, all octant runs have a 256^3 resolution with a refinement factor of 2 between levels, leading to subsequent 512^3, 1024^3, and 2048^3 effective resolutions within the refined patches. We include one full star run (08130F, see §2.7), which has a 512^3 coarse (base) resolution. The strong dependence of radius on mass in WDs results in a range of physical resolutions $\Delta x \approx 2.5$ - 15.8 km at the finest level. The 1D base state's resolution is not adaptively refined; instead, it has a fixed resolution of

[3]Note that these patches are *not* cells, but rather 3D rectangular grids containing many cells. The domain is broken up in this fashion so that the computational workload can be distributed across the many nodes of a supercomputer.

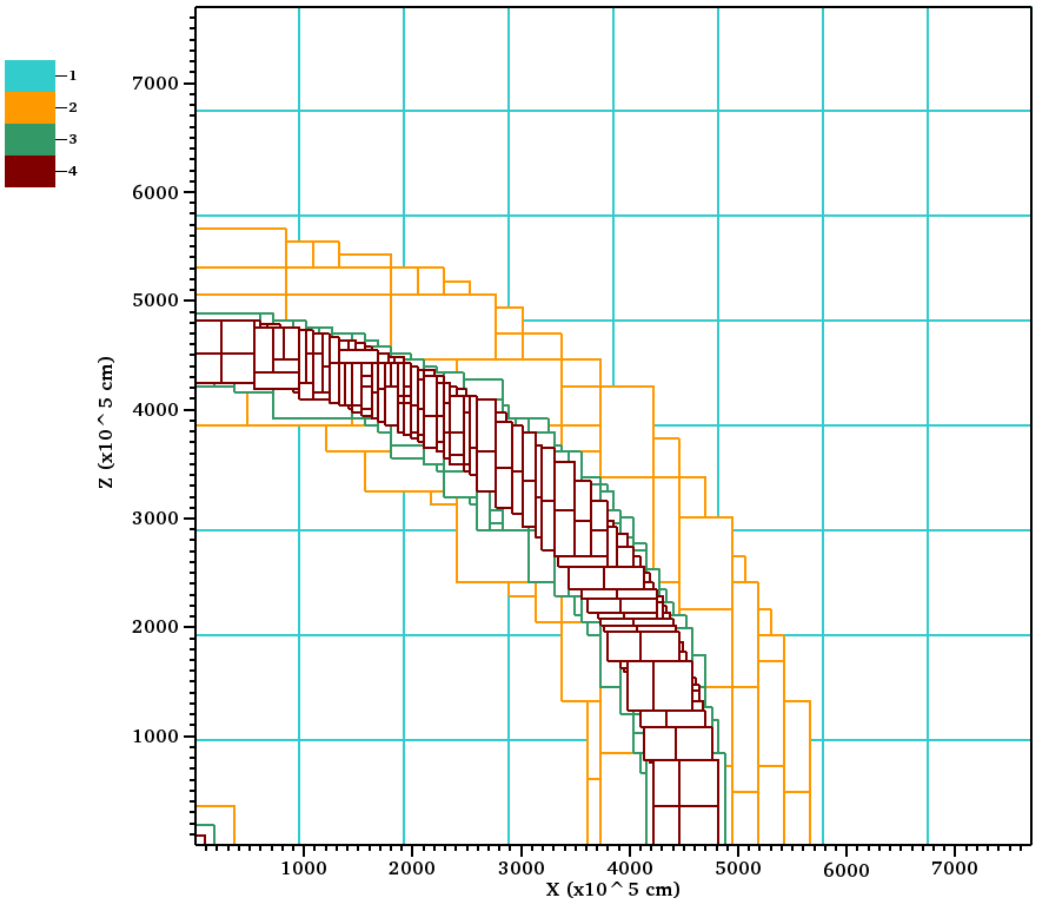


Figure 2.2 A representative slice of the initial grid for $M_{\rm WD} = 1.0\ M_\odot$, $M_{\rm He} = 0.04\ M_\odot$, $T_{\rm core} = 10^8$ K, $T_{\rm base} = 1.85 \times 10^8$ K. The different colors indicate grid patches at different levels of refinement. Level 1 is the base (coarse) level.

five times that of the finest level: 5120 cells. This factor of five is first used and discussed in Zingale et al. (2009b).

2.6 Boundaries

The boundary conditions for our simulations are reflecting on the symmetry faces of octant domains (lower x, y, and z), and outflow (zero-gradient) on the other faces. A full star simulation has outflow boundary conditions on all faces of the domain.

As can be seen in Fig. 2.2, the grid includes a coarsely-resolved region well outside the convective zone. This serves to keep the convective surface insensitive to boundary conditions. A steep drop in density occurs at stellar surfaces (as seen in Fig. 2.1). Without modification, this rapid decline precipitates a rapid spike in velocity to conserve momentum. The advantages of a low Mach method become negligible if fluid velocities in any zone approach the soundspeed. Thus much work has been put into developing strategies to address steep density gradients at stellar surfaces without significantly impacting Maestro's computational or physical validity. The details of these treatments can be found in Paper I (§2.3), Zingale et al. (2011), and Nonaka et al. (2012).

Briefly, two density cutoffs are implemented: the anelastic cutoff $\rho_{\rm anelastic}$ and the low-density cutoff $\rho_{\rm cutoff}$ (see Fig. 2.1). For zones with densities below $\rho_{\rm anelastic}$, Maestro switches to an anelastic-like velocity constraint that helps damp velocities (see Almgren et al. (2008)). Density is held constant once it falls to $\rho_{\rm cutoff}$, halting the steep decline. To prevent impacting validity its value is chosen such that the regions with $\rho \leq \rho_{\rm cutoff}$ contain an insignificant proportion of the system's total mass. These cutoffs are supplemented with a numerical sponge that damps surface velocities (Almgren et al., 2008). Cutoff values for all simulations are discussed in §2.7. Also note that we have a temperature cutoff $T_{\rm cutoff}$ at the top of the convecting region. The more we allow T to continue to drop, the lower the soundspeed and thus the higher the Mach number at the edge of the star, reducing the utility of a low Mach methodology. $T_{\rm cutoff}$ prevents this spike in the Mach number. In Paper I, we discuss $T_{\rm cutoff}$ and demonstrate that the convective dynamics and evolution of interest are insensitive to variations of it.

This combination of cutoffs, a sponge, and maintaining a buffer zone in the computational domain between the stellar surface and the domain's boundaries enables us to study surface convection in detail over long timescales without surface effects significantly impacting our results.

2.7 Model Set

Maestro's ability to take large timesteps as well as the nature of sub-M_{Ch} pre-explosive dynamics make a broad sampling of the parameter space in 3D computationally feasible. What exactly is the parameter space of interest? To determine this we draw on the results of Bildsten et al. (2007) and the many studies they inspired.

The parameters of greatest interest are the core and helium shell mass configurations. The motivating question is how ignition develops and how it is characterized in minimal helium shell mass systems for a range of core masses. Figure 2 of Bildsten et al. (2007) illustrates their determination of the minimum shell masses for which the nuclear burning timescale is on the order of the dynamical timescale for isothermal cores with $T_{\rm core} = 3 \times 10^7$ K. Such a short nuclear burning timescale suggests the possibility of thermonuclear runaway even for thin helium shells with $M_{\rm He} \lesssim 0.05$ – $0.0125\ M_{\rm core}$ for 1.0 – 1.2 $M_\odot$ cores. This work is extended and deepened in subsequent studies, which are largely consistent with the essential results of the 2007 work (Shen and Bildsten, 2009; Brooks et al., 2015). Woosley and Kasen (2011) carry out an extensive set of 1D sub-M_{Ch} calculations and generate an analogous figure (Figure 19). They include the impact of varying $T_{\rm core}$. For $M_{\rm core} = 0.7\ M_\odot$, runaway can occur with helium shells having $\sim 15\%$ of the core's mass, perhaps not sufficiently thin for SNe Ia-like spectra. As $M_{\rm core}$ increases to 1.1 $M_\odot$, runaway can be achieved with shells $\sim 2.25\%$ of the core's mass for hotter cores ($T_{\rm core} \sim 7.5 \times 10^7$ K), making SNe Ia-like spectra more achievable. The bare (no helium shell) 1D sub-M_{Ch} WD detonation calculations of Sim et al. (2010) suggest systems with $M_{\rm core} \gtrsim 1.0\ M_\odot$ can yield observables in reasonable agreement with the range of observed normal SNe Ia while lower $M_{\rm core}$ systems can produce characteristics of observed sub-luminous SNe Ia. Fink et al. (2010)'s 2D calculations also find they can produce many characteristics of the range of observed SNe Ia and that core detonation is triggered by shell detonations for $(M_{\rm core}, M_{\rm He}) = (0.810$ – $1.385,\ 0.126$ – $0.0035)\ M_\odot$.

Given these studies and the uncertainties involved we investigate systems with $M_{\rm core} = 0.8, 1.0, 1.1, and 1.2\ M_\odot$, and a range of shell masses including $M_{\rm He} = 0.02 - 0.13\ M_\odot$. Our mass configurations are summarized in Fig. 2.3 and compared with the minimum shell masses estimated by others. In choosing shell masses we had to balance a desire to model low-mass shells near the lower limit of models that run away in 1D with a need for the simulations to be computationally feasible. Lower mass cores can take many convective turnover times to reach runaway for minimum mass helium shells whereas minimum mass shells can be difficult to resolve for higher mass cores. As a result, we have the points marked in Fig. 2.3 that track near the 1D lower

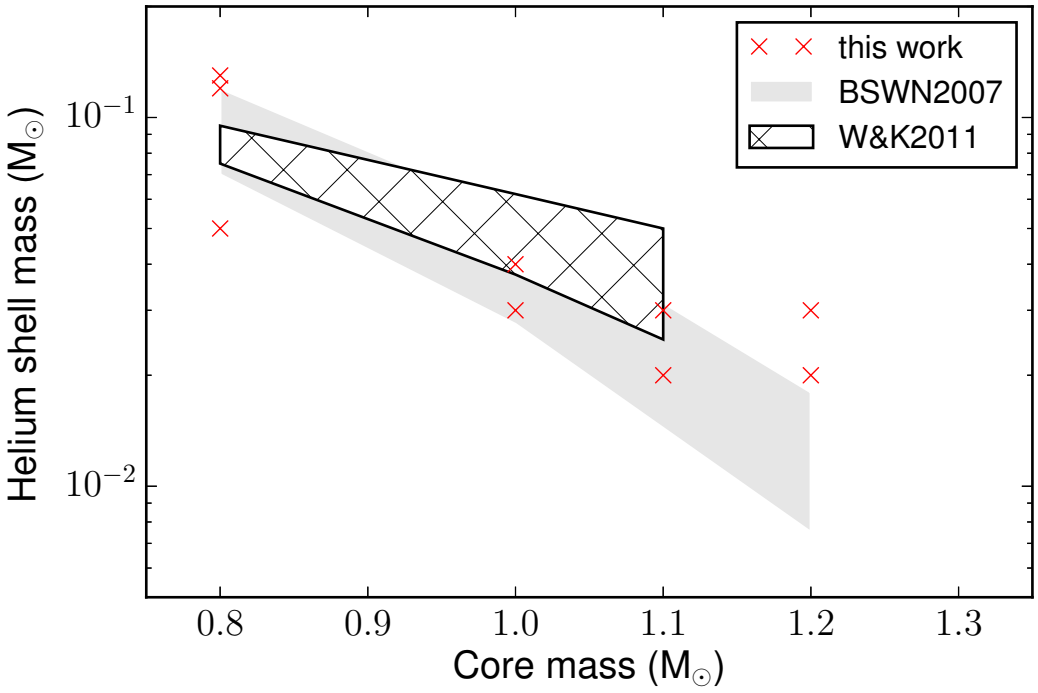


Figure 2.3 The crosses are the core-shell mass configurations modeled in this paper. For comparison, the shaded region is the range of minimum shell masses capable of initiating ignition as given in Fig. 2 of Bildsten et al. (2007) (using their $t_{\rm nuc} = t_{\rm dyn}, 10t_{\rm dyn}$ lines). The hatched region is the range of minimum shell masses that yield either a deflagration or detonation as given in Fig. 19 of Woosley and Kasen (2011). The lower bound is for their "hot" models, while the upper is for "cold" models.

limit but to varying extents.

Due to the importance of $T_{\rm core}$ demonstrated in Woosley and Kasen (2011), we include models with $T_{\rm core} = 10^7, 10^8$ K. Finally, we vary $T_{\rm base}$ from 175 MK to 250 MK. These interface temperatures are intended to be roughly what we would expect a few to several convective turnover times before runaway, based on both our own numerical experiments and the 1D literature. Table 2.1 lists the details of our model set. Note the nomenclature of the labels, which we will reference throughout this work. The first two numbers note the core mass, the next three the shell mass, and an additional "H" indicates the model has a hot core (10^8 K instead of 10^7 K as in the other models). Finally, models ending with "d" and a number indicate the factor by which δ is scaled. For example, 11030Hd0.25 would represent a model with a 1.1 $M_\odot$ core, a 0.03 $M_\odot$

Table 2.1. Model Set

label	$(M_{\rm core}, M_{\rm He})$ $[M_\odot]$	$T_{\rm base}(t=0)$ $[\times 10^8$ K]	$\rho_{\rm base}$ $[\times 10^5$ g cm$^{-3}]$	$x_{\rm max}$ [km]	$\Delta x_{\rm fine}$ [km]	$\rho_{\rm anelastic}$ $[\times 10^5$ g cm$^{-3}]$
12030H[a]	(1.2, 0.03)	1.75	10.1	5300	2.6	1.29
12030	(1.2, 0.03)	1.75	10.8	5100	2.5	1.37
12020H	(1.2, 0.02)	1.75	6.2	5500	2.7	0.80
12020	(1.2, 0.02)	1.75	6.8	5300	2.6	0.87
11030H	(1.1, 0.03)	1.85	5.7	6600	3.2	0.67
11030d0.25	(1.1, 0.03)	1.90	6.0	6400	3.1	0.68
11030d0.5	(1.1, 0.03)	1.90	6.0	6400	3.1	0.68
11030	(1.1, 0.03)	1.90	6.0	6400	3.1	0.68
11030d2	(1.1, 0.03)	1.90	6.0	6400	3.1	0.68
11020H	(1.1, 0.02)	1.85	3.6	6900	3.4	0.43
11020	(1.1, 0.02)	1.85	3.9	6600	3.2	0.46
10040H	(1.0, 0.04)	1.85	5.0	7700	3.8	0.58
10040	(1.0, 0.04)	1.85	5.3	7400	3.6	0.62
10030H	(1.0, 0.03)	1.85	3.5	7900	3.9	0.42
10030	(1.0, 0.03)	1.85	3.8	7600	3.7	0.45
08130H	(0.8, 0.13)	1.85	9.9	8300	8.1	1.15
08130F	(0.8, 0.13)	1.85	10.9	16200	15.8	1.26
08130	(0.8, 0.13)	1.85	10.7	8100	7.9	1.25
08120H	(0.8, 0.12)	1.85	8.8	8500	8.3	1.03
08120	(0.8, 0.12)	1.75	9.6	8100	7.9	1.22
08050	(0.8, 0.05)	2.50	2.6	10100	9.9	0.19

[a]"H" models have 10^8 K isothermal cores, all others are 10^7 K

helium shell, a 10^8 K isothermal core, and a δ scaled by 1/4.

While our models are motivated by the literature they are not necessarily likely to be realized in nature and are not the result of detailed stellar evolution calculations. Our focus is on broadly sampling the parameter space, characterizing the relationships between parameters and possible explosive outcomes, and quantifying the salient trends that emerge. This will guide future work studying particularly interesting parameter configurations using more realistic initial models and detailed nucleosynthesis.

In total, about 70 million core-hours were required to carry out this study along with over 100 petabytes of storage. The amount of computer time needed for each simulation varies depending on many factors, including resolution, the amount of time modeled, and the energetics of the burning. Typical numbers for a single simulation are 500000 – 1500000 core-ours. Note that arriving at the set of models reported on here involved numerical experimentation and exploration of other models, not all of which are reported.

Chapter 3

Bulk Properties of Simple Models

3.1 Outcomes

Our broad sampling of parameter space explores several different model configurations. We find a range of outcomes for these models: localized runaway, quasi-equilibrium, and convective runaway. Table 3.1 denotes the ultimate outcome of each model.

Localized runaways represent possible seeds of deflagration or detonation in the helium shell. All runs in this category have localized volumes of fluid that experience rapid temperature runaway to about 1 GK. They also have peak Mach numbers less than 0.3 before runaway, and less than 0.2 for the majority of the simulated time. Fig. 3.1 plots some of the key properties of interest over time for an igniting run (model 11030). This plot is representative of the general behavior of runs experiencing localized runaway. The plot demonstrates that the temperature of the hottest cell in the domain[1] initially follows a trend similar to that of the laterally averaged peak temperature. As the model approaches runaway the hottest cell increasingly deviates from the background conditions. We also find that the convectively unstable region moves deeper into the star. This suggests that vigorous burning and the convection it drives can result in significant changes in the density and composition of burning sites (discussed more in §3.4).

As one might expect, the radius of the hottest cell moves radially inward along with the base of the convectively unstable region. However, at the end

[1]We track the cell with the largest temperature in the entire domain, but we caution this is not a Lagrangian measure—it is simply the hottest cell without regard to the cell's mass density

Table 3.1. Outcome Summary

label	$t_{\rm final}/\langle\tau_{\rm conv}\rangle$[a] min	avg	max	outcome[b]
12030H	2.3	8.8	12.1	l
12030	1.5	6.4	8.6	l
12020H	3.1	10.3	17.1	l
12020	3.0	12.4	19.4	l
11030H	2.7	10.6	17.5	l
11030d0.25	12.1	4.7	35.9	l
11030d0.5	9.8	8.5	29.1	l
11030	1.9	26.3	13.6	l
11030d2	2.5	32.6	9.4	l
11020H	1.5	3.8	9.0	q
11020	5.2	19.5	27.2	q
10040H	7.1	23.4	27.6	q
10040	4.2	14.9	18.4	q
10030H	1.2	4.6	7.8	q
10030	1.5	6.5	8.3	q
08130H	0.6	3.8	8.4	l
08130F	0.2	1.3	6.3	l
08130	0.6	4.2	10.4	l
08120H	1.8	7.7	10.8	l
08120	8.4	25.7	27.9	l
08050	0.9	2.7	7.1	c

[a]see § 3.4 for how these values are calculated

[b]outcomes are designated as (l) localized runaway, (q) quasi-equilibrium, or (c) for convective runaway. See text for details.

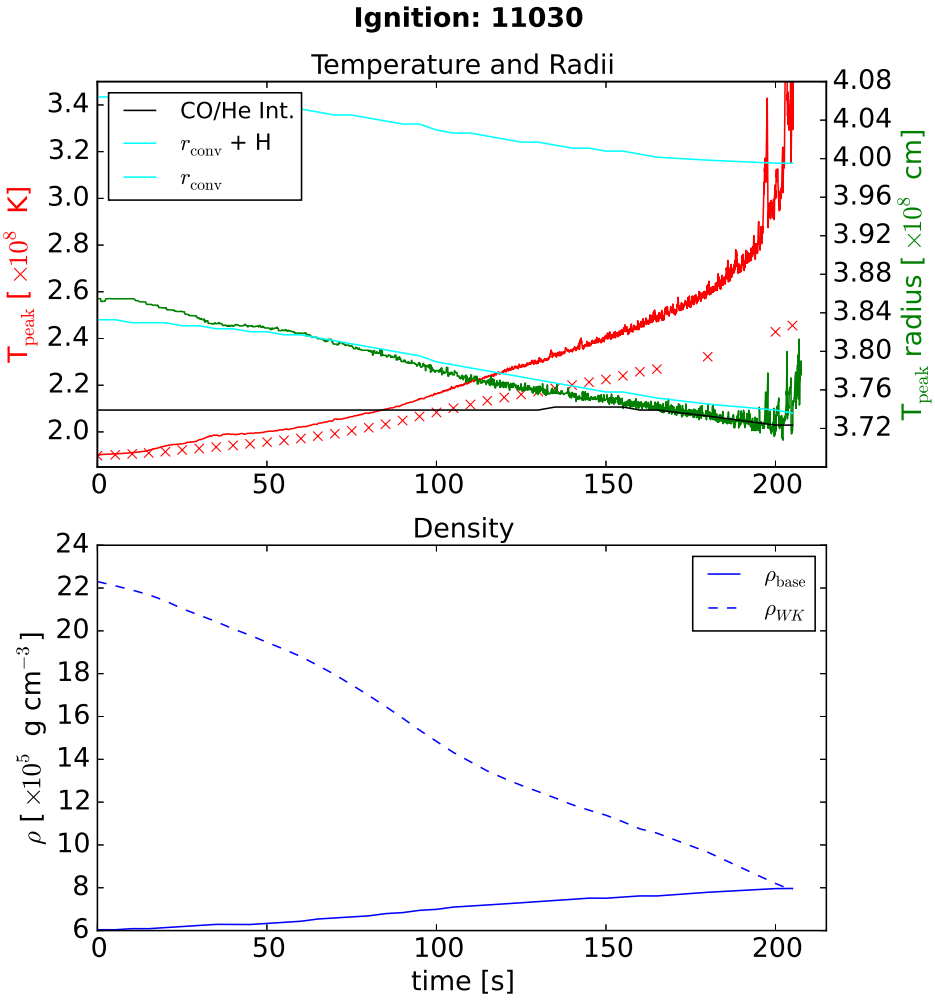


Figure 3.1 Temperature, radius, and density over time for model 11030. **Top:** Temperature is plotted in red. The solid line is the temperature of the hottest cell in the entire computational domain. The x's trace the peak laterally averaged temperature. The green, cyan, and black all plot radii. Green is the radius of the hottest cell in the domain. Black is a trace of the core/shell interface based on the radius at which the average $X_{\rm He}$ composition is 0.9. The cyan plots the base of the convective region and one pressure scale height above this base. The inset is a 2D temperature slice centered on the site of runaway, demonstrating its localized nature. The runaway happens at a boundary, hence half the inset being white (no temperature data outside the boundary). **Bottom:** The solid line plots the laterally averaged density at the radius of peak average temperature. The dashed line is the critical density above which ignition is expected according to Eq. 3.1.

we see that this radius moves outward in many models. If convection is able to transport an ignition seed to a significant height above the core/shell interface then it makes "edge-lit" double detonation models workable. In the edge-lit scenario, carbon detonation of the core is triggered by the propagation of the helium detonation wave at the surface of the core. This model is generally disfavored because it has been shown that it requires the initial helium detonation to go off at a substantial height above the core/shell interface (Woosley and Kasen, 2011; García-Senz et al., 1999). It appears this is necessary for the detonation front to build up enough energy to trigger a sustained detonation at the surface of the CO core. Detonations at lower altitudes may result in some carbon burning, but not enough to trigger a sustained detonation. If convection is more effective than expected at transporting the detonation seed then the edge-lit scenario needs to be considered more seriously.

Unfortunately, many of the localized runaway events in our models happen near the boundary of the octant being simulated. We stress that we have carried out a full star run for the localized runaway model 08130 (and also in paper I for a 10050 model) and still find localized runaway as well as a radius significantly above the interface. We have also carried out simulations in which the temperature of cells is limited to be below 3.5 MK and see that many localized runaways occur far from the boundary, though the initial runaway happens preferentially at the boundary. So while we are confident the localized runaway is not a boundary effect, the elevated radius of the hottest cell in fig. 3.1 cannot be ruled out as a boundary effect in all runs. This is an important issue that will be resolved in the next paper in this series, which focuses on the timing, thermodynamics, and geometry of ignition in this suite of simple models.

In contrast, quasi-equilibrium simulations balance nuclear burning with convective cooling for many convective turnover times (at least an average of 3.8 or more turnovers, see Table 3.1 for a range of turnover estimates and §3.4 for how we calculate these). Fig. 3.2 demonstrates this case for a model we ran for a particularly long time. We cannot say these runs have reached an equilibrium between burning and convective cooling because neither peak nor average base temperatures plateau. If we had the computational resources to run these simulations indefinitely it may be the case they would experience a runaway. What we demonstrate instead is that these models are stable against immediate runaway, i.e. runaway within a few convective turnover times.

We have also included a model similar to Woosley and Kasen (2011)'s model 8HB, which experiences a helium nova. Please note that in some bodies of literature, helium nova has a particular meaning. For the purposes of this work, a helium nova is an event where runaway is more global across the

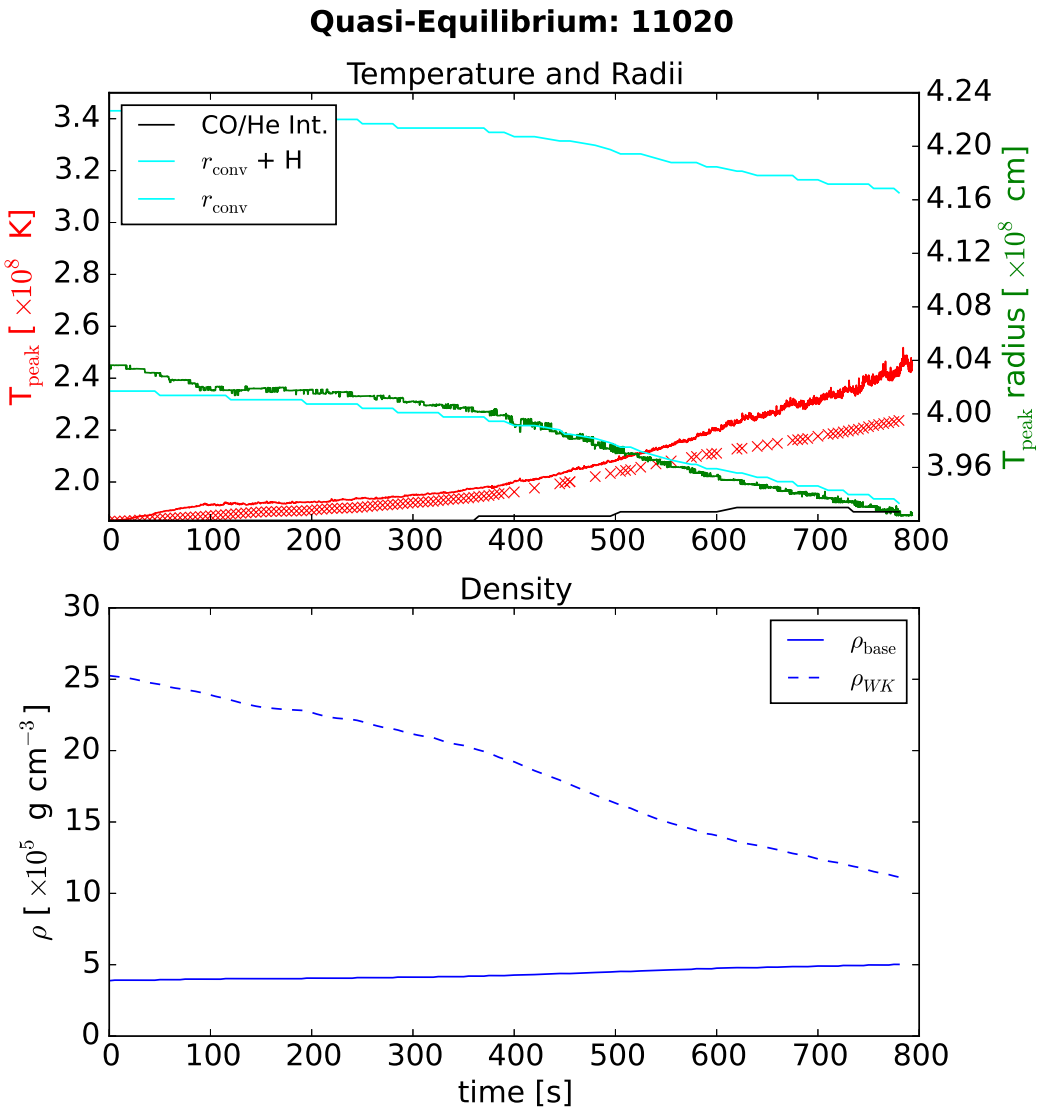


Figure 3.2 The same plot as in fig. 3.1 for model 11020, demonstrating quasi-equilibrium.

surface instead of localized, and may be driven by a runaway of the convective velocity as is the case for our 08050 and Woosley and Kasen (2011)'s 8HB. This model had a higher interface temperature than most runs, 2.5 MK instead of around 1.9 MK, to facilitate reaching runaway conditions without expending more computational resources than necessary. Within the low-Mach limits of Maestro we find convective runaway, even with the elevated interface temperature. As the base temperature increases from burning, the turnover rate of the convective shell is able to increase without plateauing until the Mach number of the fluid gets too large for us to track. This suggests such a thin shell is able to rapidly transport the energy release of nuclear burning. In contrast to localized runaway, this is a more global phenomenon and could develop into something like a helium nova, as argued in Woosley and Kasen (2011). The time series data for this convective runaway is plotted in Fig. 3.3. The existence of two regimes, convective and localized runaway, suggests researchers should investigate the transition from one to the other. The conditions of this transition point will be important for determining the minimum helium shell mass capable of achieving localized runaway.

3.2 Temperature

While the mass of the core and helium shell play a primary role in determining the thermodynamic conditions at the base of the shell, there are secondary determinants. Varying evolutionary histories can result in accreting CO WD primaries of varying temperatures. A history of helium flashes may heat the WD surface. This enables systems with similar mass configurations to have noticeable differences in burning conditions at the core/shell interface.

Our parameterization of the initial model allows us to vary the initial temperature of the actively burning base of the helium shell. However, in our attempts at varying the base temperature we find only a relatively narrow range of options can be feasibly explored with our current methods. Low initial temperatures will either not be able to initiate sufficiently vigorous burning to allow a study of ignition or will establish a trend of growing average base temperature that will steadily build until either ignition or quasi-equilibrium is achieved. However, the computational resources required to reach a dynamically interesting stage of burning with a low initial base temperature can be substantial. Too high a base temperature will either lead to unphysical ignition by not allowing time for convection to be established or will push the convective velocity beyond Maestro's ability to model. Thus, for a given model we choose an initial base temperature low enough to allow for convection to be established and for several convective turnover times of evolution, and high

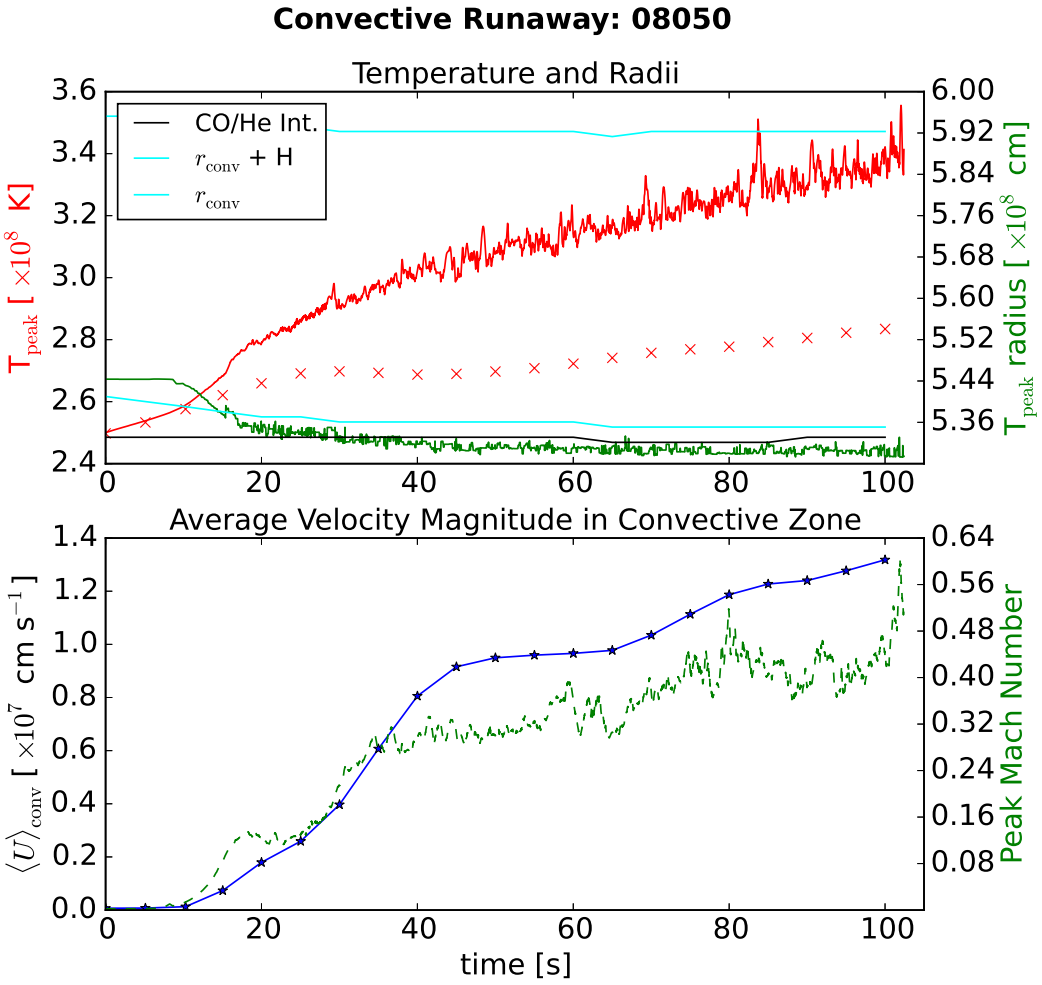


Figure 3.3 The top plot is the same as in fig. 3.1 for model 08050. The bottom plot shows the average velocity magnitude in the convective zone as well as the peak Mach number in the domain.

enough to reach a scientifically interesting stage of burning while using feasible amounts of computational resources.

The influence of the isothermal core's temperature is easier to investigate with our methods. Woosley and Kasen (2011) find that their synthetic spectra and light curves come closest to resembling observations of type Ia's for their "hot" models in which the accreting CO WD relaxes into thermodynamic equilibrium with a luminosity of 1 $L_\odot$ before accretion is modeled. The hotter core enables runaway in thinner shells than the colder core. This motivates our exploration of our own "hot" and "cold" (10^8 and 10^7 K) models (hot models are indicated with an 'H' in their label in Tab. 2.1).

Our results are consistent with that of Woosley and Kasen (2011). Hot cores allow for initiation of localized runaway at lower densities for a given core/shell mass configuration. This is largely due to an expanded core radius in hot runs, and thus a lower density at the core/shell interface where the burning occurs. The lower density favors higher temperatures at runaway as well. In Fig. 3.4 we compare a hot and cold run to demonstrate these phenomena.

3.3 Localized Runaway

We have demonstrated that many of our models achieve localized runaway through bulk diagnostics and time-series data, comparing them to 1D results. This answers a major question we are exploring: is ignition found in 1D codes consistent with 3D models? We argue the localized runaway we find is consistent, though do caution that localized runaway should not be thought of as ignition. The localized runaway reported here may ignite deflagrations or detonations, but there is insufficient evidence and analysis in this section to make a definitive determination. In the next chapter we develop an initial methodology for determining the likelihood of such ignition. For now, we move to broadly characterizing the localized runaway found in our models.

The 1D studies we have discussed necessarily model ignition as simultaneous across a spherical shell. Fink et al. (2007) contribute 2D simulations including a variety of detonation seed geometries, following up later in Fink et al. (2010) with 2D simulations seeding a single detonation in a larger range of core/shell masses including thin shells. Moll and Woosley (2013) have contributed 2- and 3D studies in which detonation is seeded at multiple points with variations in geometry as well as timing. A key conclusion of these multi-D investigations is that detonation in the helium shell very robustly triggers detonation of the CO core via radially propagating compression waves gener-

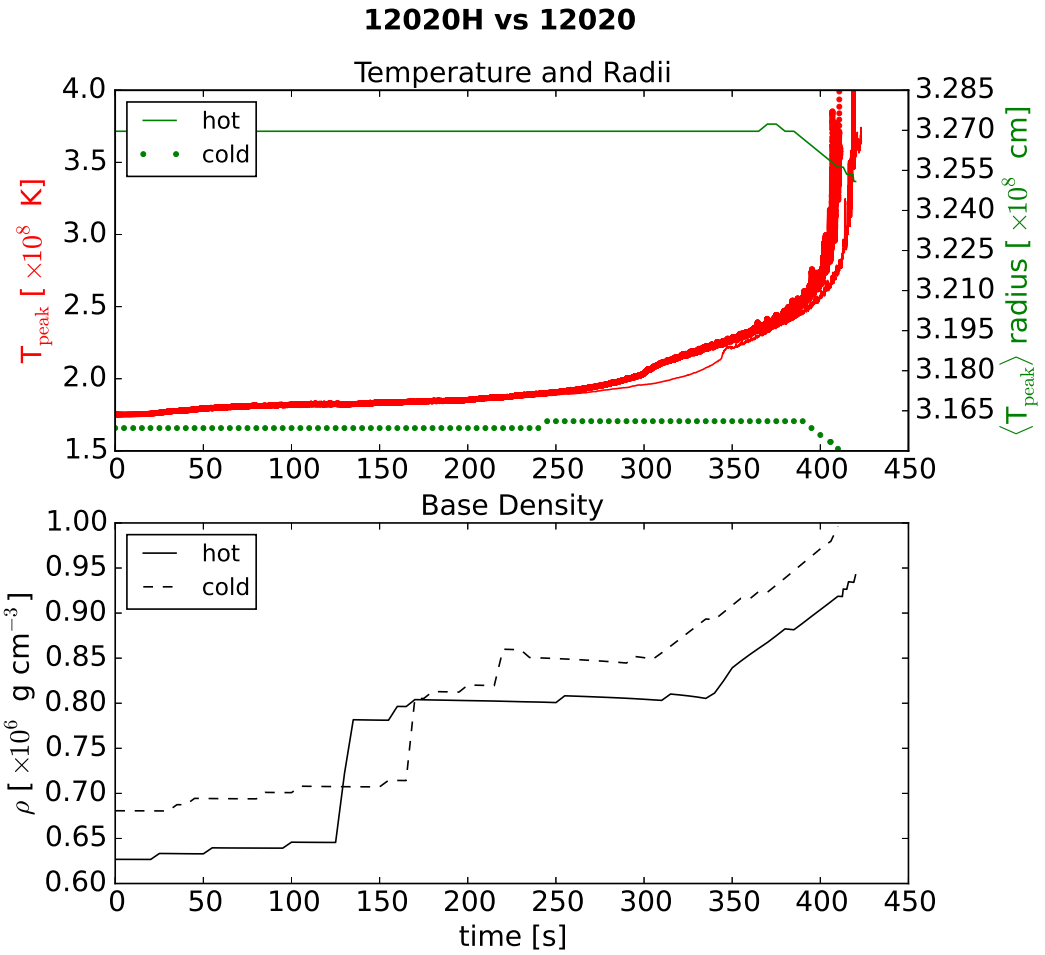


Figure 3.4 Comparison of several properties for a hot (10^8 K isothermal core, solid lines) and cold (10^7 K, dotted and dashed lines) model with a 1.2 $M_\odot$ core and 0.02 $M_\odot$ shell. **Top:** In red we plot the global peak temperature. In green we plot the radius of the temperature peak from a lateral average of the 3D data. **Bottom:** In black we plot the laterally averaged density at the radius plotted in green in the top panel.

ated by the helium detonation's shock front traversing the shell[2].

Assumptions about how many detonations to seed, where to seed them, and their timing impact the ultimate outcome of the double detonation. Single-point helium detonations lead to viewing-angle dependences that will impact synthetic observables, though the dependence becomes weaker for lighter helium shells (Kromer et al., 2010). If detonated at a great enough altitude above the core/shell interface, helium detonations can directly ignite carbon burning instead of detonating indirectly with converging shock fronts (Woosley and Kasen, 2011; García-Senz et al., 1999; Moll and Woosley, 2013). The size and shape of the detonation can also impact how easily core detonation can be triggered, especially for the thinner shells considered to be the most promising candidates for modeling normal SNe Ia (Moll and Woosley, 2013).

In light of this, what do our models suggest? To assess the conditions that foster ignition we track the hottest 0.005 % of cells in the computational domain immediately prior to runaway. Fig. 3.5 plots histograms of the radii and densities of these cells in addition to a spherical projection of their angular locations for model 08130 (see Table 2.1). The spherical projection illustrates the two regions in which volumes of hot fluid develop: at the base of convective inflows and at the intersection of outflows from neighboring convective cells (see §3.4). This is determined by contrasting projections like that in Fig. 3.5 with renderings of convective outflow like that in Fig. 3.6. The density histogram includes a reference for a critical runaway density given by Eq. 8 in Woosley and Kasen (2011):

$$\rho_{\rm cr,WK} = \left(1.68 \times 10^{-4}\ \exp(20/T_8)\right)^{1/2.3}, \tag{3.1}$$

where $T_8 = T/10^8$ is the temperature in units of 10^8 K. This is a rough critical density above which violent, hydrodynamic runaway is expected and below which convection is expected to be efficient enough to transport any energy generated by nuclear burning. For the purposes of this paper, we denote it as a critical density above which we expect localized runaway to be possible.

This density is based on 1D models and thus is only fairly compared to lateral averages of quantities in our 3D simulations. Note that Fig. 3.5 demonstrates localized runaway is achieved even though the hottest cells have densities significantly below this critical density. This is not surprising as the cells trace a 3D model. When we consider laterally averaged quantities, Eq. 3.1 is

[2]It is important to note the carbon detonation in these studies relies upon assuming certain conditions being achieved in a given computational cell will lead to detonation. Current studies of the full core and shell system do not have the resolution to model the initiation of carbon detonation fully self-consistently. See Shen and Bildsten (2014) for detailed carbon detonation calculations.

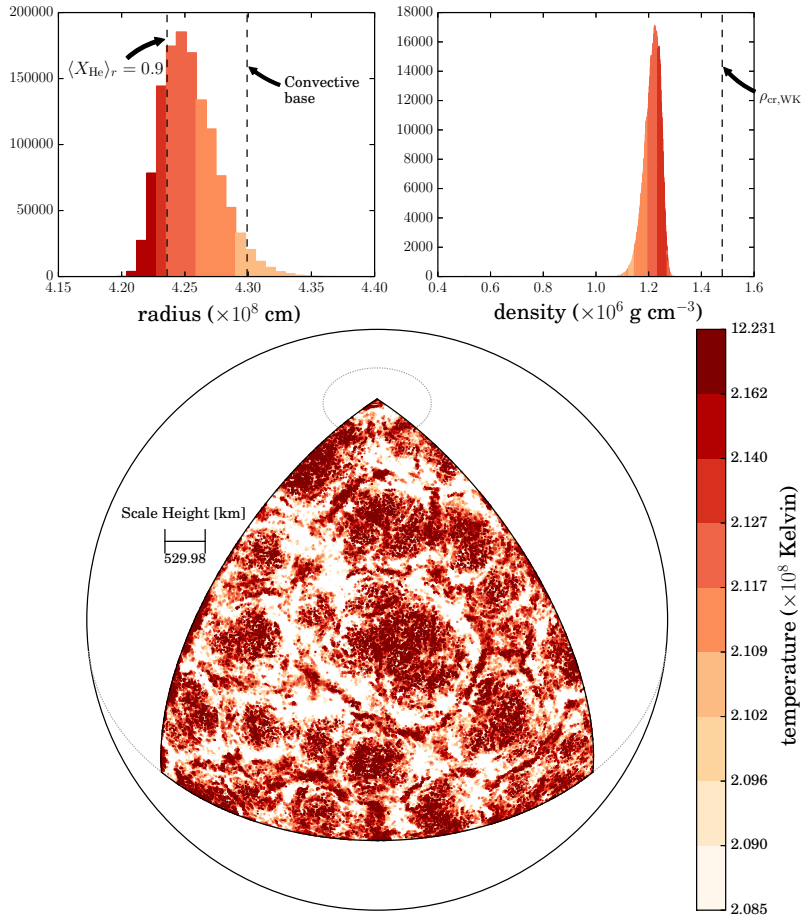


Figure 3.5 Hotspot properties for model 08130 at $t = 130.0$s. **Upper left:** histogram of the radii of the hotspots, with bin sizes chosen to match the finest level of 3D resolution. Each bin is color-coded to indicate the *maximum* temperature in the bin. The location where helium's laterally averaged mass fraction is 0.9 and the base of the convective envelope are indicated. **Upper right:** histogram of the densities of the hotspots. Again, bins are color-coded based on *maximum* temperature. The value of Eqn. 3.1 is indicated. **Bottom:** projection of the hotspots' angular location onto a sphere corresponding to the average radius of the hotspots. Hotspot pixel sizes correspond to the finest level's physical resolution, and where hotspots overlap preference is given to the one with the highest temperature. The extents of the temperature color bar's bins are set such that each bin contains an equal number of hotspots.

Table 3.2. Ignition Conditions

model	$t_{\rm peak}$ [s]	$t_{\rm peak}/\langle\tau_{\rm conv}\rangle$	$\langle r_{\rm base}\rangle$ [km]	$\langle T_{\rm base}\rangle$ [$\times 10^8$ K]	$\langle\rho_{\rm base}\rangle$ [$\times 10^5$ g cm^{-3}]
12030H	118.2	8.4	3096.4	2.180	13.877
12030	120.4	6.4	3009.4	2.178	14.541
12020H	420.0	10.3	3261.6	2.480	9.424
12020	410.0	12.4	3163.7	2.410	9.963
11030H	319.5	10.6	3839.8	2.462	7.729
11030	205.0	8.5	3739.1	2.456	7.968
08130H	161.7	3.7	4372.9	2.139	11.464
08130F	137.4	1.2	4251.7	2.096	12.218
08130	130.6	3.3	4259.6	2.069	12.106
08120H	240.0	7.7	4470.0	2.181	10.365
08120	710.0	25.7	4338.7	2.041	11.221

often an excellent predictor of ignition. Ignition is achieved almost exactly as the average density at the peak burning region surpasses the (temperature-dependent) critical density in Fig. 3.1, whereas the same average density is well below the critical value in the quasi-equilibrium case of Fig. 3.2.

This predictor does not work as well for our models with 0.8 $M_\odot$ core masses. These models achieve localized runaway despite being quite a bit below the critical density. This could either be an effect of our models capturing the 3D dynamics and thus of scientific interest, or it could be a result of our models being toward the upper limit of the minimum shell mass the critical density is calculated for in our 0.8 $M_\odot$ models. In addition, the critical density is only a rough estimate based on the outcomes of several 1D models. Still, for cores $\geq$ 1.0 $M_\odot$ it is quite accurate for our models. Further investigation of this critical density and determining a 3D version of it are future directions for this work.

Table 3.2 lays out key properties at the time of runaway for all models that experienced localized runaway. The table includes an estimate of the number of convective turnover times modeled before ignition as well as the lateral averages of temperature and density at the radius of peak burning ($r_{\rm peak}$). Note that these are values based on the 3D output with a timestamp nearest that of the peak temperature. This is why the peak times tend to end on round numbers—our 3D state is output at most every tenth of a second.

3.4 Convection

The ignition characterized in the previous section is fostered by the complex interplay between nuclear burning and convective dynamics. A detailed understanding of convective evolution is crucial, as illustrated by the conflicting results of Fink et al. (2010) and Woosley and Kasen (2011).

Fink et al. (2010) get their initial 1D models from Bildsten et al. (2007), who assume a fully convective shell all the way up to the point of ignition. Woosley and Kasen (2011) employ a time-dependent convective model based on mixing-length theory, allowing for convection to "freeze out." If the runaway timescale for a volume of fluid at the base of a convective zone, in the absence of cooling, becomes smaller than the convective turnover timescale, convection is no longer able to cool the helium-burning layer with the same efficiency. It starts to "freeze out." A fully convective shell requires a larger temperature at its base to achieve runaway than with one in which freeze-out is allowed. The temperature differences translate into entropy differences. In turn, the lower entropy of the cooler base makes possible larger base densities in the Woosley and Kasen (2011) models than those of Fink et al. (2010), even when they are modeling similar core and shell masses. These density discrepancies lead to significant discrepancies in explosive burning and the products yielded.

Our analysis of convective dynamics begins by establishing the broad context. Fig. 3.6 plots volume renderings of radial velocity for several models. These act as a proxy for convective plumes. Higher core masses result in more compact systems with smaller pressure scale heights. Thus we see the typical size of a convective cell grows inversely proportionally to core mass. The hotspots plotted in Fig. 3.5 occur at the base of convective plumes of cool in-falling fluid and at regions in which outflows of adjacent convective cells collide. The in-falling matter increases the density, setting off more vigorous nuclear burning. A competition between the nuclear burning rate of a volume of fluid and the rate at which it can be transported and cooled by convective flow is established.

In our models we see convective dynamics have two key impacts. First, convective overshoot serves to push the region of active burning deeper into the star, thus increasing the ambient density of nuclear burning sites and altering the ambient composition. Second, convection's ability to respond to increasing nuclear energy generation breaks down for sufficiently energetic burning. This is the freeze-out discussed in Woosley and Kasen (2011).

Convective overshoot is demonstrated in the top plot of Fig. 3.7 for model 11030. The first process to happen in a model is for convection to be established in response to nuclear burning, the bulk temperature profile, and our initial velocity perturbations (see Fig. 2.1). This is seen in the plot of the

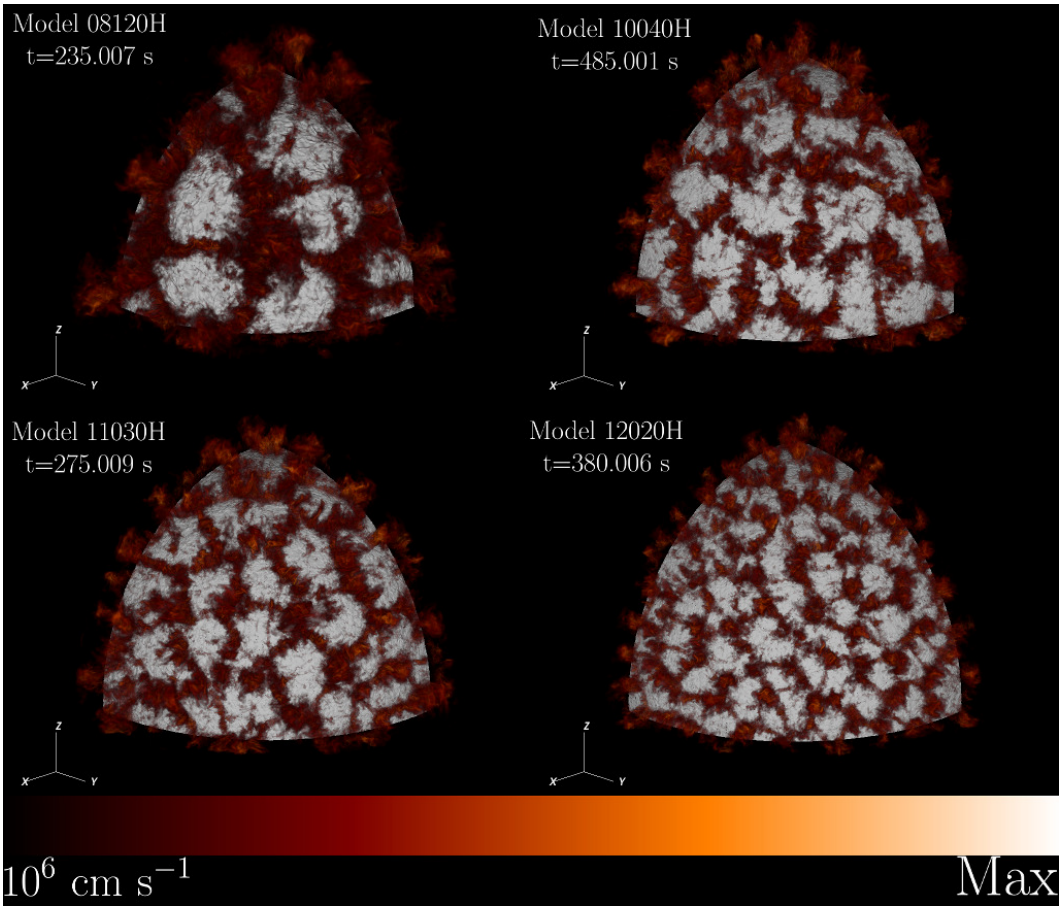


Figure 3.6 Volume rendering of radial velocities for some representative models. Maximum velocities are 2.8, 1.3, 2.0, and 1.7 $\times 10^7$ cm s^{-1} for models 08120H, 10040H, 11030H, and 12020H, respectively.

convective timescale at the bottom of Fig. 3.7. Initially it experiences a steep drop off until stabilizing as convective cells are established. After this we see that the location of the convective base steadily moves radially inward. In response, the radius of peak burning moves deeper into the star resulting in increased ambient density as well as changes in ambient composition. We find in most models that experience localized runaway, peak burning radii tend to stabilize near the location where composition is 90% helium (shell material), 10% carbon (core material). In addition, models in quasi-equilibrium do not have such substantial radial migration of the convective base.

Freeze-out is demonstrated in the bottom of Fig. 3.7. Here we plot two different timescales. To calculate a minimum nuclear burning timescale we invert the burning rate, dX_i/dt, for the most rapidly burning species i (for these models, helium). This is done for all radius bins r in our lateral averaging. The minimum value of this radial slice is used as our nuclear timescale $\tau_{\rm nuc}$. In sum,

$$\tau_{\rm nuc} = \min_r \left(\min_i \left\langle \frac{dX_i}{dt} \right\rangle^{-1} \right)_r . \tag{3.2}$$

The other timescale is a conservative estimate of the convective turnover time. Again using laterally averaged data, we invert the velocity magnitude $\langle |\mathbf{U}| \rangle$ and integrate over the convective region,

$$\tau_{\rm conv} = \int_{r_b}^{r_t} \langle |\mathbf{U}(r)| \rangle^{-1} \, dr. \tag{3.3}$$

r_b is the smallest radius at which the radial entropy profile satisfies $ds/dr = 0$ and r_t is largest radius satisfying this condition. The region is shaded in Fig. 2.1. This average value is reported in Table 3.1. In addition, Table 3.1 includes a calculation of the minimum and maximum estimates for turnover times by simply dividing the lengthscale $r_t - r_b$ by the maximum and minimum velocities in the convecting region, respectively. We want to stress these are conservative estimates. Not all plumes that form will extend the full length-scale, and there may be plumes with faster fluid than in other plumes.

In the timescale plot of Fig. 3.7 we see that as model 11030 approaches runaway the nuclear burning timescale drops more rapidly than the convective turnover timescale. We note that the focus is not on the magnitudes of the timescales but their changes. The highly nonlinear nature of nuclear burning make comparisons of the rate's magnitude at any given time with other timescales uninformative. The changes however suggest the nuclear burning rate is shrinking faster than the convective turnover rate as runaway is approached, suggesting a degree of freeze-out.

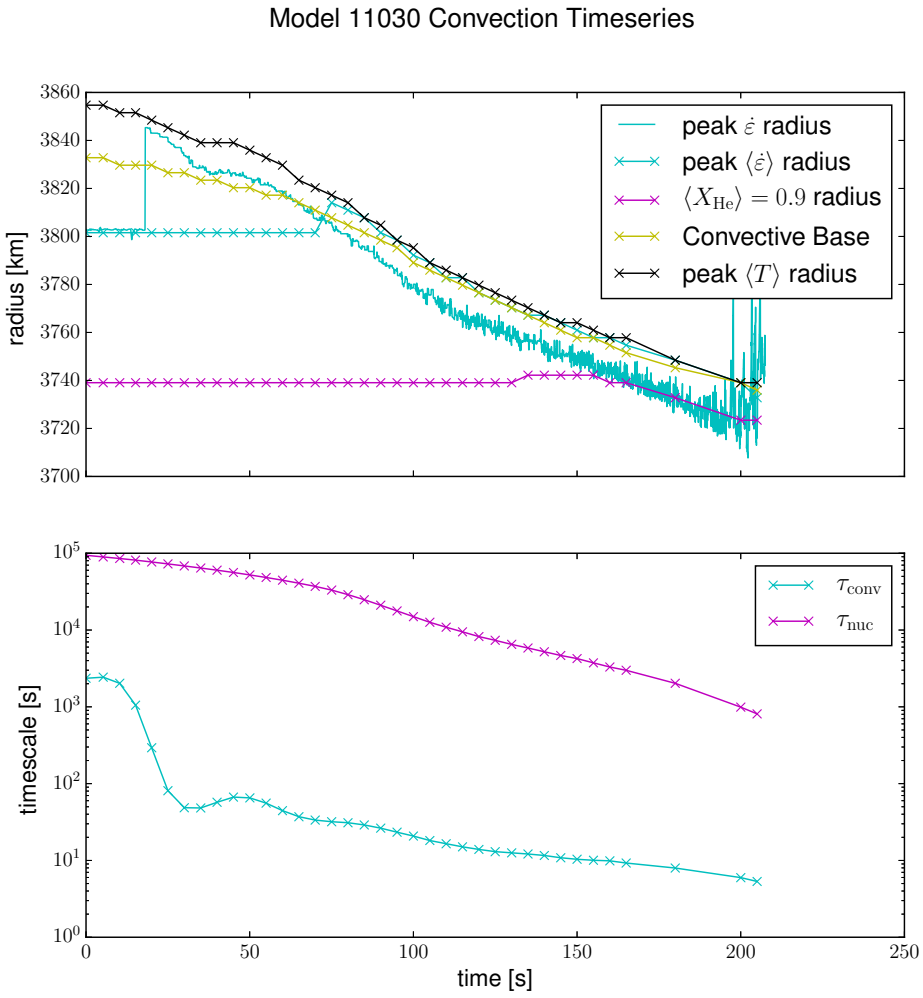
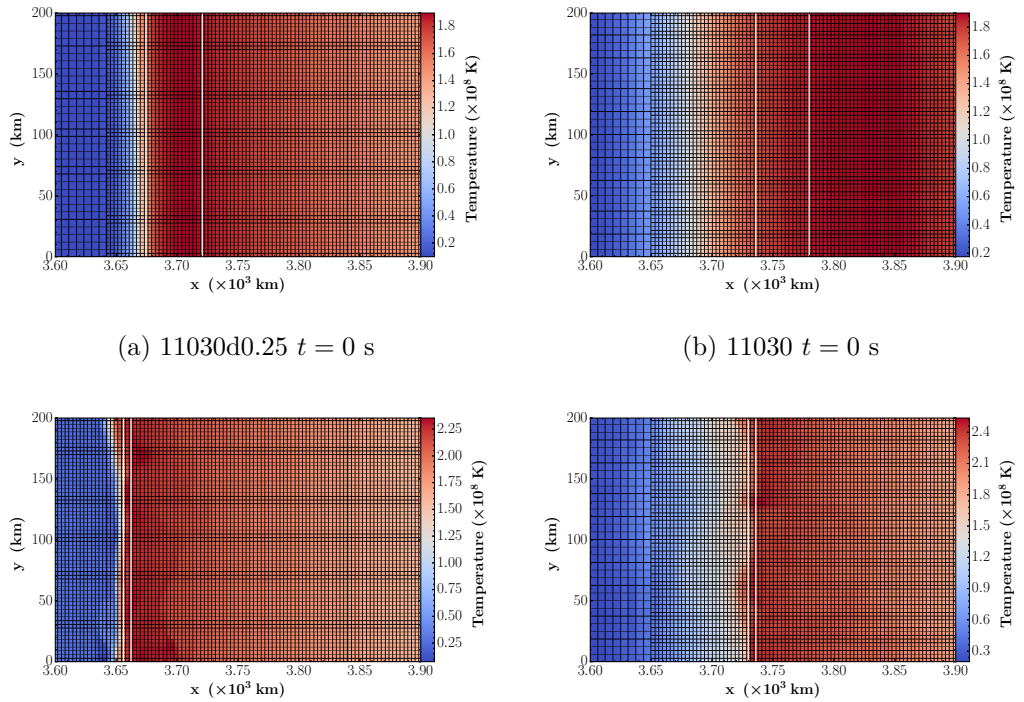


Figure 3.7 **Top:** Here we plot: the radius of the cell with the largest nuclear energy generation in the entire domain, $r(\dot{\epsilon}_{\rm peak})$, in cyan; the radius of this same quantity for the laterally averaged data, also in cyan with "x" markers indicating the timestamp the data was calculated for; the radius at which the lateral average of helium's mass fraction $\langle X_{\rm He}\rangle$ is 0.9, in magenta; the radius of the convective base as determined by the radius at which the entropy flattens out (see Fig. 2.1), in yellow; and the radius at which the lateral average of temperature $\langle T\rangle$ peaks, in black. **Bottom:** A comparison of a nuclear burning timescale (magenta) with a convective turnover timescale (cyan). See text for details.



(a) 11030d0.25 $t = 0$ s

(b) 11030 $t = 0$ s

(c) 11030d0.25 $t = 375$ s

(d) 11030 $t = 180$ s

Figure 3.8 Zoomed-in X-Y slices of temperature at the base of the convecting helium envelope for models 11030 and 11030d0.25 at initial and late times (see subplot labels). Cell edges are overlayed. The two white vertical lines mark, left-to-right, the radius at which the lateral average of the velocity magnitude is 25% and 50% of its peak.

In contrast, quasi-equilibrium models and the convective runaway model have relatively flat peak burning radii. Their convection is able to respond to nuclear burning before any localized runaway can develop.

3.5 Varying δ and Comparison with 1D

As discussed in §2.4, we have carried out an additional numerical experiment exploring the impact of varying the δ parameter that determines the sharpness of the transition from core to shell (see Paper I for details). In Fig. 3.9 we plot the temperature profiles of models 11030d0.25, 11030d0.5, 11030, and 11030d2, which have identical initial models except for their δ parameters: 12.5, 25, 50, and 100 km, respectively. In addition, we plot the temperature

profile of model 10HC from Woosley and Kasen (2011) at a time near ignition (model data courtesy of Woosley, private communication). The radius of 10HC is offset by 500 km to facilitate comparison of transition widths. We plot both the initial temperature profile and those from later times. Initially, our default δ value results in a more gradual transition than in 10HC. At later times, the temperature profile develops a more pronounced peak like that of 10HC, though the transition continues to be less sharp than 10HC and leaves a substantial amount of hot fluid below the temperature peak. We see that our smaller δ values, in particular that of 11030d0.25, are about as sharp as that of 10HC.

One potential worry with the thicker transition is that it allows a thin shell of hot fluid (80-90% of the peak interface temperature) to exist below the region initially unstable to convection (see Fig. 2.1). If convection is not well-established in these hot shells as the model approaches runaway then it becomes hard to determine if the runaway is a result of the uncooled hot shell in some initial models or the convective dynamics. To address this concern, we plot slices of temperature at $t = 0$ and late times for 11030d0.25 and 11030 in Fig. 3.8. This plot includes white vertical lines marking the radius at which the laterally averaged velocity magnitude is 25% and 50% of its peak value. This gives the reader an idea of how much convective cooling penetrates. The late-time temperature profile develops into a thin, hot layer for both δ values, with similar velocity penetration. This demonstrates that even with a thicker transition the cooling in our models penetrates to the thin shell of vigorous burning. The primary impact of the smaller δ is to shift the radius at which the thin, hot layer develops. This is an expected imprint of the initial model, as the thinner δ locates a more concentrated shell of hot fluid at a lower radius, as seen in Fig. 3.8. In addition, the thicker δ results in a larger region of intermediate temperature below the thin, hot layer. Finally, we note that a history of surface runaway events can heat the base of the convecting envelope.

We find that all of the runs with varied δ's also experience localized runaway. In addition, the convective dynamics reported in previous sections are qualitatively insensitive to varied δ. Figs. 3.10 and 3.11 illustrate this using model 11030d0.25. We also see a new result: the dramatic rise of the typical radius at which helium's mass fraction satisfies $X_{\rm He} = 0.9$. This suggests substantial mixing as carbon is dredged up, displacing helium. Future work characterizing localized runaway will explore this mixing in more detail.

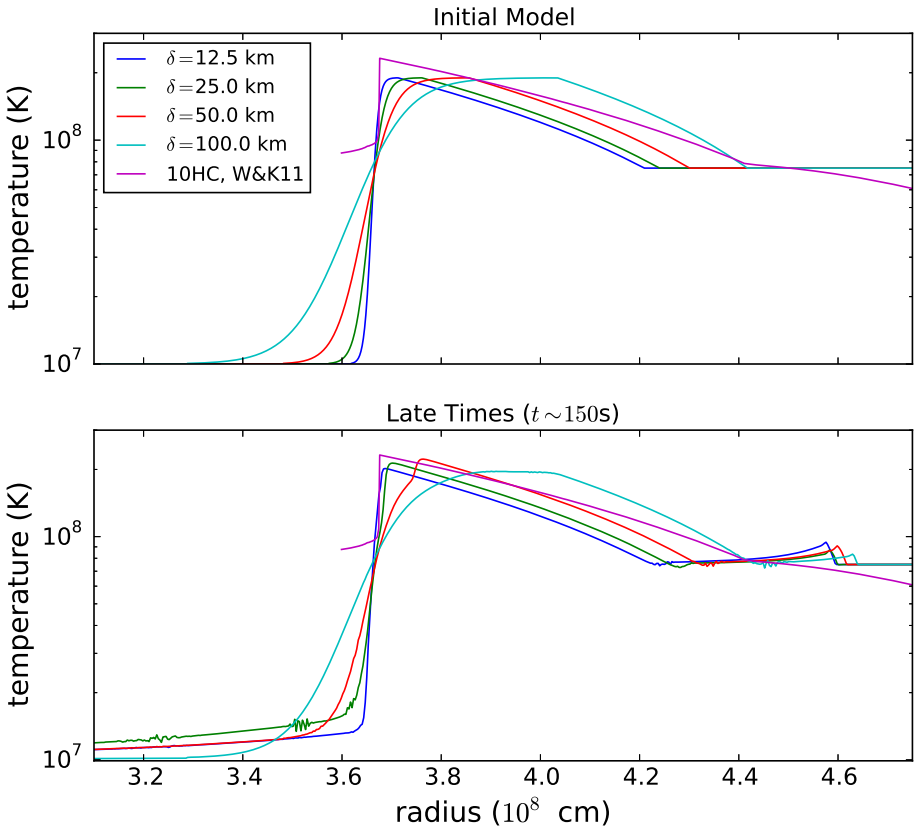


Figure 3.9 Comparison of temperature profiles for varying δ. **Top:** The temperature profiles of initial models with varying δ along with a realistic 1D reference (model 10HC from Woosley and Kasen (2011), radius shifted -500 km to facilitate comparison). **Bottom:** The same reference model along with profiles for the laterally averaged temperature data from models 11030d0.25, 11030d0.5, 11030, 11030d2 after $\sim$ 150s of evolution.

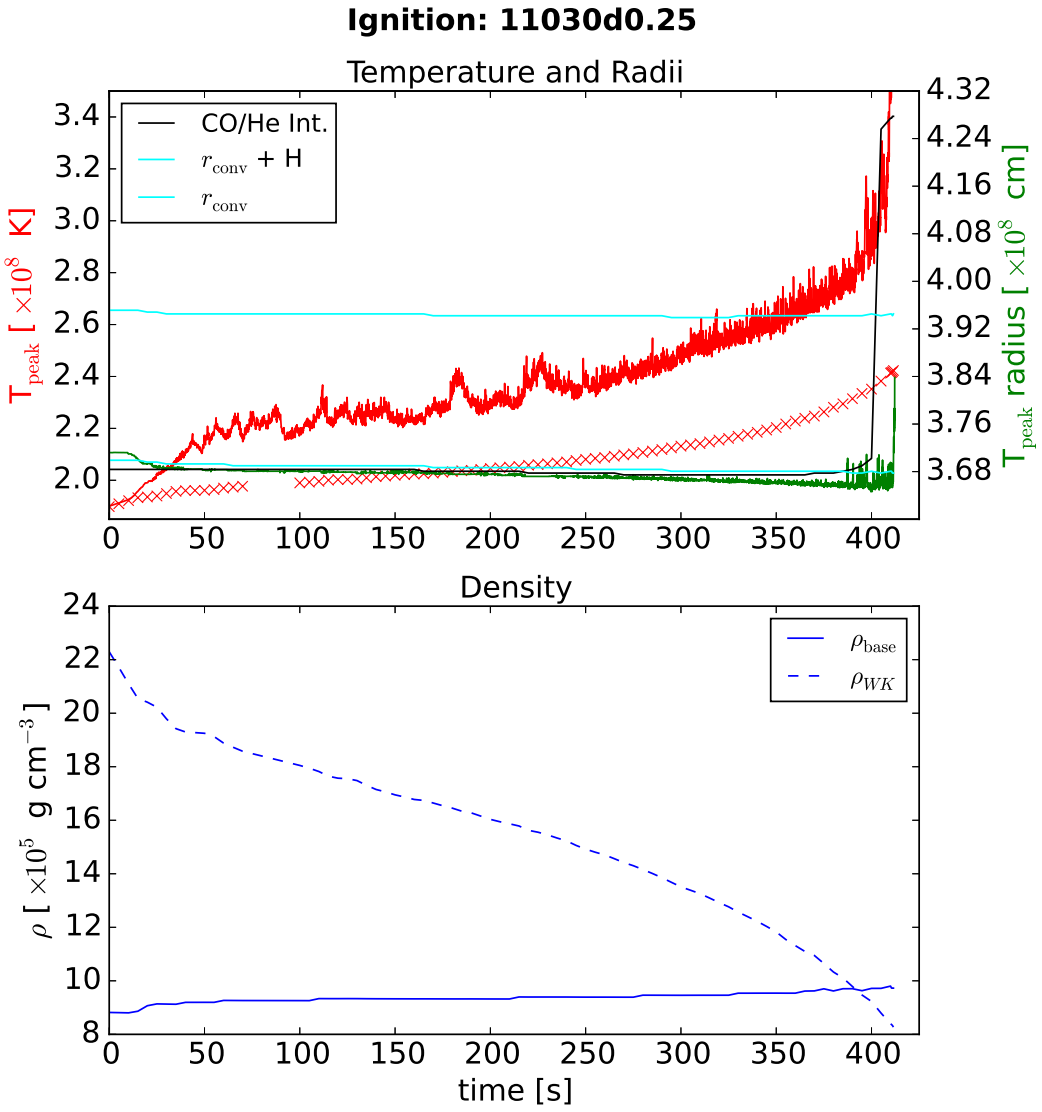


Figure 3.10 Same as Fig. 3.1 for model 11030d0.25

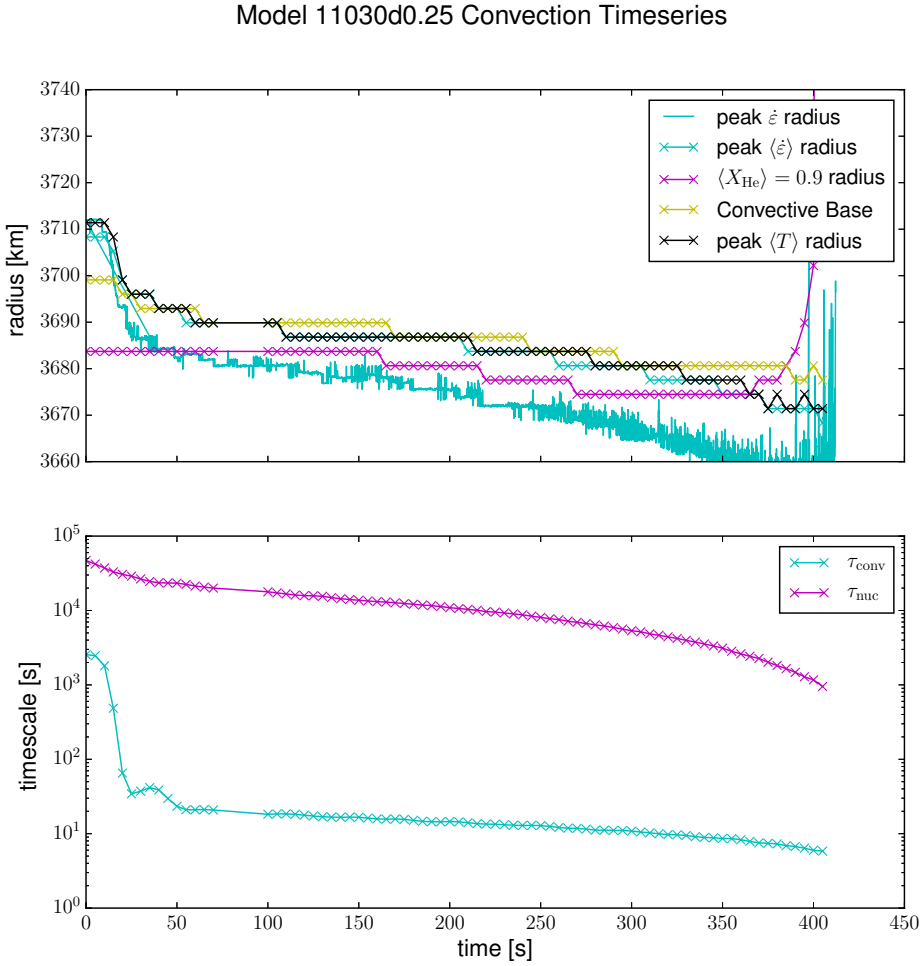


Figure 3.11 Same as Fig.3.3 for model 11030d0.25

Chapter 4

Localized Runaway in Simple Models

One of the most crucial open questions regarding the various explosive possibilities of runaway in helium envelopes is if and how it develops when spherical symmetry is not assumed. As detailed in Chs. 1 and 3, only spherically symmetric (1D) models have demonstrated the development of a detonation in the shell. Multi-dimensional models have focused on the question of what happens *if* a detonation forms in the helium shell. In Ch.3 we demonstrated what we call localized runaway based on bulk trends. In this chapter, we develop a methodology for a focused analysis of runaway events. This is the first step toward answering the question of if a detonation is likely to develop, how the seeds of ignition are characterized, and if other possible outcomes exist such as shell deflagrations.

For the purposes of developing this methodology we focus on model 08130F (see Tab. 2.1). This is not a model likely to yield a SNe Ia, and the shell is quite massive, perhaps larger than is expected in nature. However, 0.8 $M_\odot$ core models have a larger spatial extent, which in turn makes it possible to resolve the shell without as much refinement as required by larger cores. This makes it computationally expedient to execute a model of the full star (instead of just an octant) and to develop our methodology. In the future, this methodology will be applied to more computationally expensive models.

4.1 Characterizing localized runaway

To begin, we consider a single localized volume of fluid experiencing runaway. We start with the location of the hottest cell in the domain, which is tracked as a diagnostic at every timestep of the model. We choose the time at which

Table 4.1. Localized Runaway Properties

model, event	$\langle\rho\rangle$ [$\times 10^6$ g cm^{-3}]	$\rho_{\rm peak}$ [$\times 10^6$ g cm^{-3}]	$\langle T\rangle$ [$\times 10^9$ K]	$T_{\rm peak}$ [$\times 10^9$ K]	scale[a] [km]	$T_{\rm thresh}$ [$\times 10^9$ K]	$N_{\rm cells}$
08130F, 1	0.41	0.75	1.15	1.49	123.2	0.7	3777
08130F, 2	0.65	0.77	0.96	1.08	16.5	0.7	9

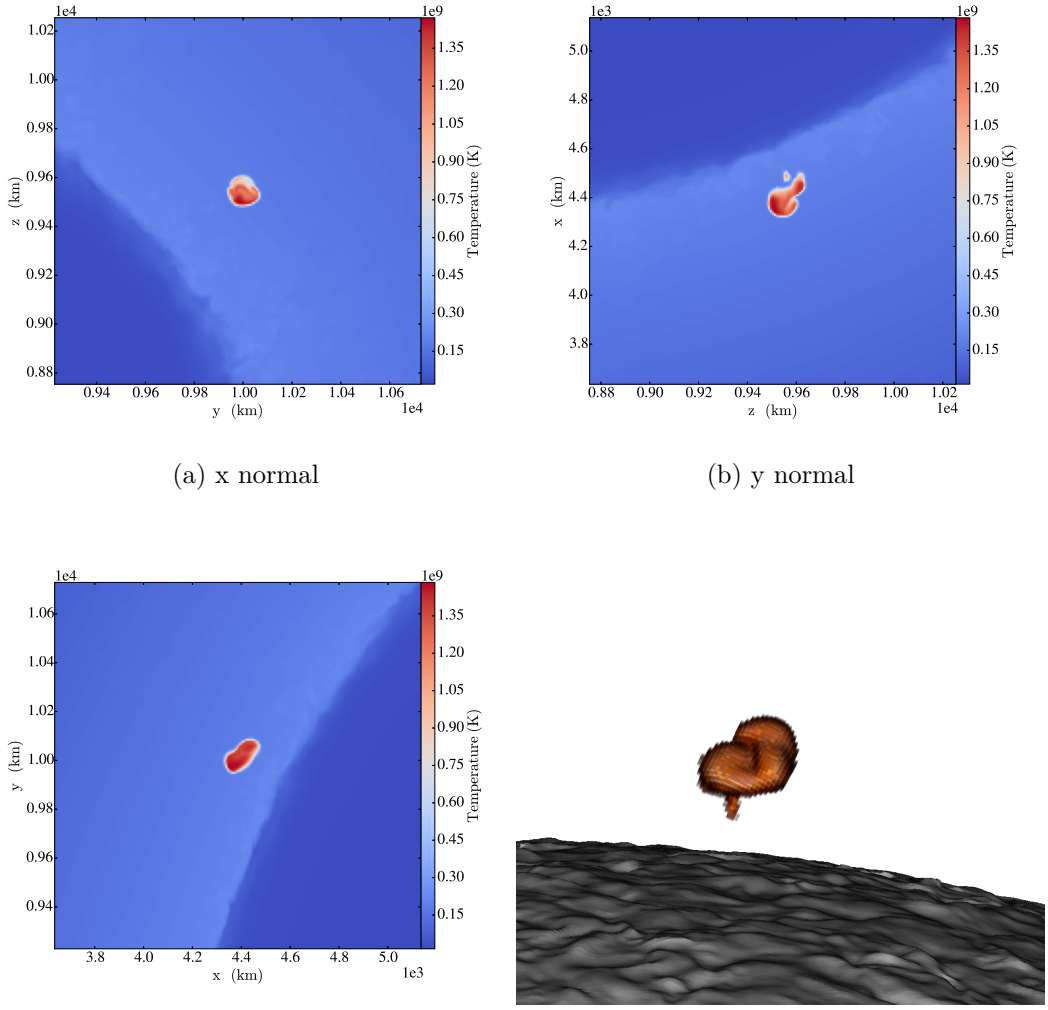
[a]This is the cube root of the volume.

the peak Mach number first reaches 0.3, roughly the limit at which we expect the low-Mach approximation to hold. The domain at this time is analyzed to generate a list of the hottest cells from which we can determine the collection of cells running away in the vicinity of our peak temperature diagnostic. Fig. 4.1 provides an initial impression with three orthogonal slices and a volume rendering. The morphology is plume-like, similar to a mushroom cloud's shape. The averaged radius of the cells is 116.5 km above the average radius of the base of the convecting shell. This is about 22% of the scale height for this model, $H = 530$ km.

From the slice data, we can determine a temperature bounding the runaway volume. All cells at or below this threshold temperature are analyzed to determine the volume's quantitative properties. In Tab. 4.1 we detail the properties for the listed threshold.

We have characterized the volume running away in the vicinity of the peak temperature of the domain. However, a key open question is if runaway events are isolated or if we expect multiple events. Once one event is initiated, there is a limited window during which additional events can form, if we assume detonation. An estimate of this window is the sound-crossing time required to traverse half the circumference of surface. For 08130F, this is about 5.7 s (using $r = 4252$ km, $c_s = 2335$ km/s, the conditions at the radius of peak laterally averaged temperature). In fact, using the data from the same timestep we find another runaway developing, well within this window. This volume is much smaller, consisting of only 9 cells. Still, the temperature is well on its way to runaway and it is well-separated from the larger runaway volume. We also note the higher density. This is related to the fact that the second event has an average radius 43.5 km *below* the average radius of the convecting base[1]. This suggests, as may be expected, that runaway events develop close to the convective base and are transported by the convective velocity field toward

[1]Note that the average base is based on a lateral average of the 3D state. Individual convective plumes will have bases above and below this average



(a) x normal

(b) y normal

(c) z normal

(d) volume rendering

Figure 4.1 Three orthogonal slices and a volume rendering of a localized runaway event in 08130F at t=138.55 s.

the outer envelope. An interesting question for future work will be to take the initial conditions we find here at a peak Mach number of 0.3 and track their evolution in a compressible hydrodynamics code that can capture high Mach dynamics.

4.2 Comparison with critical conditions

The focus of this chapter is developing a methodology for understanding runaway in our models. As mentioned, 08130F is chosen because it is a computationally expedient target but is not the most interesting scientific target. Still, we can compare the conditions characterized in the previous section with critical conditions determined in the literature, allowing speculation on whether an event is likely to evolve into a detonation.

The spherically symmetric study of Woosley and Kasen (2011) includes 0.8 $M_\odot$ core models labeled 8HBC and 8HC with helium shells of 0.097 and 0.139 $M_\odot$, respectively. Our 08130F has a shell mass between these two, but close to 8HC at 0.13 $M_\odot$. The lighter shell experiences detonation and triggers a secondary detonation in the core via compression waves. The heavier shell experiences a detonation in the shell that propagates directly into the core at the core/shell interface. From this, we may expect the runaway events we have examined above will yield a detonation. Of course, spherically symmetric models cannot fully capture the inherently 3D evolution of the convecting shell and all detonations will manifest as a spherical shell detonating simultaneously. To see what our results might say about the impact of such 3D effects, we consider critical conditions estimated in Shen and Moore (2014).

Fig. 8 of Shen and Moore (2014) provides conservative estimates of the minimum size of hotspots (what we are calling localized runaway volumes in this work) in the helium shell that will lead to detonation via the Zel'dovich gradient mechanism Zel'Dovich et al. (1970). We caution that the hotspots in Shen and Moore (2014) are simplified in that they have a central peak temperature that linearly drops to the ambient temperature. Those in Fig. 8 assume constant density throughout the hotspot. If we let the peak temperature of event 1 in Tab. 4.1 represent the central temperature and the average density the constant density, then we have $T_{\rm center} = 1.49 \times 10^9$ K, $\rho = 4.1 \times 10^5$ g cm^{-3}. The best line to compare this with is the $\rho = 3 \times 10^5$ g cm^{-3} line of Fig. 8. For a pure helium shell with a simple reaction network, the minimum detonation size is about 1000 km, much larger than the 123 km scale of event 1. However, the authors also carry out models with some traces of C-N-O burning mixed in with the helium and a large 206 isotope network. In this case, the minimum detonatable size is greatly reduced to about 10 km. In this case,

we expect detonation to develop, in agreement with the 1D findings discussed previously. A more detailed analysis is warranted, but already we have very suggestive evidence that event 1 in our model would proceed as a detonation were we to track its evolution beyond Mach numbers of 0.3.

Chapter 5

Accelerating Nuclear Burning Algorithms

For the simulations reported in prior chapters, nuclear reaction integration typically takes 10-20% of each timestep. Given that we deploy a very simple three isotope network, this should convey how computationally intense this calculation is. For the purposes of this dissertation, a simple network is sufficient. We are focused on the bulk dynamics for which we need only the reactions that dominate energy release. But rich science is possible if we can model larger reaction networks. Larger networks will more fully capture energetics and, more importantly, will enable us to more precisely calculate the initial composition of the envelope immediately prior to ignition. This composition has a significant impact on the ultimate outcome of a detonation Piro (2015) and enables higher fidelity calculations of the nucleosynthesis in these systems. This is one of the first steps to connecting computational models to observables.

To make this possible, we have developed accelerated versions of our nuclear burning algorithms. We target Oak Ridge National Lab's Titan supercomputer, which contains NVIDIA K20X Kepler GPU accelerators. However, our code is run on a variety of systems and supercomputer architecture is constantly evolving — especially in recent years. Thus, we want to develop a single codebase that can be run on machines with no or different accelerators and we want to avoid code that is vendor- or architecture-specific. The only way to do this currently is to make use of the OpenACC standard. This standard, like the much more mature OpenMP, consists of directives[1] and is implemented by leading compiler developers for supercomputers. Most

[1]These are simply comments in code that instruct compilers to generate code for hardware without having to write hardware-specific code.

notably, PGI compilers have seen enormous effort directed at implementing the OpenACC standard, which if anything has been accelerated by NVIDIA's K20X acquisition of PGI. This standard enables us to get the benefits of GPU acceleration without having to write code specific to NVIDIA hardware (such as CUDA).

In the sections to follow, we give an overview of GPUs and the OpenACC standard, present both a simplified accelerated prototype and an accelerated production integrator, discuss the lessons learned during development, and analyze the performance of the accelerated algorithm.

5.1 GPUs and OpenACC

In 2008, the first hybrid supercomputer was commissioned in the form of Los Alamos National Lab's Roadrunner[2]. This novel machine, the first to demonstrate a peak theoretical performance over one petaflop[3], consisted of nodes that contained not only traditional processors (in the form of dual-core AMD Opteron 2210's) but also *co-processors* in the form of IBM PowerXCell 8i's[4]. In a hybrid design, the traditional processor carries out computation and local system management while the co-processor is designed to specialize in rapid computation. Typically, code must be modified to make productive use of the co-processor. Often code that runs on the co-processor cannot be as complex as that written for the processor. In exchange, specific types of calculations can be carried out much faster than on the processor.

This will all be made much more concrete by describing the particular system our work targets: Oak Ridge National Lab's Titan supercomputer. Titan was commissioned and tested in 2012 and became available to researchers in 2013. It was ranked as the fastest supercomputer in the world in November 2012. As of June 2016 it is ranked #3[5]. Titan consists of 18 688 nodes. In each node is a 16-core AMD Opteron 6274 and an NVIDIA Tesla K20X (Kepler) GPU. This yields a peak theoretical performance of 27.1 petaflops. It is important to stress that about 90% of this peak performance is realized in the GPUs! If code does not utilize these, it is only exploiting a fraction of the available computational power.

Fig. 5.1 is a schematic of the K20X. It is composed of 14 streaming multiprocessors (SMXs). Communication between the CPU (host) and GPU (device) is over a PCIe interface. The device includes 6 GB of global memory

[2] http://www-03.ibm.com/press/us/en/pressrelease/24405.wss

[3] A "flop" is a floating point operation, such as adding two decimal numbers.

[4] The same hardware SONY's PlayStation 3 used to render video game graphics

[5] https://www.top500.org

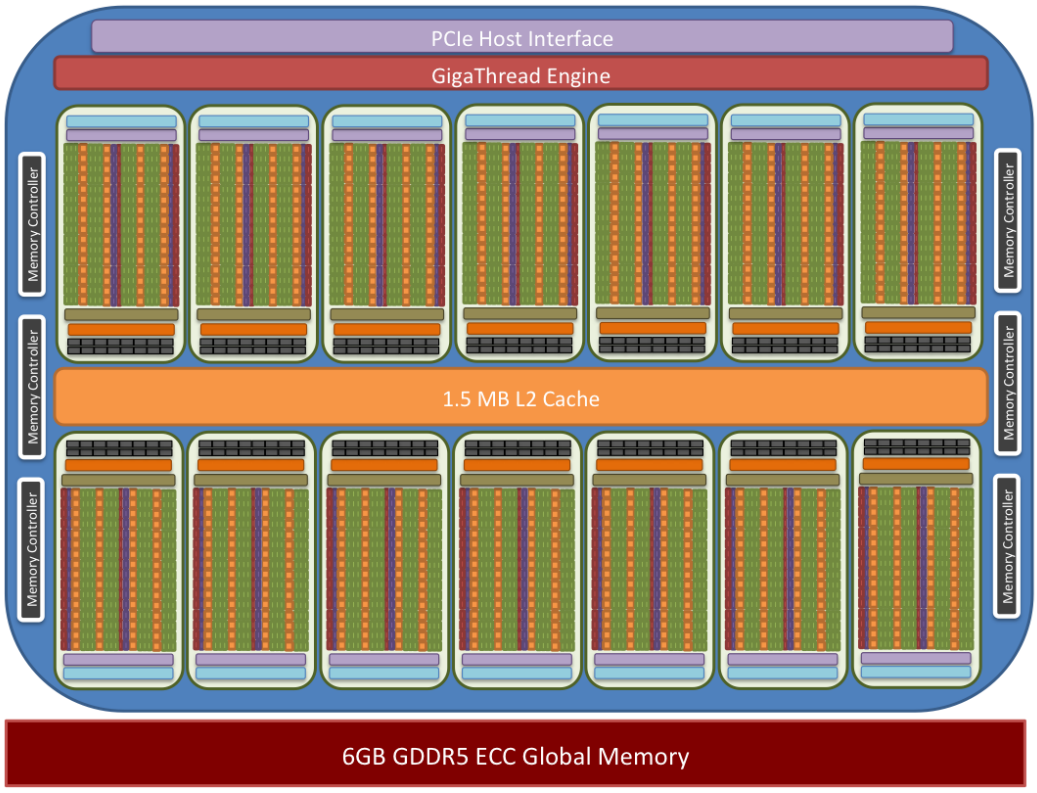


Figure 5.1 Schematic of the NVIDIA K20X GPU. See text for details. Image credit: Oak Ridge National Lab, *Accelerated Computing Guide*

accessible by all SMXs and 1.5 MB of shared L2 cache. The GigaThread Engine is part of the proprietary hardware that makes GPUs capable of so much parallel computation. This engine is capable of switching data contexts with no (negligible) overhead. Roughly, the engine manages multiple threads of computation for each CUDA core (more on these soon). Eventually, a thread will need to access global memory, which is relatively slow. When this occurs, the engine rapidly switches in a thread ready to compute and its data context so that the core spends as little time waiting and as much time computing as possible. This is called *latency hiding*.

Fig. 5.2 details an individual SMX. Each of these contains 192 CUDA cores capable of carrying out single-precision (32 bit) operations and integer arithmetic, 64 double-precision (64 bit) cores, and 32 special function units for transcendental function calculations. There is also a small amount of local memory. Of particular note are the registers. These hold the fastest memory on the chip accessible by threads. The threads running on an SMX must share the 65 536 32-bit registers. This is where memory private to a thread resides.

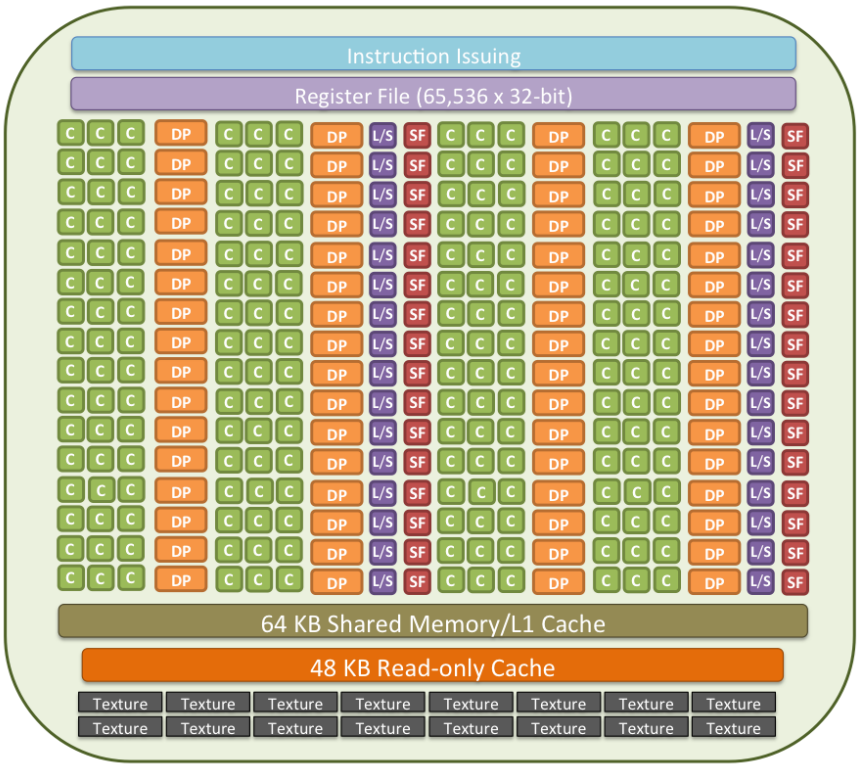


Figure 5.2 Schematic of an individual K20X SMX. See text for details. Image credit: Oak Ridge National Lab, *Accelerated Computing Guide*

They are a scare resource and are often a bottleneck in developing performant GPU code. If threads require many registers, this limits the total number of threads that can be deployed, which in turn largely nullifies the benefits of latency hiding.

Beyond latency hiding, the incredible performance of GPUs is made possible by their massively parallel nature. The K20X can execute 2688 threads of single-precision or integer calculation simultaneously, or 896 simultaneous threads of double-precision calculation (which is most of what we will be doing in a scientific code). Compare this to the 16 threads Titan's CPU can handle simultaneously. So even if you cannot fully exploit latency hiding, you may still be able to achieve better performance than on a threaded CPU.

Sadly, we have not discovered the mythical "free lunch." As can be seen in the schematic, each SMX has clusters of cores arranged in four columns. In GPU parlance, these are called *warps*. Each warp will execute threads assigned

to it in a single instruction, multiple data (SIMD) fashion[6]. This is similar to vector instructions that some CPUs are capable of. The same instruction will be applied in lock-step to a vector of data. So unlike threads on CPUs that can each execute an independent set of instructions, clusters of threads on GPUs execute the same instruction. In scientific computing, we often have loops or nests of loops in which each iteration does not depend on data from the previous iteration. This data independence means such loops can be decomposed into parallel threads in which the same instruction is applied to a vector of data generated from independent iterations. For example, consider the following simple loop:

```
do i = 1, 64
   a(i) = b(i) + c(i)
end do
```

The iteration $i + 1$ does not depend on iteration i. So on the GPU, we could load the data from the arrays a, b, c into 4 warps of 16 threads each and execute this entire loop as concurrent threads executing on a single SMX. So while we can achieve impressive parallelism it must be in this vector form in which subsets of threads execute the same instructions.

Another major limitation is the PCIe connection mentioned previously. GPUs are not self-hosted, they are managed by the CPU. They do not have access to the system's main memory. Code runs on the CPU until GPU-specific instructions are reached. Data must be moved from the CPU to the GPU's memory, and any needed output must be moved back to the CPU over the PCIe bus. To make effective use of the GPU, the amount of such data transfer must be limited. Computation time must dominate the data transfer time. To offset some of this cost, GPUs will often reserve a section of the CPU's memory. This is called pinned memory. Each node in Titan will have a pool of memory available to the CPU (in particular, 32 GB of RAM). This memory is managed by the operating system which means it will be paged to the hard disk so that the memory address space can be larger than the available on-chip RAM. A region of this memory will be reserved (pinned) for use by the GPU. This pinned memory will *not* be paged. This allows the GPU to safely directly access the memory, improving performance of data management on average. Our strategy for addressing memory issues and data transfer will be elucidated in sections to follow.

We have laid out the fundamentals of a GPU. But how do we program such a device? There are a few options. Perhaps the most popular low-level language is specific to and developed by NVIDIA: CUDA. CUDA code can be written in the major languages of scientific computing (C, C++, Fortran).

[6]Sometimes called SIMT for GPUs: single instruction, multiple threads

CUDA allows the programmer to specifically target a CUDA-enabled GPU (most prominently, NVIDIA GPUs). The programmer has low-level, fine control of the GPU. The costs of such power include a steep learning curve, the need to rewrite and perhaps redesign algorithms in CUDA, and code that is tightly coupled to a specific architecture.

Another low-level option is OpenCL. This is a C-like programming language designed for heterogeneous architectures. It is an open standard agnostic of hardware and thus alleviates the cost of being tied to a specific vendor like NVIDIA or particular architecture like a GPU. Other co-processors can be targeted. However, we still have the cost of a steep learning curve and the need to extensively rewrite and maintain code in a new language, adding significant costs to the development and maintenance.

Both of these options present challenges for a code like Maestro. Instead, we deploy the OpenACC standard[7] to target accelerators like GPUs. Like the widely-used and mature OpenMP standard, it is a directive-based programming model. Code is marked up with comments that only have meaning to OpenACC-enabled compilers, so the same code can be compiled for a GPU one day and a CPU the next.

OpenACC's directives can be divided into data and compute categories. Data directives express how data should be communicated between the host and device. Compute directives are designed to instrument loops. Independent iterations are spread across the GPU and executed in parallel. For the purposes of this dissertation, we will introduce key directives needed for our development.

First we consider data directives. The most basic data directive creates a *region* that encloses a section of code that is to be accelerated. The directive describes how data (such as scalar variables, arrays, and data structures) in this region should be communicated to and from the GPU, if at all. A simple region has the form

```
!$acc data <data clauses...>
a = b + c
q = q/a
d = b/c
!$acc end data
```

The variables a, b, c, d, and q will be managed across the CPU and GPU based on the data clauses used. Such clauses include copy(...), copyin(...), copyout(...), and create(...). All can be illustrated using the previous example:

[7]For a full specification of the OpenACC standard, please see http://www.openacc.org/.

```
!$acc data copy(q) copyin(b,c) &
!$acc      copyout(d) create(a)
a = b + c
q = q/a
d = b/c
!$acc end data
```

copy(...) allocates memory on the GPU, copies data from the CPU to the GPU upon entry of the region, and copies the data back from the GPU to the CPU on exit. copyin(...) only does the first part of allocating memory on the GPU and copying the specified data from the CPU to the GPU. On exit, the memory allocated on the GPU is freed, but is *not* copied back to the CPU. copyout(...), as one might expect, does the obverse. Finally, create(...) only ever allocates memory on the GPU with no copying. Due to the expense of moving data across the PCIe bus it is crucial to make use of these clauses appropriately. The minimum amount of data needed on the GPU should be copied in, the minimum possible copied out, and data that is only needed for computation on the GPU should be created. By default, all data in a region will be copy(...)'d, so it is crucial to express explicitly how data should be treated.

As of the 2.0 version of the OpenACC standard, data can be managed in an unstructured way and data in Fortran modules can be managed as well. Thus, the programmer is not restricted to only using data in the immediate scope of the code being marked up and more sophisticated data management can be designed.

For module data we have !$acc declare <data clauses...>. This can take similar clauses as the !$acc data construct. OpenACC treats module data as global, so it resides on the GPU for the duration of the program execution.

Data can be communicated in an unstructured way across possibly widely separated code using the !$acc enter data <data clauses...> and !$acc exit data <data clauses...>

Unstructured data regions and use of module data introduces the issue that data may be modified or initialized between the time that it is declared and the first OpenACC compute region is encountered. To address this, we have the !$acc update <device(...) | host(...)>. At any time, the programmer can copy updated data from the CPU (host) to the GPU (device) or from the device to the host.

The primary compute directive we utilize is the !$acc parallel <compute clauses, data clauses...> directive. This too forms regions and is designed to accelerate loops. It takes the form

```
!$acc parallel
```

```
!$acc loop gang vector
do i=1, N
   a(i) = b(i) + c(i)
end do
!$acc end parallel
```

We have introduced some new directives and clauses above. A parallel region can be initialized without a loop immediately following, but unless a !$acc loop construct is encountered the code will effecively be run in serial on the GPU. These constructs can be combined as !$acc parallel loop....

The gang and vector compute clauses tell the compiler how to parallelize the loop iterations. Loosely, gang parallelizes iterations over the SMX's while vector parallelizes over the cores within an SMX. By combining the two, we instruct the compiler to break the iterations up across both the SMX's and the cores within. We see here that OpenACC exposes the hierarchical nature of GPU parallelism, in which we have coarse (across SMX's) and fine (across CUDA cores) levels of parallelism.

These are the essential parts of the standard needed to discuss our acceleration of Maestro's nuclear burning calculations.

5.2 Accelerating a Maestro Reaction Calculation

Before diving into the accelerated code, we overview Maestro's strategy for calculating nuclear burning evolution. For an extensive elucidation of how nuclear reactions are implemented within the overall Maestro algorithm, see Almgren et al. (2008).

Maestro, like many astrophysical reactive hydrodynamics codes, calculates the evolution of nuclear burning via Strang operator splitting. Schematically, this looks like

$$\mathbf{R}_{\Delta t/2}\mathbf{A}_{\Delta t}\mathbf{R}_{\Delta t/2}, \tag{5.1}$$

where we have represented reacting the hydrodynamics state and advecting it as operators $\mathbf{R}$ and $\mathbf{A}$, respectively. Strang splitting isolates the hydrodynamics from the reactions with an independent reaction half-step, a full advective step, and a concluding reaction half-step. During a burn, the density does not evolve but we can evolve temperature. This allows the burning algorithm to focus on calculating reaction rates and solving the stiff system of ordinary differential equations describing the evolution of modeled species.

Within the Maestro source, the algorithm for reacting is found in MAESTRO/Source/react_state.f90. Broadly, the algorithm consists of a set of nested

loops. The outermost loop iterates over grids of state data (such as temperature, density, and composition) that have been assigned to the local node the code is running on. Each grid contains cells of state data along each spatial dimension being modeled. A triply-nested loop then iterates over each dimension such that each cell of state data is passed to a burner subroutine that takes care of the integration. When the model is configured, a reaction network is chosen. Networks are found in MAESTRO/Microphysics/networks. This code defines the species being evolved and anything else being evolved, such as temperature. Physical properties are defined including reaction rates. These properties are used to provide the integrator with a right-hand-side and sometimes an analytic Jacobian for the purposes of implicitly solving a system of stiff ODEs.

The pre-acceleration code deploys the widely-used VODE integrator Brown et al. (1989). The benefit of the VODE code is that it has been around for decades. It is proven to be robust and fast for solving stiff systems of ODEs. However, this is also what makes VODE a terrible code to accelerate with OpenACC. Such old code makes heavy use of programming practices prohibited on the GPU (more on restrictions soon). For example, goto statements and common blocks. Thinking of the hardware model, we can see that if concurrent threads hit goto statements that jump to divergent parts of the source then we have violated the lock-step "single instruction" part of the SIMD model, whereas common blocks are global memory which does not map readily to the GPU's heirarchy of global and private memory. Legacy code is also generally difficult to read, maintain, develop, and debug.

Thus we reached out to collaborators and were provided with a code that we call VBDF in this work. This integrator is based on the same numerical principles as VODE, but is written in modern Fortran and deploys contemporary best-practices for scientific software development. We will briefly sketch out the principles of an implicit integrator.

For the network we will be considering, we solve the typically stiff system

$$\frac{dX_k}{dt} = \dot{\omega}_k(\rho, X_k, T), \tag{5.2}$$

$$\frac{dT}{dt} = f(\dot{\omega}_k), \tag{5.3}$$

where X_k is the mass fraction of the k^{th} isotope in the network, ρ is density, T is temperature, and $\dot{\omega}$ is the reaction rate.

If we let the right-hand-side in the above be $\boldsymbol{f}(\boldsymbol{y})$ where $\boldsymbol{y}$ is a vector of mass fractions and temperature, implicit stiff ODE solvers use Newton-Raphson-like

iterative methods to solve

$$\boldsymbol{\Delta y} = \left(\frac{\mathbf{I}}{\Delta t} - \mathbf{J}(\boldsymbol{y}^n) \right)^{-1} \boldsymbol{f}(\boldsymbol{y}^n), \tag{5.4}$$

where $\mathbf{J}$ is the Jacobian matrix for $\boldsymbol{f}(\boldsymbol{y})$. Recall that both are provided by the chosen network.

The strategy for accelerating this code is as follows. The **first step** is to locate a *very* expensive loop to instrument. As a rough rule-of-thumb, an expensive loop should have about 100 000 iterations or more to be an attractive target for OpenACC. This should offer enough of a computational workload to achieve a net performance gain over threaded computation across CPU cores, even with the time needed to move data. In our experience, the more of your code that can be put onto the GPU the better. Even if parts of the compute region are not computationally intense, putting large sections of code on the GPU greatly simplifies data transfer and minimizes the amount of data to be managed.

The loop we instrument is the previously mentioned loop over a 3D grid of cells. The loop can be found in the source code under the vbdf_oac branch of the Maestro repository in the burner_loop_3d() subroutine of MAESTRO/Source/react_state.f90. A simplified overview looks like

```
!$acc parallel loop gang vector collapse(3)  &
!$acc    private(rho,x_in,T_in,x_test,x_out) &
!$acc    private(rhowdot,rhoH,sumX,n)
do k = lo(3), hi(3)
   do j = lo(2), hi(2)
      do i = lo(1), hi(1)
         call burner(rho, T_in,...
      end do
   end do
end do
!$acc end parallel
```

We use the previously described parallel loop compute construct with the gang vector clauses. We also have some new statements. collapse(3) tells the compiler to collapse the triply-nested loop into a single, large loop before accelerating. Without this clause, only the outermost loop would be parallelized. Other configurations should be explored, but this relatively simple configuration has proven to be capable of achieving significant performance and is sufficient for an initial exploration. private() is a data clause that tells the compiler each thread will need a private copy of the indicated data. Otherwise, the compiler must assume the data is potentially accessed by multiple threads and is thus

stored in global memory. Such memory will be shared among threads and thus should never be accessed by multiple iterations. It must be determined what data within a loop is needed for every loop iteration. In the above example, rho is the variable used to store the current cell's (the current i, j, k) density. If we did not declare it private, then all iterations would attempt to write and read to this variable resulting in incorrect behavior.

This loop is chosen because typical grid sizes for many problems are 48^3–64^3, satisfying the 100 000 iteration guideline. It is also at a relatively high level in the burning algorithm, meaning most of the computation will take place on the GPU.

The **second step** is to mark up *all* subroutines contained within the compute region so that a version of the routine will be compiled for the GPU. The instrumenting of the burner() subroutine looks like

```
subroutine burner(dens, temp,...
!$acc routine seq
...
```

The seq clause denotes the routine is sequential, gauranteeing it uses no compute clauses like gang or vector. It can be run on individual CUDA cores.

We have expressed the computation, but the default treatment of data will be very inefficient. So the **third step** is to express the optimal data transfer and ensure any needed module data is available on the GPU. In the same burner_loop_3d() subroutine we see that the compute region is contained within a data region. A reduced example of this data region looks like

```
!$acc data copyin(sold) copyout(snew)
...compute here...
!$acc end data
```

Here we make sure that the incoming state array sold is only copied and that the state resulting from the completed burn snew is only copied out. All array data in a data region should be explicitly managed to minimize data transfer. By default, scalar variables are managed well and need not be explicitly managed in most cases.

That takes care of the data in the scope of the burner_loop_3d() subroutine, but we still need to manage all module data that is needed for GPU computation. Again in the vbdf_oac branch, source files containing unstrucured data management include

```
Util/VBDF/bdf.f90
Microphysics/networks/ignition_simple_bdf/network.f90
```

A reduced overview of the data management in network.f90 looks like

```
module network
   ...
   real(kind=dp_t) :: aion(nspec)
   !$acc declare create(aion)
contains
   subroutine network_init()
      aion(ic12_)  = 12.0_dp_t
      !$acc update device(aion)
      ...
   end subroutine
end module network
```

Here we have declared that the array of atomic numbers for each species aion should be allocated (created) in the GPU. We have no initialized this array yet, so we should not waste time with any copying. We just need to reserve the memory. Later, when an initialization routine is called on the CPU we update the GPU memory with a update device() directive.

The steps enumerated above in an ideal case would yield working code that can run on the GPUs. Rarely will a case be ideal. The process of this development has included an extensive series of lessons learned and limitations discovered for code that is to be run on the GPU. In practice, developers will have to modify code within compute regions to conform to these limitations before arriving at a code that will actually run correctly on the GPU. We overview these in the next section.

5.3 Limitations and Lessons

The OpenACC standard has been evolving rapidly as have compilers implementing it[8]. This often means that what the standard appears to allow may not always actually work in practice. Compilers may not have fully implemented the standard and will often have restrictions on what code is valid on the GPU. Many of the limitations on the GPU are unclear until confronted in practice. This work has involved extensive experimentation and consultation with compiler and GPU experts. By describing our findings in the process of development, we hope to expedite the development process for others. Before describing various specific, technical limitations we overview broad lessons and strategy we have found to be effetive.

Perhaps the first significant lesson encountered is one already mentioned: put as much of your code on the GPU as feasible. This has to do in part with data transfer costs and the nature of typical scientific algorithms. If

[8]in fact, our code would not have been possible this time last year

a developer only accelerates "hot spots" in their code it is easy to arrive at a situation where the computation of the hot spot is indeed very rapid but such gains come at the cost of excessive communication between the host and device. It is possible to launch computation on the GPU and then continue executing subsequent code on the CPU, but scientific algorithms tend to have a linear nature preventing subsequent code from running before getting results from the GPU. It may be possible in some cases to redesign an algorithm to eliminate or reduce any such linearity, but it is hard to justify doing so in an initial accelerated implementation. It may be that such a redesign would still not yield a net performance gain. For rapid development and testing, we find the best strategy is to instead put all of the existing algorithm on the GPU. This can be achieved by putting any OpenACC compute region at a sufficiently high level in the code's call tree.

This does have a cost. Putting a larger, more complex region of code on the GPU requires all such code must be modified to conform to GPU restrictions. It also requires greater overhead for the compiler-generate GPU code and typically means more of the GPU's scare fast memory is needed. In particular, recall that each thread's private data is stored in the registers. The more registers per thread a code requires, the fewer threads it is able to spawn on the GPU. This can prevent code from benefitting from latency hiding. Further, if each thread requires a relatively large amount of data then it can experience what is called register spillage. The data needed by a thread may not be available in the register, requiring access of slower memory. Now not only does the code have fewer threads, but those threads are slower. We are still exploring techniques for ammeliorating this register pressure. As the standard and compilers evolve, it may be worth exploring a nested GPU parallelism model. In this model, we still put the same large region of code on the GPU but we configure that code to run effectively in serial on a single SMX. This code will then spawn, from the GPU, new compute regions that fully utilize it will be launched. The benefit, in theory, is that the new compute regions are more lightweight with less register pressure. This is possible with low-level languages like CUDA and is technically permissible in the latest OpenACC standard, but it is not widely implemented by compilers yet. For the code discussed here, an analysis using NVIDIA's `nvprof` and `nvvp` tools reveals we indeed have significant register pressure, but it is not severe enough to eliminate a net performance gain. This suggests that even this performant code has room for significant tuning and further performance gains.

In addition to these broad lessons, we find many restrictions on code to be executed on the GPU. Note that these restrictions apply to the PGI 16.3 compiler and the Fortran implementation of OpenACC. All code was testing

on Titan nodes. The restrictions include:

- **allocation:** Fortran's allocate statement is not permitted on the GPU. The memory model is quite different than that of a CPU and this statement simply does not have a straight-forward, stable meaning on the GPU that yields the same behavior as on the CPU. Any dynamic memory needed in GPU code will have to be allocated on the CPU.

- **Fortran character arrays:** Fortran's character array type is not implemented on the GPU. We have often found such variables can be converted to integers to achieve the same functionality.

- **I/O:** I/O is limited on the GPU. It is permissible to print output in the list-directected format (print *, var1, var2), but the number of such statements that will be handled may be restricted and less trivial I/O is likely to fail.

- **intrinsics:** Not all of Fortran's intrinsic functions and subroutines will be available on the GPU. Examples we found include `eoshift()` and `minloc()`.

- **functions as arguments:** Our integrator was originally designed to take the right-hand-side subroutine provided by the network as an argument. However, passing functions as arguments is not permitted on the GPU.

- **multiple exits and branching:** The SIMD nature of the fine-grained parallelism on GPUs means that subroutines with multiple exit points or loops with multiple possible exit points should be avoided. We have found that sometimes this causes incorrect behavior, but even when it behaves properly one must be wary of performance impacts. When a section of code with multiple possible branches is compiled for the GPU, it is done in such a way that *all* threads execute *all* branches. This means if only a single thread in a block of threads needs to execute a divergent branch, all threads in the block will have to execute it as well. They will not actually have their data or behavior impacted, so in effect it is as if all threads have to wait for the one divergent threat to execute. The more diverging branches, the more risk of significant performance losses.

- **Fortran's slice and array notation:** Fortran conveniently allows array operations, such as a = b + c where a, b, and c are arrays of the same shape and size. Fortran will add the arrays b and c element-wise

and store the result in a. Slices (subsets) of arrays can also be used in this fashion, such as a(lo:hi) = b(lo:hi) + c(lo:hi), with lo and hi being boundaries of the slice. We have found using this valid syntax often triggers errors on the GPU. We suspect that often the compiler has optimizations for carrying out operations like this, but the optimization fails on the GPU. To be safe, array syntax can be converted to explicit loops.

- **automatic arrays:** An automatic array is an array that is an argument in a subroutine that has its size passed in as an argument as well. This implicitly requires allocation (which is permissible for the compiler, just not the developer). While automatic arrays will behave correctly on the GPU, we have found they can have a significant impact on performance, especially if they are used in subroutines that are called many times by all threads. In one instance, use of automatic arrays resulted in code running about 150 times slower!

5.4 Performance

Having provided an overview of our code and OpenACC development, we now turn to the performance of the code. This analysis is based on the unit test found in the vbdf_oac branch of the Maestrorepository at MAESTRO/Exec/UNIT_TESTS/test_react_bdf. The test was configured to use a modified version of the ignition_simple network, which can be found at MAESTRO/Microphysics/networks/ignition_simple_bdf. The VBDF integrator, discussed previously, is utilized and can be found at MAESTRO/Util/VBDF/bdf.f90. We compile using the 16.3 version of the PGI compilers and we execute on a node of the Titan supercomputer.

Fig. 5.3 plots the runtime for the nuclear burning in different configurations. We plot the "perfect" threaded performance, which on Titan is simply the time to run in serial on the CPU divided by the node's 16 cores. We also plot runtimes for two GPU versions, one without using pinned memory (the default) and one with. All plots are versus the problem size as determined by the sidelength of the 3D cube of data being burned. We consider cubes of $16^3, 32^3, 48^3$, and 64^3.

Recall that pinned memory will not be paged. This allows the GPU to directly access it safely. By default, code running on the CPU will utilize paged memory. This creates an extra step in moving data from the CPU to the GPU because it must first be moved to the pinned region and then the GPU will move that to its local RAM. We have found that under some circumstances this can result in major performance losses.

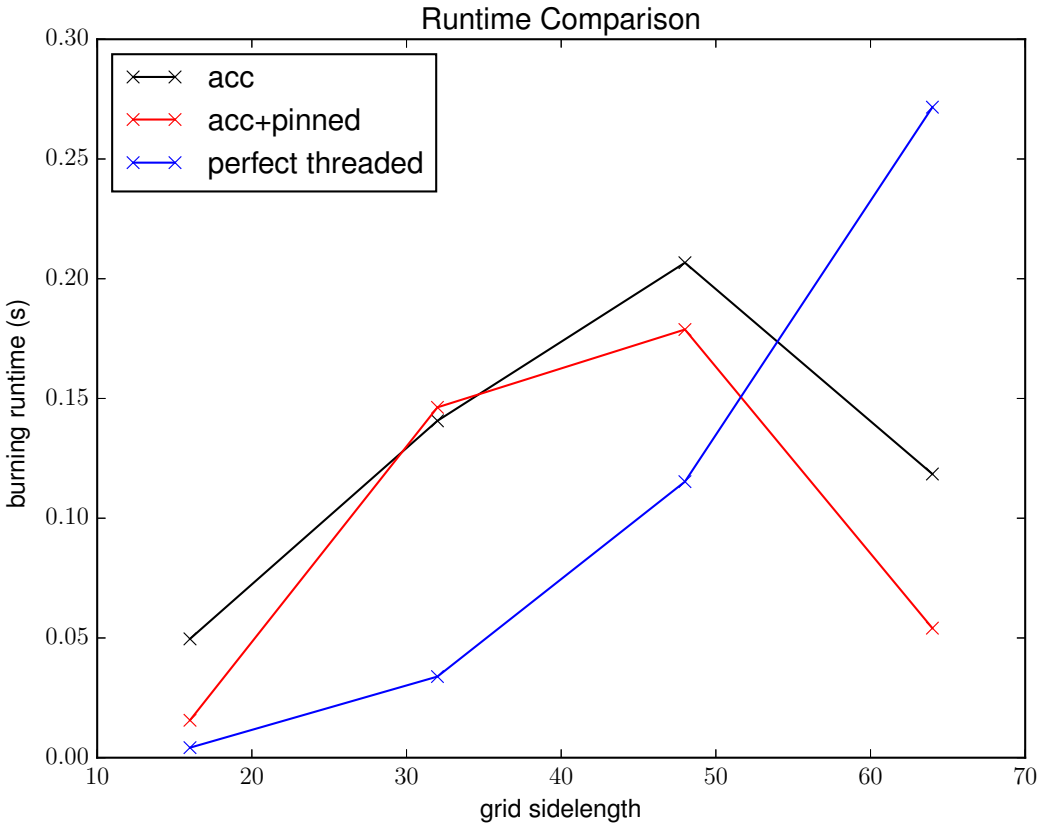


Figure 5.3 Runtimes of the burning algorithm in different configurations. See text for details.

Fortunately, the compiler can be configured to have the CPU directly utilize pinned memory. Due to the extra overhead of the implementation, this means all allocations and deallocations on CPU will be slower, but the data transfer between the CPU and GPU will be faster. We can do even better by utilizing a small amount of CUDA Fortran to specify precisely which data should be allocated directly into pinned memory.

We see in the figure for the larger cube sizes, using pinned memory is much faster. We caution this ability to achieve such a significant performance gain is in part a result of the simple network being used. Only one isotope and temperature are integrated. For larger networks, computation costs will strongly dominate data costs and tuning data transfer performance may only yield negligible gains.

Only for the largest cube, 64^3, do we see the desired performance: the GPU is significantly faster than the threaded CPU. This emphasizes how important it is to have enough computation to justify using the GPU. In our case, the transition from a 48^3 grid to 64^3 one is where we see the GPU starts to beat the CPU.

5.5 The State of OpenACC

Developing this code required extensive consulation with compiler experts, GPU experts, and OpenACC experts (made possible via our INCITE allocation on Titan and a GPU hackathon hosted by Oak Ridge). Many bugs in the compilers and unexpected behavior were encountered, some of which has been discussed in the previous section. The promise of OpenACC is rapid porting of code to target GPUs (and other accelerators) with minimal need to completely refactor the code to be accelerated. Our experience is that this promise is largely satisfied with contemporary iterations of the OpenACC standard and contemporary implementations of the standard in PGI compilers. The development is by no means trivial, but ultimately we were able to achieve a performing unit test without the need to use low-level, less portable code like CUDA. This was not the case even a year ago, illustrating the significant investment that has been put into both the standard and compilers.

To our knowledge, this code is the only example of an open source reaction network integrator targetting GPUs via OpenACC. The lessons learned in this development will be applied to the creation of an open source repository of microphysics modules that will be usable by a range of astrophysical codes, not just BoxLib codes.

Chapter 6

Conclusions & Discussion

We have explored 18 physically motivated, simple models of convective burning in sub-Chandrasekhar carbon/oxygen white dwarfs with an accreted, low-mass helium shell. These systems have been modeled in 3D with detailed microphysics using the low-Mach hydrodynamics code Maestro. This is the broadest suite of 3D simulations of this phase of evolution. In addition, we have modified the nuclear reaction modules of Maestroto utilize GPU accelerators, achieving performance gains for sufficiently large cubes of data.

As discussed in Ch. 2, our models are simple and limited in the number of species and reactions we track. This is sufficient for our focus on exploring a large number of model systems and capturing the dominant energetics and general dynamical trends. Our findings here will serve as the foundation for future work focused on a smaller number of more detailed models. From this set of simple models we can draw several important insights into potential double detonation SNe Ia or .Ia progenitors as well as other runaway events involving sub-Chandrasekhar white dwarfs with envelopes of accreted helium, such as possible helium novae.

We find that localized runaway is indeed achievable in 3D. Whether this develops into a deflagration or detonation requires more study. Of particular interest is making use of the Maestro-to-`Castro` mapping developed in Malone et al. (2014) to map Maestro's 3D state to `Castro`, enabling us to follow the near- and super-sonic evolution. Like Woosley and Kasen (2011), we find that hotter cores allow for localized runaway to develop at lower densities than in models with cooler cores. Thus, hotter cores are able to achieve runaway with thinner (lower mass) helium shells than cooler cores of the same mass. But even within the low-Mach framework, we have developed a methodology for analyzing localized runaway in our models. Initial results are suggestive that detonation can be seeded in the helium shell and that this manifests at multiple points for model 08130F.

Our results indicate that the complex dynamics of 3D convection should be a component of investigations of double detonation progenitor models. In cases of localized runaway, we find significant convective overshoot and a steady freeze-out of convection. These effects substantially impact the density, temperature, and composition of the region in which localized runaway is initiated.

This dissertation provides a foundation for a variety of exciting scientific inquiries. Most straight-forward are those hinted at throughout related to sub-M_{Ch} systems. These systems are fascinating in part because they have been proposed to be related to a variety of possible astrophysical transients. How many such proposals can be borne out? Can these systems be progenitors for multiple transients? If so, what system properties determine the transient outcome?

Let us first consider transients that leave the core intact. Work to date is suggestive that these events are more probable for lower mass cores, roughly $\leq 0.8\ M_{\odot}$. The methods developed here should be applied to a suite of lower mass cores to investigate such possibilities. In the case of ".Ia" events, where a detonation of the shell is expected, a study could be carried out using the increasingly common workflow of calculating initial conditions with Maestro to be utilized in compressible codes like `Castro` to capture the evolution of any possible detonations or otherwise near- or trans-sonic flow. There may be ranges of core masses which are capable of sustaining a shell-only detonation for some shell masses and a double-detonation for more massive shells. This leads to interesting questions of achievable accretion rates and possibly complex evolutionary histories. Can a system experience multiple shell-only transients? Such questions have yet to be confronted with 3D models of the pre-explosive evolution to the author's knowledge. There may also be a shell mass regime for which the shell-only runaway manifests as a deflagration, which will have significantly different nucleosynthetic yields. Woosley and Kasen (2011) suggest such deflagrations may account for the observed solar abundance of ^{44}Ca. Finally, there may be regimes where like those we call convective runaway in this work. How do these shell-only events relate to possible novae driven by helium burning instead of hydrogen?

The potential of these systems to yield such a variety of transients should make possible rather tight constraints on theoretical models. If we can trace multiple events occurring at different phases and system configurations we can more stringently test progenitor models than is possible for a model leading to only one type of event. With lower mass WDs being more common and more sensitive transient surveys being developed, it may be that a robust understanding informed by a rich set of data can be developed for the lower mass systems, informing understanding of rarer systems with higher mass

cores.

Such higher mass core models currently appear likely to be the site of events that disrupt the entire star, namely double-detonation SNe Ia's. To make progress in understanding double-detonations, we plan to expand our models with larger reaction networks and more realistic initial models. Our work developing OpenACC-accelerated nuclear reaction modules facilitates this expansion. This work is being integrated into a stand-alone, open source microphysics code repository. We are collaborating with developers of other leading astrophysical codes to make these modules generally usable by the community, not just by our codes. We also plan to collaborate with experimentalists and developers of nuclear reaction rate libraries to ensure this repository gives theorists access to the latest available data. Currently, codes tend to incorporate their own, hard-coded reaction rates that can be out-of-date. Greater collaboration among analytic theorists, computational theorists, and nuclear astrophysics experimentalists also offers the possibility of new insights and ripe opportunities for interdisciplinary investigation.

Finally, we note that the methodology developed here may have applications outside of WD transients. There is growing interest in calculating 3D initial conditions for core-collapse supernovae. The progenitors for these systems are dominated by convective burning in shells, similar to that on the surface of the WDs reported here. We hope to collaborate on projects expanding this work to calculate the properties of such progenitors, perhaps making possible the resolution of the long-standing question of how stalled shocks gain enough energy to unbind the star.

Bibliography

A. M. Jacobs, M. Zingale, A. Nonaka, A. S. Almgren, and J. B. Bell. Low mach number modeling of convection in helium shells on sub-chandrasekhar white dwarfs. ii. bulk properties of simple models. *The Astrophysical Journal*, 827 (1):84, 2016. URL http://stacks.iop.org/0004-637X/827/i=1/a=84.

W. Hillebrandt, M. Kromer, F.K. Röpke, and A.J. Ruiter. Towards an understanding of type ia supernovae from a synthesis of theory and observations. *Frontiers of Physics*, 8(2):116–143, 2013. ISSN 2095-0462. doi: 10.1007/s11467-013-0303-2. URL http://dx.doi.org/10.1007/s11467-013-0303-2.

A. G. Riess, A. V. Filippenko, P. Challis, A. Clocchiatti, A. Diercks, P. M. Garnavich, R. L. Gilliland, C. J. Hogan, S. Jha, R. P. Kirshner, B. Leibundgut, M. M. Phillips, D. Reiss, B. P. Schmidt, R. A. Schommer, R. C. Smith, J. Spyromilio, C. Stubbs, N. B. Suntzeff, and J. Tonry. Observational Evidence from Supernovae for an Accelerating Universe and a Cosmological Constant. *AJ*, 116:1009–1038, September 1998. doi: 10.1086/300499.

S. Perlmutter, G. Aldering, G. Goldhaber, R. A. Knop, P. Nugent, P. G. Castro, S. Deustua, S. Fabbro, A. Goobar, D. E. Groom, I. M. Hook, A. G. Kim, M. Y. Kim, J. C. Lee, N. J. Nunes, R. Pain, C. R. Pennypacker, R. Quimby, C. Lidman, R. S. Ellis, M. Irwin, R. G. McMahon, P. Ruiz-Lapuente, N. Walton, B. Schaefer, B. J. Boyle, A. V. Filippenko, T. Matheson, A. S. Fruchter, N. Panagia, H. J. M. Newberg, W. J. Couch, and T. S. C. Project. Measurements of Ω and Λ from 42 High-Redshift Supernovae. *ApJ*, 517:565–586, June 1999. doi: 10.1086/307221.

D. Branch, A. Fisher, and P. Nugent. On the relative frequencies of spectroscopically normal and peculiar type IA supernovae. *AJ*, 106:2383–2391, December 1993. doi: 10.1086/116810.

M. M. Phillips. The absolute magnitudes of Type IA supernovae. *ApJ*, 413: L105–L108, August 1993. doi: 10.1086/186970.

W. Li, J. Leaman, R. Chornock, A. V. Filippenko, D. Poznanski, M. Ganeshalingam, X. Wang, M. Modjaz, S. Jha, R. J. Foley, and N. Smith. Nearby supernova rates from the Lick Observatory Supernova Search - II. The observed luminosity functions and fractions of supernovae in a complete sample. *MNRAS*, 412:1441–1472, April 2011. doi: 10.1111/j.1365-2966.2011.18160.x.

M. Livio. The Progenitors of Type Ia Supernovae. In J. C. Niemeyer and J. W. Truran, editors, *Type Ia Supernovae, Theory and Cosmology*, page 33, 2000.

K. Nomoto, F.-K. Thielemann, and K. Yokoi. Accreting white dwarf models of Type I supernovae. III - Carbon deflagration supernovae. *ApJ*, 286:644–658, November 1984. doi: 10.1086/162639.

M. Fink, M. Kromer, I. R. Seitenzahl, F. Ciaraldi-Schoolmann, F. K. Röpke, S. A. Sim, R. Pakmor, A. J. Ruiter, and W. Hillebrandt. Three-dimensional pure deflagration models with nucleosynthesis and synthetic observables for Type Ia supernovae. *MNRAS*, 438:1762–1783, February 2014. doi: 10.1093/mnras/stt2315.

A. M. Khokhlov. Delayed detonation model for type IA supernovae. *A&A*, 245:114–128, May 1991.

F. K. Röpke and J. C. Niemeyer. Delayed detonations in full-star models of type Ia supernova explosions. *A&A*, 464:683–686, March 2007. doi: 10.1051/0004-6361:20066585.

G. C. Jordan, IV, R. T. Fisher, D. M. Townsley, A. C. Calder, C. Graziani, S. Asida, D. Q. Lamb, and J. W. Truran. Three-Dimensional Simulations of the Deflagration Phase of the Gravitationally Confined Detonation Model of Type Ia Supernovae. *ApJ*, 681:1448–1457, July 2008. doi: 10.1086/588269.

T. Plewa, A. C. Calder, and D. Q. Lamb. Type Ia Supernova Explosion: Gravitationally Confined Detonation. *ApJ*, 612:L37–L40, September 2004. doi: 10.1086/424036.

A. C. Calder, B. K. Krueger, A. P. Jackson, and D. M. Townsley. The influence of chemical composition on models of Type Ia supernovae. *Frontiers of Physics*, 8:168–188, April 2013. doi: 10.1007/s11467-013-0301-4.

K. Nomoto. White dwarf models for type 1 supernovae and quiet supernovae, and presupernova evolution. *Space Sci. Rev.*, 27:563–570, November 1980.

S. E. Woosley, T. A. Weaver, and R. E. Taam. Models for Type I supernovae. In J. C. Wheeler, editor, *Texas Workshop on Type I Supernovae*, pages 96–112, 1980.

K. Nomoto. Accreting white dwarf models for type 1 supernovae. II - Off-center detonation supernovae. *ApJ*, 257:780–792, June 1982a. doi: 10.1086/160031.

E. Livne. Successive detonations in accreting white dwarfs as an alternative mechanism for type I supernovae. *ApJ*, 354:L53–L55, May 1990. doi: 10.1086/185721.

E. Livne and A. S. Glasner. Geometrical effects in off-center detonation of helium shells. *ApJ*, 361:244–250, September 1990. doi: 10.1086/169189.

E. Livne and A. S. Glasner. Numerical simulations of off-center detonations in helium shells. *ApJ*, 370:272–281, March 1991. doi: 10.1086/169813.

S. E. Woosley and T. A. Weaver. Sub-Chandrasekhar mass models for Type IA supernovae. *ApJ*, 423:371–379, March 1994. doi: 10.1086/173813.

E. Livne and D. Arnett. Explosions of Sub–Chandrasekhar Mass White Dwarfs in Two Dimensions. *ApJ*, 452:62, October 1995. doi: 10.1086/176279.

M. H. van Kerkwijk, P. Chang, and S. Justham. Sub-Chandrasekhar White Dwarf Mergers as the Progenitors of Type Ia Supernovae. *ApJ*, 722:L157–L161, October 2010. doi: 10.1088/2041-8205/722/2/L157.

A. J. Ruiter, K. Belczynski, S. A. Sim, W. Hillebrandt, C. L. Fryer, M. Fink, and M. Kromer. Delay times and rates for Type Ia supernovae and thermonuclear explosions from double-detonation sub-Chandrasekhar mass models. *MNRAS*, 417:408–419, October 2011. doi: 10.1111/j.1365-2966.2011.19276.x.

S. E. Woosley and D. Kasen. Sub-Chandrasekhar Mass Models for Supernovae. *ApJ*, 734:38, June 2011. doi: 10.1088/0004-637X/734/1/38.

L. Bildsten, K. J. Shen, N. N. Weinberg, and G. Nelemans. Faint Thermonuclear Supernovae from AM Canum Venaticorum Binaries. *ApJ*, 662:L95–L98, June 2007. doi: 10.1086/519489.

B. Wang, S. Justham, and Z. Han. Producing Type Iax supernovae from a specific class of helium-ignited WD explosions. *A&A*, 559:A94, November 2013. doi: 10.1051/0004-6361/201322298.

T. Kato, K. Kanatsu, K. Takamizawa, A. Takao, and R. Stubbings. Possible Nova in Puppis. *IAU Circ.*, 7552, December 2000.

N. M. Ashok and D. P. K. Banerjee. The enigmatic outburst of V445 Puppis - A possible helium nova? *A&A*, 409:1007–1015, October 2003. doi: 10.1051/0004-6361:20031160.

T. Iijima and H. Nakanishi. Spectroscopic observations of the first helium nova V445 Puppis. *A&A*, 482:865–877, May 2008. doi: 10.1051/0004-6361:20077502.

P. Hoeflich and A. Khokhlov. Explosion Models for Type IA Supernovae: A Comparison with Observed Light Curves, Distances, H 0, and Q 0. *ApJ*, 457:500, February 1996. doi: 10.1086/176748.

P. Nugent, E. Baron, D. Branch, A. Fisher, and P. H. Hauschildt. Synthetic Spectra of Hydrodynamic Models of Type IA Supernovae. *ApJ*, 485:812, August 1997. doi: 10.1086/304459.

Brian Warner. The AM canum venaticorum stars. *Astrophysics and Space Science*, 225(2):249–270, 1995. ISSN 0004-640X. doi: 10.1007/BF00613240. URL http://adsabs.harvard.edu/abs/1995Ap{%}26SS.225..249W.

G. Nelemans. AM CVn stars. *The Astrophysics of Cataclysmic Variables and Related Objects*, 330, 2005. URL http://adsabs.harvard.edu/abs/2005ASPC..330...27N.

S. E. Woosley, R. E. Taam, and T. A. Weaver. Models for Type I supernova. I - Detonations in white dwarfs. *ApJ*, 301:601–623, February 1986. doi: 10.1086/163926.

D. García-Senz, E. Bravo, and S. E. Woosley. Single and multiple detonations in white dwarfs. *A&A*, 349:177–188, September 1999.

K. Nomoto. Accreting white dwarf models for type I supernovae. I - Presupernova evolution and triggering mechanisms. *ApJ*, 253:798–810, February 1982b. doi: 10.1086/159682.

Z. Ivezic, J. A. Tyson, B. Abel, E. Acosta, R. Allsman, Y. AlSayyad, S. F. Anderson, J. Andrew, R. Angel, G. Angeli, R. Ansari, P. Antilogus, K. T. Arndt, P. Astier, E. Aubourg, T. Axelrod, D. J. Bard, J. D. Barr, A. Barrau, J. G. Bartlett, B. J. Bauman, S. Beaumont, A. C. Becker, J. Becla, C. Beldica, S. Bellavia, G. Blanc, R. D. Blandford, J. S. Bloom, J. Bogart,

K. Borne, J. F. Bosch, D. Boutigny, W. N. Brandt, M. E. Brown, J. S. Bullock, P. Burchat, D. L. Burke, G. Cagnoli, D. Calabrese, S. Chandrasekharan, S. Chesley, E. C. Cheu, J. Chiang, C. F. Claver, A. J. Connolly, K. H. Cook, A. Cooray, K. R. Covey, C. Cribbs, W. Cui, R. Cutri, G. Daubard, G. Daues, F. Delgado, S. Digel, P. Doherty, R. Dubois, G. P. Dubois-Felsmann, J. Durech, M. Eracleous, H. Ferguson, J. Frank, M. Freemon, E. Gangler, E. Gawiser, J. C. Geary, P. Gee, M. Geha, R. R. Gibson, D. K. Gilmore, T. Glanzman, I. Goodenow, W. J. Gressler, P. Gris, A. Guyonnet, P. A. Hascall, J. Haupt, F. Hernandez, C. Hogan, D. Huang, M. E. Huffer, W. R. Innes, S. H. Jacoby, B. Jain, J. Jee, J. G. Jernigan, D. Jevremovic, K. Johns, R. L. Jones, C. Juramy-Gilles, M. Juric, S. M. Kahn, J. S. Kalirai, N. Kallivayalil, B. Kalmbach, J. P. Kantor, M. M. Kasliwal, R. Kessler, D. Kirkby, L. Knox, I. Kotov, V. L. Krabbendam, S. Krughoff, P. Kubanek, J. Kuczewski, S. Kulkarni, R. Lambert, L. Le Guillou, D. Levine, M. Liang, K. Lim, C. Lintott, R. H. Lupton, A. Mahabal, P. Marshall, S. Marshall, M. May, R. McKercher, M. Migliore, M. Miller, D. J. Mills, D. G. Monet, M. Moniez, D. R. Neill, J. Nief, A. Nomerotski, M. Nordby, P. O'Connor, J. Oliver, S. S. Olivier, K. Olsen, S. Ortiz, R. E. Owen, R. Pain, J. R. Peterson, C. E. Petry, F. Pierfederici, S. Pietrowicz, R. Pike, P. A. Pinto, R. Plante, S. Plate, P. A. Price, M. Prouza, V. Radeka, J. Rajagopal, A. Rasmussen, N. Regnault, S. T. Ridgway, S. Ritz, W. Rosing, C. Roucelle, M. R. Rumore, S. Russo, A. Saha, B. Sassolas, T. L. Schalk, R. H. Schindler, D. P. Schneider, G. Schumacher, J. Sebag, G. H. Sembroski, L. G. Seppala, I. Shipsey, N. Silvestri, J. A. Smith, R. C. Smith, M. A. Strauss, C. W. Stubbs, D. Sweeney, A. Szalay, P. Takacs, J. J. Thaler, R. Van Berg, D. Vanden Berk, K. Vetter, F. Virieux, B. Xin, L. Walkowicz, C. W. Walter, D. L. Wang, M. Warner, B. Willman, D. Wittman, S. C. Wolff, W. M. Wood-Vasey, P. Yoachim, H. Zhan, and for the LSST Collaboration. LSST: from Science Drivers to Reference Design and Anticipated Data Products. *ArXiv e-prints*, May 2008.

M. Fink, W. Hillebrandt, and F. K. Röpke. Double-detonation supernovae of sub-Chandrasekhar mass white dwarfs. *A&A*, 476:1133–1143, December 2007. doi: 10.1051/0004-6361:20078438.

M. Fink, F. K. Röpke, W. Hillebrandt, I. R. Seitenzahl, S. A. Sim, and M. Kromer. Double-detonation sub-Chandrasekhar supernovae: can minimum helium shell masses detonate the core? *A&A*, 514:A53, May 2010. doi: 10.1051/0004-6361/200913892.

K. J. Shen and L. Bildsten. The Ignition of Carbon Detonations via Converging

Shock Waves in White Dwarfs. *ApJ*, 785:61, April 2014. doi: 10.1088/0004-637X/785/1/61.

R Moll and SE Woosley. Multi-Dimensional Double Detonation of Sub-Chandrasekhar Mass White Dwarfs. *arXiv preprint arXiv:1303.0324*, 2013. URL http://arxiv.org/abs/1303.0324.

Dean M. Townsley, Kevin Moore, and Lars Bildsten. LATERALLY PROPAGATING DETONATIONS IN THIN HELIUM LAYERS ON ACCRETING WHITE DWARFS. *The Astrophysical Journal*, 755(1):4, August 2012. ISSN 0004-637X. doi: 10.1088/0004-637X/755/1/4. URL http://iopscience.iop.org/0004-637X/755/1/4/fulltext/.

S. A. Sim, F. K. Röpke, W. Hillebrandt, M. Kromer, R. Pakmor, M. Fink, A. J. Ruiter, and I. R. Seitenzahl. Detonations in Sub-Chandrasekhar-mass C+O White Dwarfs. *ApJ*, 714:L52–L57, May 2010. doi: 10.1088/2041-8205/714/1/L52.

M. Kromer, S. A. Sim, M. Fink, F. K. Röpke, I. R. Seitenzahl, and W. Hillebrandt. Double-detonation Sub-Chandrasekhar Supernovae: Synthetic Observables for Minimum Helium Shell Mass Models. *ApJ*, 719:1067–1082, August 2010. doi: 10.1088/0004-637X/719/2/1067.

Zheng-Wei Liu, Richard J. Stancliffe, C. Abate, and B. Wang. PRE-EXPLOSION COMPANION STARS IN TYPE Iax SUPERNOVAE. *The Astrophysical Journal*, 808(2):138, jul 2015a. ISSN 1538-4357. doi: 10.1088/0004-637X/808/2/138. URL http://adsabs.harvard.edu/abs/2015ApJ...808..138L.

Zheng-Wei Liu, Takashi J. Moriya, Richard J. Stancliffe, and Bo Wang. Constraints on single-degenerate Chandrasekhar mass progenitors of Type Iax supernovae. *Astronomy & Astrophysics*, 574:A12, jan 2015b. ISSN 0004-6361. doi: 10.1051/0004-6361/201424532. URL http://adsabs.harvard.edu/abs/2015A{%}26A...574A..12L.

S. Geier, T. R. Marsh, B. Wang, B. Dunlap, B. N. Barlow, V. Schaffenroth, X. Chen, A. Irrgang, P. F. L. Maxted, E. Ziegerer, T. Kupfer, B. Miszalski, U. Heber, Z. Han, A. Shporer, J. H. Telting, B. T. Gänsicke, R. H. Ø stensen, S. J. O'Toole, and R. Napiwotzki. A progenitor binary and an ejected mass donor remnant of faint type Ia supernovae. *Astronomy & Astrophysics*, 554:A54, June 2013. ISSN 0004-6361. doi: 10.1051/0004-6361/201321395. URL http://adsabs.harvard.edu/abs/2013A%26A...554A..54G.

W. R. Brown, M. Kilic, C. Allende Prieto, and S. J. Kenyon. The merger rate of extremely low mass white dwarf binaries: links to the formation of AM CVn stars and underluminous supernovae. *MNRAS*, 411:L31–L35, February 2011. doi: 10.1111/j.1745-3933.2010.00986.x.

M. R. Drout, A. M. Soderberg, P. A. Mazzali, J. T. Parrent, R. Margutti, D. Milisavljevic, N. E. Sanders, R. Chornock, R. J. Foley, R. P. Kirshner, A. V. Filippenko, W. Li, P. J. Brown, S. B. Cenko, S. Chakraborti, P. Challis, A. Friedman, M. Ganeshalingam, M. Hicken, C. Jensen, M. Modjaz, H. B. Perets, J. M. Silverman, and D. S. Wong. The Fast and Furious Decay of the Peculiar Type Ic Supernova 2005ek. *ApJ*, 774:58, September 2013. doi: 10.1088/0004-637X/774/1/58.

Ken J. Shen and Kevin Moore. The Initiation and Propagation of Helium Detonations in White Dwarf Envelopes. *The Astrophysical Journal*, 797(1): 46, November 2014. ISSN 1538-4357. doi: 10.1088/0004-637X/797/1/46. URL http://adsabs.harvard.edu/abs/2014ApJ...797...46S.

Anthony L. Piro. TURBULENT MIXING ON HELIUM-ACCRETING WHITE DWARFS. *ApJ*, 801(2):137, March 2015. ISSN 1538-4357. doi: 10.1088/0004-637X/801/2/137. URL http://iopscience.iop.org/0004-637X/801/2/137/article/.

M. Zingale, A. Nonaka, A. S. Almgren, J. B. Bell, C. M. Malone, and R. J. Orvedahl. Low Mach Number Modeling of Convection in Helium Shells on Sub-Chandrasekhar White Dwarfs. I. Methodology. *ApJ*, 764:97, February 2013. doi: 10.1088/0004-637X/764/1/97.

M. S. Day and J. B. Bell. Numerical simulation of laminar reacting flows with complex chemistry. *Combustion Theory and Modelling*, 4:535–556, December 2000. doi: 10.1088/1364-7830/4/4/309.

J. B. Bell, M. S. Day, C. A. Rendleman, S. E. Woosley, and M. A. Zingale. Adaptive low Mach number simulations of nuclear flame microphysics. *Journal of Computational Physics*, 195:677–694, April 2004a. doi: 10.1016/j.jcp.2003.10.035.

J. B. Bell, M. S. Day, C. A. Rendleman, S. E. Woosley, and M. Zingale. Direct Numerical Simulations of Type Ia Supernovae Flames. I. The Landau-Darrieus Instability. *ApJ*, 606:1029–1038, May 2004b. doi: 10.1086/383023.

M. Zingale, S. E. Woosley, C. A. Rendleman, M. S. Day, and J. B. Bell. Three-dimensional Numerical Simulations of Rayleigh-Taylor Unstable Flames in

Type Ia Supernovae. *ApJ*, 632:1021–1034, October 2005. doi: 10.1086/433164.

A. J. Aspden, J. B. Bell, M. S. Day, S. E. Woosley, and M. Zingale. Turbulence-Flame Interactions in Type Ia Supernovae. *ApJ*, 689:1173-1185, December 2008. doi: 10.1086/592726.

A. S. Almgren, J. B. Bell, C. A. Rendleman, and M. Zingale. Low Mach Number Modeling of Type Ia Supernovae. I. Hydrodynamics. *ApJ*, 637: 922–936, February 2006a. doi: 10.1086/498426.

A. S. Almgren, J. B. Bell, C. A. Rendleman, and M. Zingale. Low Mach Number Modeling of Type Ia Supernovae. II. Energy Evolution. *ApJ*, 649: 927–938, October 2006b. doi: 10.1086/507089.

A. S. Almgren, J. B. Bell, A. Nonaka, and M. Zingale. Low Mach Number Modeling of Type Ia Supernovae. III. Reactions. *ApJ*, 684:449–470, September 2008. doi: 10.1086/590321.

M. Zingale, A. S. Almgren, J. B. Bell, A. Nonaka, and S. E. Woosley. Low Mach Number Modeling of Type IA Supernovae. IV. White Dwarf Convection. *ApJ*, 704:196–210, October 2009a. doi: 10.1088/0004-637X/704/1/196.

A. Nonaka, A. S. Almgren, J. B. Bell, M. J. Lijewski, C. M. Malone, and M. Zingale. MAESTRO: An Adaptive Low Mach Number Hydrodynamics Algorithm for Stellar Flows. *ApJS*, 188:358–383, June 2010. doi: 10.1088/0067-0049/188/2/358.

A. Nonaka, A. J. Aspden, M. Zingale, A. S. Almgren, J. B. Bell, and S. E. Woosley. HIGH-RESOLUTION SIMULATIONS OF CONVECTION PRECEDING IGNITION IN TYPE Ia SUPERNOVAE USING ADAPTIVE MESH REFINEMENT. *The Astrophysical Journal*, 745(1):73, January 2012. ISSN 0004-637X. doi: 10.1088/0004-637X/745/1/73. URL http://adsabs.harvard.edu/abs/2012ApJ...745...73N.

F. X. Timmes and F. D. Swesty. The Accuracy, Consistency, and Speed of an Electron-Positron Equation of State Based on Table Interpolation of the Helmholtz Free Energy. *ApJS*, 126:501–516, February 2000. doi: 10.1086/313304.

F. X. Timmes. Stellar Equations Of State, 2008. URL http://cococubed.asu.edu/code_pages/eos.shtml.

Ken J. Shen and Lars Bildsten. UNSTABLE HELIUM SHELL BURNING ON ACCRETING WHITE DWARFS. *The Astrophysical Journal*, 699(2):1365–1373, July 2009. ISSN 0004-637X. doi: 10.1088/0004-637X/699/2/1365. URL http://adsabs.harvard.edu/abs/2009ApJ...699.1365S.

G. R. Caughlan and W. A. Fowler. Thermonuclear Reaction Rates V. *Atomic Data and Nuclear Data Tables*, 40:283, 1988. doi: 10.1016/0092-640X(88)90009-5.

H. C. Graboske, H. E. Dewitt, A. S. Grossman, and M. S. Cooper. Screening Factors for Nuclear Reactions. II. Intermediate Screen-Ing and Astrophysical Applications. *ApJ*, 181:457–474, April 1973. doi: 10.1086/152062.

T. A. Weaver, G. B. Zimmerman, and S. E. Woosley. Presupernova evolution of massive stars. *ApJ*, 225:1021–1029, November 1978. doi: 10.1086/156569.

A. Alastuey and B. Jancovici. Nuclear reaction rate enhancement in dense stellar matter. *ApJ*, 226:1034–1040, December 1978. doi: 10.1086/156681.

F. Itoh, T. Honda, and K. Suzuki. Experimental and Monte Calro Studies of Multiple Scattering Correction in the Compton Profile of Aluminum. *Journal of the Physical Society of Japan*, 46:1201, April 1979.

T. A. Weaver and S. E. Woosley. Nucleosynthesis in massive stars and the ^{12}C(α, γ)^{16}O reaction rate. *Phys. Rep.*, 227:65–96, May 1993. doi: 10.1016/0370-1573(93)90058-L.

D. R. Garnett. Constraints on the ^{12}C(α,γ)^{16}O nuclear reaction rate from Hubble Space Telescope observations of C/(O) in low-abundance galaxies. *Nuclear Physics A*, 621:27–30, February 1997. doi: 10.1016/S0375-9474(97)00206-6.

M. Zingale, A. S. Almgren, J. B. Bell, A. Nonaka, and S. E. Woosley. Low Mach Number Modeling of Type IA Supernovae. IV. White Dwarf Convection. *ApJ*, 704:196–210, October 2009b. doi: 10.1088/0004-637X/704/1/196.

M. Zingale, A. Nonaka, A. S. Almgren, J. B. Bell, C. M. Malone, and S. E. Woosley. THE CONVECTIVE PHASE PRECEDING TYPE Ia SUPERNOVAE. *The Astrophysical Journal*, 740(1):8, October 2011. ISSN 0004-637X. doi: 10.1088/0004-637X/740/1/8. URL http://adsabs.harvard.edu/abs/2011ApJ...740....8Z.

Jared Brooks, Lars Bildsten, Pablo Marchant, and Bill Paxton. AM CANUM VENATICORUM PROGENITORS WITH HELIUM STAR DONORS

AND THE RESULTANT EXPLOSIONS. *The Astrophysical Journal*, 807 (1):74, July 2015. ISSN 1538-4357. doi: 10.1088/0004-637X/807/1/74. URL http://adsabs.harvard.edu/abs/2015ApJ...807...74B.

Y. B. Zel'Dovich, V. B. Librovich, G. M. Makhviladze, and G. I. Sivashinskil. On the onset of detonation in a nonuniformly heated gas. *Journal of Applied Mechanics and Technical Physics*, 11:264–270, March 1970. doi: 10.1007/BF00908106.

Peter N. Brown, G. D. Byrne, and A. C. Hindmarsh. Vode: A variable-coefficient ode solver. *SIAM J. Sci. Stat. Comput.*, 10(5):1038–1051, September 1989. ISSN 0196-5204. doi: 10.1137/0910062. URL http://dx.doi.org/10.1137/0910062.

C. M. Malone, A. Nonaka, S. E. Woosley, A. S. Almgren, J. B. Bell, S. Dong, and M. Zingale. The Deflagration Stage of Chandrasekhar Mass Models for Type Ia Supernovae. I. Early Evolution. *ApJ*, 782:11, February 2014. doi: 10.1088/0004-637X/782/1/11.