IMPROVED METHODS FOR GENERAL RELATIVISTIC RADIATION HYDRODYNAMICS AND THEIR IMPACT ON SIMULATIONS OF NEUTRON STAR MERGERS AND CORE-COLLAPSE SUPERNOVAE

By

Steven Anthony Fromm

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Physics — Doctor of Philosophy
Computational Mathematics, Science and Engineering — Dual Major

2024

ABSTRACT

Neutron star mergers and core-collapse supernovae are some of the most energetic events in the universe, reaching conditions not attainable in terrestrial laboratories. The study of these high energy-density astrophysical events relies on detailed multi-physics multi-scale modeling, ranging from nuclear and neutrino interactions to the large-scale dynamics governed by general relativity. Simulations prove useful in exploring these models, but they are sensitive to the physical approximations and numerical methods used to build them, requiring a balance to be struck between higher computational cost and increasingly detailed physical models. Choices made for the treatment of the neutrinos and the inclusion of general relativistic effects greatly impact the dynamics of these systems evolve, and impact the nucleosynthesis that occurs during these events. The `Flash-X` multi-physics code provides an ideal framework for creating the large-scale simulations necessary for studying both core-collapse supernovae and neutron star mergers. This dissertation will detail extending the capabilities in `Flash-X` with the addition of fully general relativistic solvers for neutrino radiation transport, hydrodynamics, a dynamic spacetime, the supporting infrastructure necessary for coupling them all together, and utilities to facilitate development of these solvers. A multi-group two-moment neutrino radiation transport solver makes use of a novel frequency discretization to improve computational efficiency. A high-order finite-difference scheme is applied to the hydrodynamics. A custom-built code-generator aids in the development of the dynamic spacetime solvers. A new method-of-lines time-discretization in `Flash-X` provides increased numerical stability and flexibility in choosing time-integration schemes appropriate for both the new and existing solvers. A full suite of rigorous tests validate these capabilities. Continuing work towards the coupled multi-physics multi-scale simulations necessary for neutron star mergers and core-collapse supernovae will be presented.

ACKNOWLEDGMENTS

Coming soon!

TABLE OF CONTENTS

Chapter 1

Introduction

Neutron star mergers (NSMs) are some of the most cataclysmic and energetic events in the universe. Neutron stars are highly-compact objects supported by degenerate neutron pressure, with masses $\sim M_\odot$ (the mass of the sun) and radii $\sim 10^6$ cm (Misner et al., 2017). Neutron stars are one possible remnant formed by the collapse of massive stars resulting in supernovae, as first proposed in Baade & Zwicky (1934) and later confirmed in Hewish et al. (1968) by the discovery of the pulsar, or magnetized rapidly spinning neutron star, at the center of the Crab Nebula, the remnant of a supernova observed in 1054 A.D. (Duyvendak, 1942; Mayall & Oort, 1942). Binary systems of neutron stars will radiate gravitational energy, leading to decaying orbits as first observed in Taylor et al. (1979). The inspiral and eventual merger of a binary neutron star system produces a violent collision that forms a remnant compact object; these type of mergers were first confirmed by the gravitational wave event GW170817 observed by the Advanced LIGO and Virgo observatories (Abbott et al., 2017).

Neutron stars, their mergers, and the core-collapse supernovae (CCSNe) that form them reach conditions not attainable in terrestrial laboratories, capable of reaching densities of $\sim 10^{14}$ g cm^{-3} and temperatures of $\sim$100 MeV (Perego et al., 2019). These extreme conditions provide ideal sites to study nuclear structure and reactions, such as constraining nuclear equations of state, and determining how and where heavier elements are formed through r-process nucleosynthesis. While information can be gleaned from observational evidence, numerical simulations based on detailed theoretical models are aptly suited for these studies and interpreting observations. The requisite physical details in these models span disparate physical processes and time- and length-scales, encompassing everything

from nuclear reactions to the large-scale dynamics governed by Einstein's theory of general relativity.

Numerical simulations for events involving compact objects become increasingly computationally expensive with an increasing level of physical detail and the use of more sophisticated numerical methods. Limited by available computational resources, approximations must be made in order to make the necessary calculations in these simulations practical to perform. Achieving high-fidelity in these simulations requires balancing these concerns, often requiring the development or application of new methods to more efficiently include higher levels of detail. This can be incredibly challenging for large multi-physics multi-scale simulations of events like neutron star mergers; every effort must be made to apply, improve, and create the tools necessary to accurately study these events.

This work will make use of the common notations used in numerical relativity. Additionally, a system of geometrized units will simplify the presentation of the mathematical formalisms presented in the following chapters. Please refer to appendix A for a description of these conventions.

1.1 Neutron Stars Mergers and Nucleosynthesis

Neutron stars and their mergers serve as astrophysical laboratories for nuclear physics. As neutron stars have densities similar to those of nuclei, their structures can be used to constrain nuclear equations of state for dense matter (Greif et al., 2020). The hot, dense matter ejected during neutron star mergers has been shown to produce r-process elements (Kasen et al., 2017; Hotokezaka et al., 2018; Rosswog et al., 2018). It is absolutely crucial to ensure models of neutron stars and their mergers include sufficient detail to be able to study these

and relate back to the observational data.

Relativistic stars in static equilibrium can be described by the Tolman-Oppenheimer-Volkoff (TOV) equations derived from Einstein's equations of general relativity and the assumption of a perfect fluid (Misner et al., 2017). These equations describe the structure of the star through its enclosed mass, metric potential, and pressure; an equation of state relates the pressure and the fluid state (see appendix B for more on the TOV equations). Generating a series of equilibrium TOV solutions using a specific equation of state produces a range of mass-radius relations for neutron stars. Comparison of these masses and radii to observed and theoretical constraints can provide a measure of whether the chosen equation of state is capable of reproducing these values.

A sample calculation of the mass-radius curves for different nuclear equations of state are shown in fig. 1.1. Five tabular nuclear equations of state[1] are compared: LS220 (Lattimer & Swesty, 1991), DD2 (Hempel et al., 2012), BHB$\Lambda\phi$ (Banik et al., 2014), and SHFo/SFHx (Steiner et al., 2013a). The most recent constraint on the maximum mass for a non-rotating neutron star of $2.25^{+0.08}_{-0.07}$ found by Fan et al. (2024) is shown for comparison. Only one equation of state, DD2, reaches the maximum mass constraint, but all equations of state used in this analysis are capable of producing the typically-assumed neutron star mass of $1.4M_{\odot}$ (Steiner et al., 2013b; Greif et al., 2020; Sotani et al., 2022).

A further example of the importance of neutron stars in nuclear physics is the r-process nucleosynthesis that occurs in the material ejected during a neutron star merger. The neutron-rich outflow provides ideal conditions that allow for rapid neutron-capture rates that exceed β-decay rates to build quickly build up heavy nuclei (Lattimer & Schramm,

[1]These tables are available for download from stellarcollapse.org and were originally produced for use in O'Connor & Ott (2010). This collection has been updated and added to since the original publication, including the tables used here for the TOV comparisons.

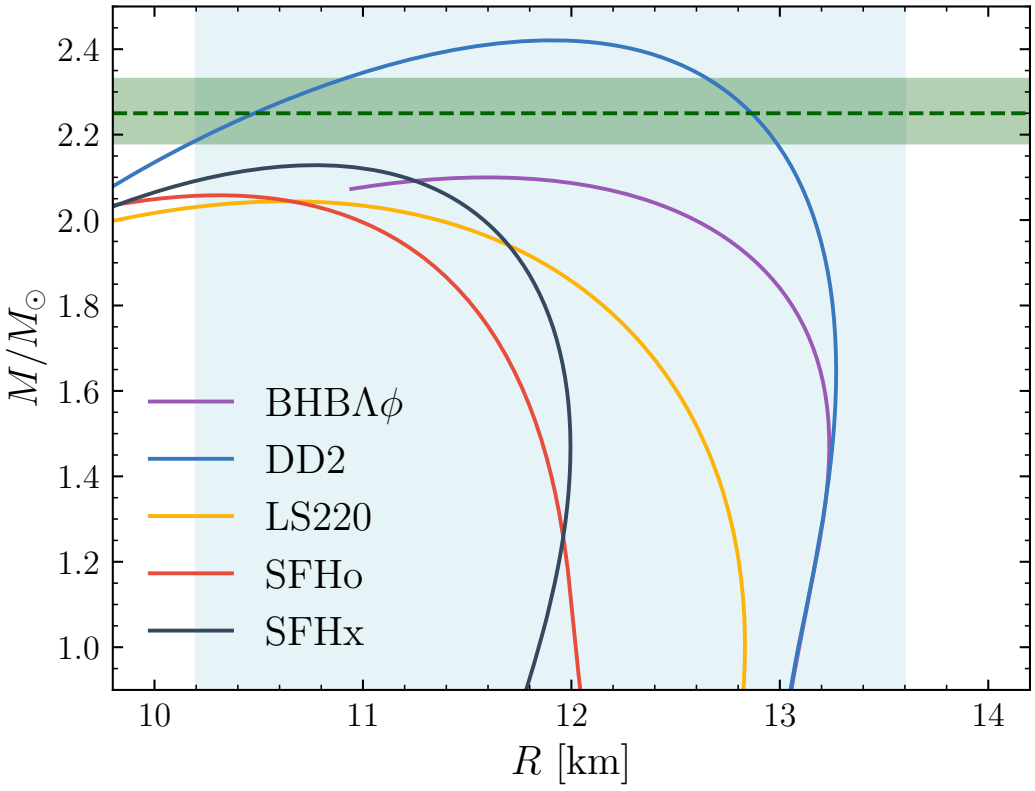

Figure 1.1: Mass-radius relation for non-rotating neutron stars. Results from using various nuclear equations of state are compared. Maximum non-rotating neutron star mass $2.25^{+0.08}_{-0.07} M_\odot$ from Fan et al. (2024) (green dashed line) and the allowed radius of a $1.4 M_\odot$ neutron star 10.2 km $< R <$ 13.6 km from Greif et al. (2020) (blue shaded region) are shown for comparison.

1974; Symbalisty & Schramm, 1982; Eichler et al., 1989; Freiburghaus et al., 1999; Goriely et al., 2005). These predictions were finally confirmed in the observed kilonova afterglow of the GW170817 neutron star merger (Kasen et al., 2017; Hotokezaka et al., 2018; Rosswog et al., 2018).

Detailed numerical simulations of neutron star mergers over a range of initial conditions and input physics are necessary for characterizing the ejected material and predicting the nucleosynthesis yields. Incredibly neutron-rich material is ejected during mergers on dynamical timescales of a few milliseconds due to tidal effects and shock heating of the collision, and on longer secular timescales via mechanisms such as neutrino-driven winds (Radice et al.,

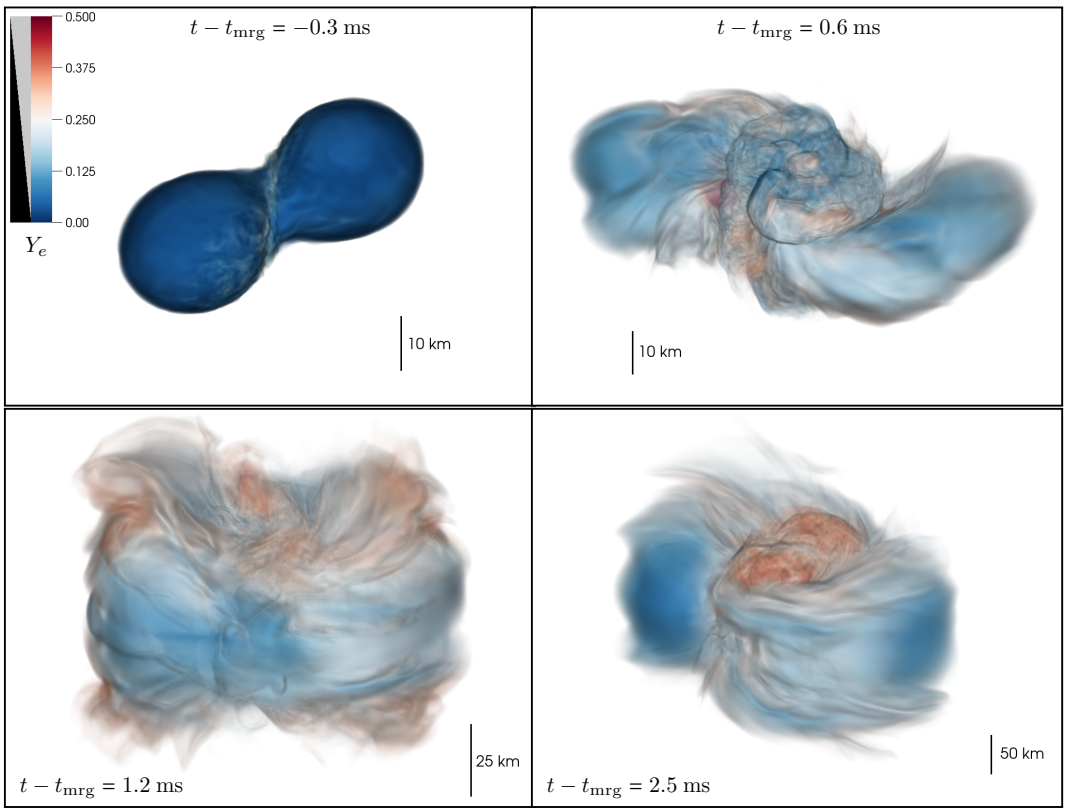

Figure 1.2: Volume rendering for a simulation of an equal-mass ($1.35M_\odot$) binary neutron star merger using the SFHo equation of state. Results display the electron fraction of the material in the system from just before the merger and until 2.5 ms after. Figure from Radice et al. (2018).

2018). Figure 1.2 shows an example of the earlier so-called dynamical ejecta in the first few milliseconds after the merger of an equal-mass ($1.35M_\odot$) binary system from one of our simulations in Radice et al. (2018). Initially, very neutron-rich material, characterized by low values of the electron fraction Y_e, is ejected in the tidal plane, followed by the more isotropic shock-heated material. For this particular simulation, the ejected material remained fairly neutron rich with an average value of $\langle Y_e \rangle = 0.22$. The resulting prediction for the yields of r-process nucleosynthesis are shown in fig. 1.3, which distinctly capture the production of the expected peaks near $A \sim 130$ and $A \sim 195$ when compared to the measured solar abundances given in Anders & Grevesse (1989); Arlandini et al. (1999).

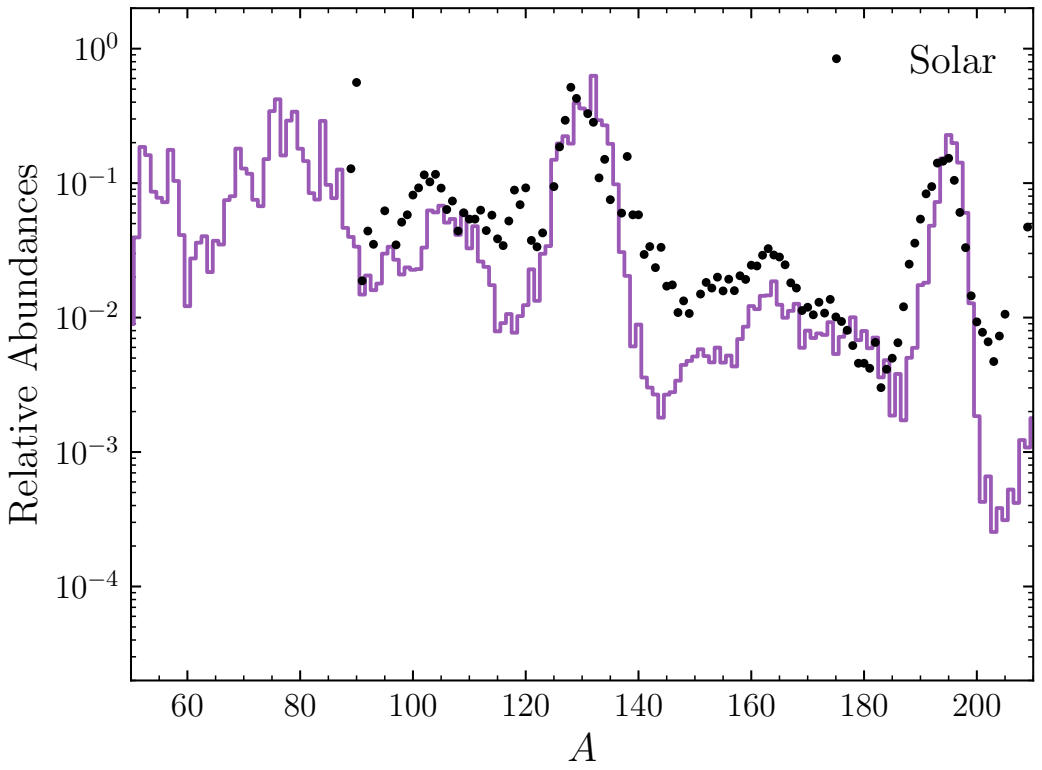

Figure 1.3: Predicted nucleosynthesis equal-mass ($1.35M_\odot$) binary neutron star merger using the SFHo equation of state. Solar abundances are included for reference. Based on data from Radice et al. (2018).

1.2 Methods for Simulating NSMs and CCSNe

Simulations of neutron star mergers and core-collapse supernovae pose significant computational challenges. Accurately modeling the radiative transport of neutrinos, the hydrodynamics of the fluid, and the evolution of a dynamic spacetime rely on detailed numerical calculations over many degrees of freedom. Sufficient resolution for these degrees-of-freedom, such as time, space, and momentum, often requires a large number of computing resources and lengthy periods of time spent to run these simulations. For more practical use of the available resources and time, large-scale simulations typically employ approximations to reduce the degrees-of-freedom; these approximations lead to a broad array numerical methods

for use in these problems.

This section will survey some of the more commonly used numerical methods that are applied in simulations of neutron star mergers and core-collapse supernovae. To serve as a comparison to the methods described in later chapters, this survey will focus on methods for general relativistic radiation transport, hydrodynamics, and spacetime evolution.

1.2.1 Neutrino Radiation Transport

Neutrinos are effectively massless particles, and typically treated as radiation in a similar manner as photons. Neutrino radiation interacting with matter significantly impacts energy transport (Burrows et al., 2006) and the composition of the material (Foucart, 2023) at the density and temperature scales present in core-collapse supernovae and neutron star mergers. Each species of neutrino, i.e., $\nu_e, \bar{\nu}_e, \nu_\mu, \bar{\nu}_\mu, \nu_\tau, \bar{\nu}_\tau$, can be characterized by a distribution function $f(t, \vec{x}, \vec{p})$ at a time t for a location $\vec{x}$ with momentum $\vec{p}$. These seven degrees-of-freedom per neutrino species can be computationally challenging and expensive to resolve, leading to varying levels of approximation for including neutrinos and their interactions with matter in numerical simulations.

1.2.1.1 Neutrino Leakage

One of the simplest approaches to including neutrinos in these simulations is by a method referred to as neutrino leakage. In this method, neutrinos are not directly evolved, but rather their interactions with the fluid are treated parametrically and used to directly alter the energy and composition of the fluid. These type of methods were originally applied to simulations of CCSNe in van Riper & Lattimer (1981) and have seen widespread use of the years, such as the simulations performed in O'Connor & Ott (2010); Perego et al. (2016).

Ruffert et al. (1996) presented one of the first applications of a neutrino leakage scheme to neutron star merger simulations, and this method has seen continued use and success in works such as Rosswog & Liebendörfer (2003); Galeazzi et al. (2013); Radice et al. (2016, 2018).

In a neutrino leakage scheme, the mean free paths for neutrinos are calculated based on the thermodynamic state of the fluid. These mean free paths determine the optical depth of neutrinos within dense regions of matter, and are used to determine the emission rates from these regions. Typical interactions for these calculations include charged-current β-processes, e.g. electron (e^-) and positron (e^+) capture on neutrons (n) and protons (p), scattering processes, e.g., an arbitrary neutrino species ν_i scattering off of nucleons, and pair-production of neutrinos, e.g., by electron-positron annihilation and plasmon (γ) decay (Ruffert et al., 1996); see table 1.1 for examples. The emission rates are then used to determine the lepton number and energy losses, and directly update the energy and composition of the fluid. As the name implies, a leakage scheme only considers the loss of neutrinos from the system, and there is no mechanism for re-absorbing the emitted neutrinos. Generally, these methods work in terms of the energy- or number-averaged rates, but energy-dependent (monochromatic) schemes are also possible, such as the one presented in Perego et al. (2016).

Charged-Current	Scattering	Pair-Production
$e^- + p \leftrightharpoons \nu_e + n$	$\nu_i + n \leftrightharpoons \nu_i + n$	$e^- + e^+ \leftrightharpoons \nu_e + \bar{\nu}_e$
$e^+ + n \leftrightharpoons \bar{\nu}_e + p$	$\nu_i + p \leftrightharpoons \nu_i + p$	$\gamma \leftrightharpoons \nu_e + \bar{\nu}_e$

Table 1.1: Example neutrino-matter interactions

1.2.1.2 Moment Expansions

Another approximate method for including and evolving neutrinos in neutron star merger and core-collapse supernovae simulations is performing a moment expansion over the angular degrees-of-freedom in the neutrino distribution function, effectively reducing the momentum degrees-of-freedom down to a single energy (or equivalently frequency in natural units). A procedure for generating a general relativistic moment expansion for photonic radiation was proposed in Thorne (1981), and has been adapted for use in neutrino radiation transport in many different works, e.g., Shibata et al. (2011); Cardall et al. (2013); O'Connor (2015); Foucart et al. (2015); Roberts et al. (2016); Radice et al. (2022).

Formally, the moment expansion is infinite, so the series is typically truncated for a small number of moments. However, the hierarchy of the expansion results in dependence on higher-rank moments than the truncation is limited to, requiring a closure relating the higher- to lower-rank moments. Different truncations result in schemes suitable to different types of conditions and problems. The first few moments in the frame of the fluid correspond to physically-meaningful quantities, such as the energy, momentum, and pressure for the zeroth-, first-, and second-rank moments, respectively. Neutrino-matter interactions, such as those in table 1.1, are typically characterized by emissivities and absorption and scattering opacities. Similarly to leakage schemes, moment schemes can also be frequency-averaged or monochromatic.

Retention of the zeroth-moment leads to evolution schemes for the radiation energy and are closed specifying the radiation momentum (or flux) to the energy. In a flux-limited diffusion scheme as described in Pons et al. (2000), the radiation is assumed to be nearly isotropic in the fluid frame, and behaves diffusively, i.e., its flux is proportional to the gradient

of the energy over some diffusive timescale related to the opacity of the material trapping the radiation. Corrections for deviations away from the completely diffusive limit are then made by the addition of an artificial opacity, while a flux-limiter prevents diffusive behavior when the radiation approaches the free-streaming limit. In the M0 scheme, as described in Radice et al. (2016, 2018), free-streaming radiation is assumed to travel along null radial trajectories, such that the flux scales directly with the energy. This type of scheme will typically be matched with a separate treatment for the diffusive limit.

Retaining the the zeroth- and first-rank moments for the radiation energy and momentum lead to the widely used M1 scheme. In the M1 scheme, the radiation pressure is specified by interpolation between its diffusive and free-streaming limiting forms. Sharing a similar structure with common hydrodynamics formulations, many of the same numerical methods can be applied to the M1 scheme. The M1 scheme has been used extensively in simulations of core-collapse supernovae, e.g., in O'Connor (2015); Roberts et al. (2016); O'Connor & Couch (2018a,b), and mergers involving neutron stars, e.g., Foucart et al. (2015); Radice et al. (2022). The M1 scheme will be described further in chapter 3.

1.2.1.3 Boltzmann Equation Methods

The full evolution of the neutrino distribution function is described by the Boltzmann equation, which relates its temporal, spatial, and momentum changes to the interaction rates of the neutrinos with the fluid. Discretization of the momentum-dependence of the distribution function into discrete angles or rays for the basis of the discrete ordinates methods, or S_N. This method can be applied to directly solving the Boltzmann equation for the distribution function, as in Mezzacappa & Bruenn (1993); Mezzacappa & Messer (1999); Liebendörfer et al. (2005), or as a means of closing a moment scheme, e.g., by determining the radia-

tion pressure as is done in Asahina et al. (2020). Instead of the typical finite-difference approaches to solving the Boltzmann equation in the preceding examples, particle-based Monte Carlo methods provide an alternative route to evolving the distribution function (see for example Abdikamalov et al. (2012)). The increased physical detail and possible accuracy of solving the Boltzmann equation directly incur a higher computational cost than the more-approximate leakage and moment schemes, particularly when additional discretizations of the distribution momentum space by angular grids or representations of the distribution function by particles are coupled to the more-common spatial grid discretizations used in hydrodynamics and spacetime solvers.

1.2.2 Hydrodynamics

The fluids in simulations of neutron star mergers and core-collapse supernovae are most easily described by their rest-mass density and a stress-energy tensor describing their energy, momentum, and pressure. The standard approach to evolving these fluids takes the form of a system of conservation and balance laws for the time rate-of-change in the rest-mass, energy, and momentum densities. One of the earliest examples of numerical relativistic hydrodynamics was applied to gravitational collapse problems in May & White (1966); they derive a a system for the fluid equations of motions alongside mass and energy conservation of a similar form presented earlier in Misner & Sharp (1964). In Wilson (1972), an Eulerian form of the relativistic hydrodynamics were introduced that took a similar form as their Newtonian counterparts. The most common form of the general relativistic hydrodynamics equations used today was presented in(Banyuls et al., 1997) and is referred to as the Valencia formulation; this formalism will be covered in greater detail in chapter 4. This section will survey three commonly used numerical methods applied to this formalism: finite-difference,

finite-volume, and smoothed particle hydrodynamics.

1.2.2.1 Finite-Difference

Finite-difference methods have long been used in hydrodynamics simulations. These methods treat the fluid quantities as point-like values on a discrete computational fixed (Eulerian) or moving (Lagrangian) computational mesh, and use stencil-based approximations of the spatial and temporal derivatives. While generally robust and straightforward to implement, some features, such as shock and other discontinuities in the fluid, can prove tricky to resolve without additional numerically- or physically-motivated corrections, or the use of higher-order methods. A high-order finite-difference scheme will be described in chapter 4.

While finite-difference methods for relativistic hydrodynamics may not be as widely used as other methods, they have the advantage of being straightforward to implement with high-order schemes. In Zhang & MacFadyen (2006), high-order flux reconstruction coupled with adaptive mesh refinement demonstrate the accuracy and scalability of these methods. Finite-difference methods also can provide a robust and accurate treatment of relativistic magnetohydrodynamics, as shown in Del Zanna et al. (2007). Radice et al. (2014) demonstrates the accuracy of high-order finite-difference methods applied to neutron star mergers.

1.2.2.2 Finite-Volume

In contrast to finite-difference methods, finite-volume methods treat the fluid quantities as their averages over discrete cells in a computational mesh. This allows the use of flux-conservative methods to ensure that the flux between cells is exactly conserved, i.e., the flux of density, energy, or momentum leaving one cell is the same as the corresponding flux entering an adjacent cell. The methods used to guarantee flux conservation between cells,

which may have a discontinuity at their interface between the adjacent cell-averages, are capable of capturing shocks and similar features in the fluid.

Finite-volume methods are commonly used in general relativistic hydrodynamics. Examples of finite-volume methods used in core-collapse simulations include O'Connor & Ott (2010); Mösta et al. (2014); O'Connor (2015); Roberts et al. (2016); Pajkos (2022). Rezzolla et al. (2010); Radice et al. (2016, 2018) provide examples of finite-volume methods applied to neutron star mergers.

1.2.2.3 Smoothed Particle Hydrodynamics

The previous finite-difference and finite-volume methods can present challenges when trying to achieve sufficient resolution of their Eulerian or Lagrangian meshes. Smoothed particle hydrodynamics (SPH) provides an alternative mesh-free Lagrangian discretization that can automatically adapt the resolution to where (and only where) it is required. The fluid is discretized into (typically) equal-mass particles, from which fluid properties and derivatives are calculated over a local neighborhood of particles. By adapting the size and weights of these calculations, referred to as smoothing kernels, proportionally to the density, each calculation can be made over an equal number of particles. Coupled with a tree-based representation of the particles, this allows for consistent computational costs throughout the domain. Additionally, SPH methods can directly handle regions in vacuum (by not performing calculations there), in contrast to mesh-based methods that typically must impose an artificial marginal-density atmosphere to avoid numerical issues. Examples of SPH methods used in general relativistic hydrodynamics include Rosswog & Davies (2002); Rosswog (2015); Liptai & Price (2019).

1.2.3 General Relativity

The evolution of dynamic spacetimes are governed by Einstein's equations of general relativity relating the curvature of spacetime to its distribution of matter. Accurately describing a dynamic spacetime is essential when dealing with compact objects such as neutron stars and black holes. Numerically, these prove challenging to solve, with the whole field of numerical relativity dedicated to tackling these challenges Baumgarte & Shapiro (2010). Representing the spacetime in terms of the time and spatial discretizations suitable for numerical simulations can lead to violations of Einstein's equations due to simplifying assumptions and numerical errors. This section will survey some of the more common approaches used to solve Einstein's equations.

1.2.3.1 Unconstrained

One of the earliest formulations of Einstein's equations suitable for numerical simulations is referred to as the Arnowitt, Deser, and Misner (ADM) formalism proposed in Arnowitt et al. (1962). This formulation utilizes a 3+1 decomposition of the spacetime (see appendix C for more on this decomposition), and evolves the spatial metric and extrinsic curvature of the spacelike hypersurfaces of the decomposition. This formalism assumes that the Hamiltonian and momentum constraints of the Einstein equations hold exactly. Unfortunately, outside of spherically symmetric or axisymmetric spacetimes, this can lead to numerical instabilities due to numerical errors causing the violation of the constraints (Baumgarte & Shapiro, 2010). Other spherically-symmetric evolution schemes were introduced in Misner & Sharp (1964); Hernandez & Misner (1966).

1.2.3.2 Constraint solvers

Alternatively, the the constraint equations can be solved to determine the metric and curvature quantities. One popular method in early neutron star merger simulations is the conformally flat condition (CFC) used in Wilson & Mathews (1995). Other examples include the waveless approximations in Isenberg (2008), an extended CFC in Cordero-Carrión et al. (2009), and a fully-constrained formulation in Bonazzola et al. (2004). All of these methods rely on solving elliptic equations, including vector Laplacian equations. For three-dimensional simulations, these type of elliptic equations can be computationally expensive to solve, and can present challenges in ordering sets of calculations for the spacetime, matter, and radiation.

1.2.3.3 Constraint Damping

One of the more common approaches seeks to use the hyperbolic form of the ADM equations while minimizing constraint violation errors through damping or propagation out of the system. One of the first methods was proposed by Baumgarte & Shapiro (1998) and Shibata & Nakamura (1995), referred to as BSSN after the authors, and makes a conformal decomposition of the ADM equations. The family of Z4 methods makes use of a modification of Einstein's equations that includes terms for advecting and damping constraint violations Gundlach et al. (2005), such that the system of equations reduces to Einstein's equations when the constraints hold exactly. Two commonly used formulations of the Z4 formalism include the conformal Z4c formalism used in Bernuzzi & Hilditch (2010); Ruiz et al. (2011); Cao & Hilditch (2012); Hilditch et al. (2013); Daszuta et al. (2021), and the conformal and covariant CCZ4 formalism used in Alic et al. (2012); Dumbser et al. (2018); Clough et al.

(2015); Radia et al. (2022). The Z4 formalisms share a similar structure as the generalized harmonics formulation, an alternative to standard 3+1 formalisms, that has been used in the simulation of binary black hole mergers performed in Pretorius (2005).

Chapter 2

Sensitivity to Neutrino Treatment

Simulations of neutron star mergers (NSMs) and core-collapse supernovae (CCSNe) are sensitive to the physical approximations made and numerical methods used to model these events. In particular, the treatment of neutrinos can influence the evolution and outcomes of these types of simulations. In the extreme conditions present in NSMs and CCSNe, neutrinos readily interact with the matter, and can play a significant role in transporting energy. In the presence of the compact objects and relativistic velocities, neglecting relativistic effects can vastly impact the dynamics of these systems. As such, the approximations and methods used to include neutrinos in these type of simulations can greatly impact not only the dynamics, but also the composition of the matter.

Increased detail in the physical models and less-approximate numerical methods can more accurately account for the effects of neutrinos and general relativity, but typically incur significantly larger computational costs. In order to balance these concern, it is absolutely critical to assess how results change, or stay the same, when using different treatments for neutrinos and their interactions. This chapter will examine the sensitivities of NSM and CCSN simulations to these effects.

2.1 Neutron Star Mergers

The material ejected during NSMs can be strongly influenced by the presence of neutrinos. Neutrino winds from remnant's accretion disk can drive material out of the system, while the interactions of neutrinos with the ejected material, referred to as ejecta, can alter the

material's composition and impact the nucleosynthesis that occurs here (Radice et al., 2018). However, fully accounting for these neutrino effects in numerical simulations of NSMs proves to be difficult. Neutrinos can be both fully trapped by the high-density material in the remnant objects and free-streaming through less-dense material in the ejecta, requiring the neutrino treatment to handle both of these regimes as well as the transition between them (Foucart, 2023).

Approximating neutrinos in NSM simulations can lead to varying results for the properties of and the nucleosynthesis that occurs in the ejecta. This section will explore the effects of the neutrino treatment on the ejecta by comparing different approximations made in a common set of simulations. The data used for analysis in this section comes from a set of NSM simulations performed in Radice et al. (2018). The simulations use an unequal mass binary with initial neutron star masses $M_a = 1.4M_\odot$ and $M_b = 1.2M_\odot$. Two simulations of this binary separately use a leakage scheme and the M0 moment scheme for the neutrinos. Both simulations make use of the SFHo nuclear equation of state (Steiner et al., 2013a). These simulations are referred to as `SFHo_M140120_LK` and `SFHo_M140120_M0` for the leakage and M0 schemes, respectively, in table 2 of Radice et al. (2018). The properties of the ejecta are measured at an extraction radius of $300M_\odot$.

As a first comparison, fig. 2.1 shows the distribution of velocities within the ejecta. Overall, the M0 scheme produces higher maximum velocities ($v_{\max} = 0.91$ for M0, $v_{\max} = 0.79$ for leakage); however, only $\sim$0.025% of the ejecta in the M0 simulation exceeds the maximum velocity in the leakage simulation. The leakage scheme has an overall higher average velocity throughout the ejecta ($\langle v \rangle = 0.16$ for M0, $\langle v \rangle = 0.20$ for leakage). While both simulations have similar overall velocity distributions throughout the ejecta, changing only the neutrino treatment does have a small but noticeable impact.

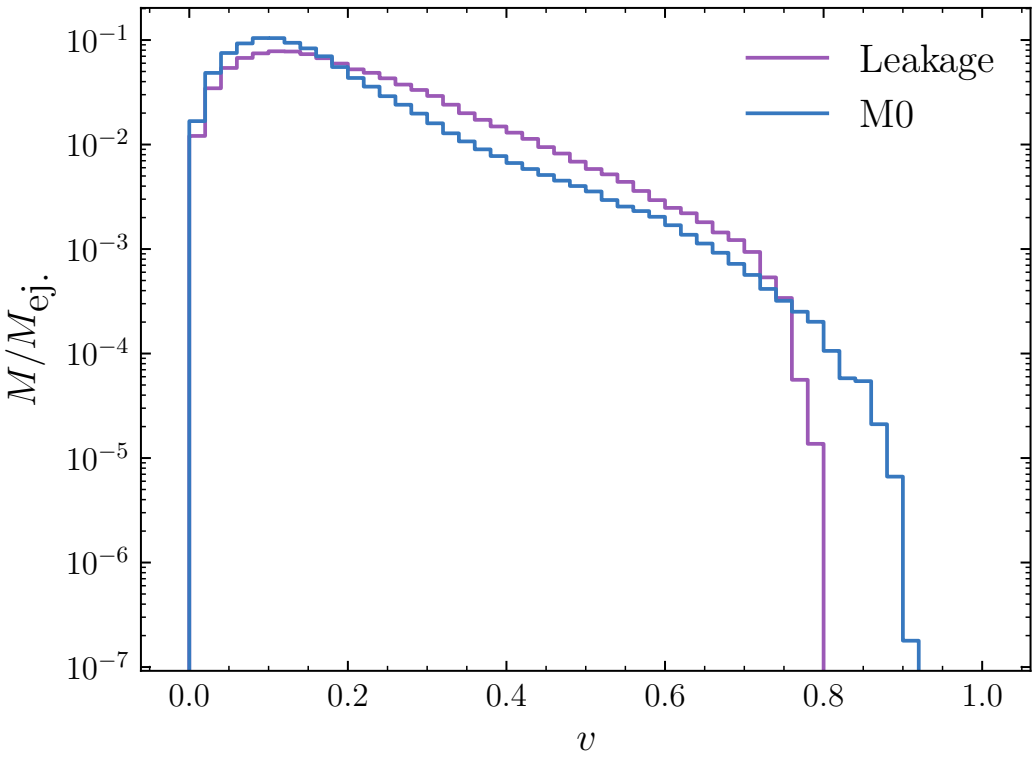

Figure 2.1: Impact of neutrino treatment on ejecta velocities. The the fractions of the total ejected mass $M_{\rm ej.}$ at different velocities are compared for the leakage (purple) and M0 (blue) schemes.

The next comparison examines the impact of the neutrino treatment on the composition of the ejecta, in particular neutron-richness of the material. This is most easily characterized by the electron fraction Y_e; the material is more neutron-rich for decreasing values of $Y_e < 0.5$ ($Y_e = 0.5$ indicates an equal number of neutrons and protons). Figure 2.2 shows a comparison of the distribution of mass for across a range of Y_e from the separate simulations using the leakage and M0 schemes. In contrast with the previous comparison of the ejecta velocities, the difference in the composition between the two schemes is more severe. The leakage scheme leaves a larger fraction of the ejected mass in a more neutron-rich state: $\sim$84% of the ejected mass has $Y_e \leq 0.2$, compared to $\sim$42% for the M0 scheme. The M0 scheme also produces higher $Y_e > 0.5$ values in a small fraction of the ejected mass. These results follow from one

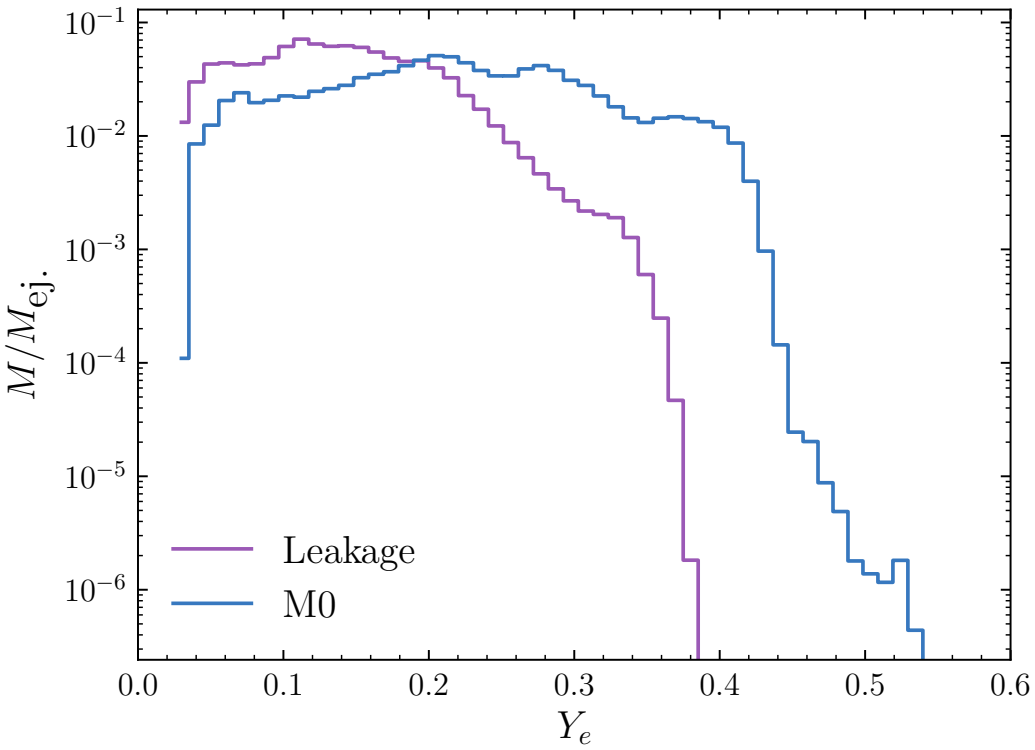

Figure 2.2: Impact of neutrino treatment on ejecta composition. The the fractions of the total ejected mass $M_{\rm ej.}$ at different values of the electron fraction Y_e are compared for the leakage (purple) and M0 (blue) schemes.

of the primary differences between the two neutrino schemes, the inclusion or omission of the re-absorption of emitted neutrinos, for the M0 and leakage schemes, respectively. Since there is no mechanism for the ejecta to interact with emitted neutrinos in the leakage scheme, a larger fraction of the ejected material is left in a neutron-rich state.

These scheme-dependent compositional differences in the ejecta can further be seen in the predicted r-process nucleosynthesis yields. Figure 2.3 compares the relative abundances of heavier nuclei produced in the ejecta in reference to the measured solar abundances in Anders & Grevesse (1989); Arlandini et al. (1999). These results are calculated during post-processing of the simulation data with the nuclear reaction network `SkyNet` (Lippuner & Roberts, 2017). The thermodynamic trajectories necessary for integrating the network

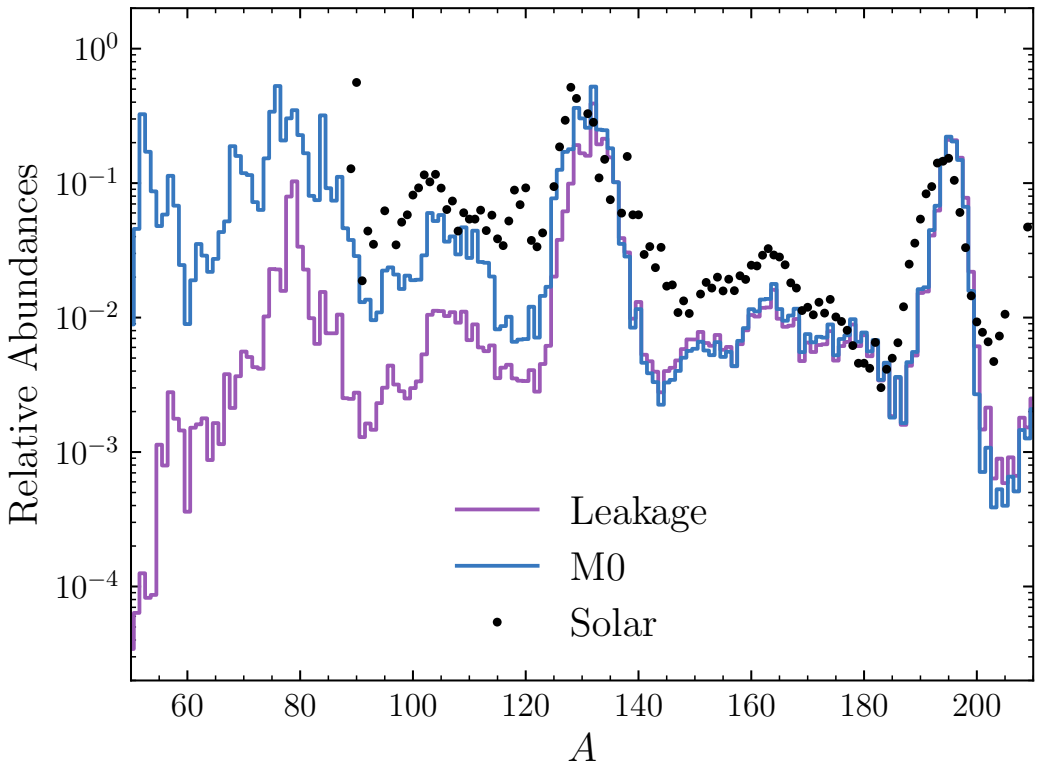

Figure 2.3: Impact of neutrino treatment on nucleosynthesis in the ejecta. Predicted abundances of r-process nuclei produced in the ejecta are compared for the leakage (purple) and M0 (blue) schemes. Observed solar abundances are included for reference.

assume homologous expansion of the ejecta based on the thermodynamic state at the extraction radius. In Radice et al. (2018), we found this approximation works well and produces similar results when compared to nucleosynthesis calculations performed on the trajectories of a large number of Lagrangian tracer particles in the ejecta. The results for mass numbers $A > 130$ show similar predictions for both neutrino schemes, but for lower mass numbers the results become more disparate.

Figure 2.4 breaks down the contribution of different ranges of Y_e values to the predicted yields. These results make use of the actual thermodynamic trajectories as measured by Lagrangian tracer particles in the M0 simulation; tracer particle data is not available for the the leakage simulation. Material with higher values of Y_e, the less neutron-rich material,

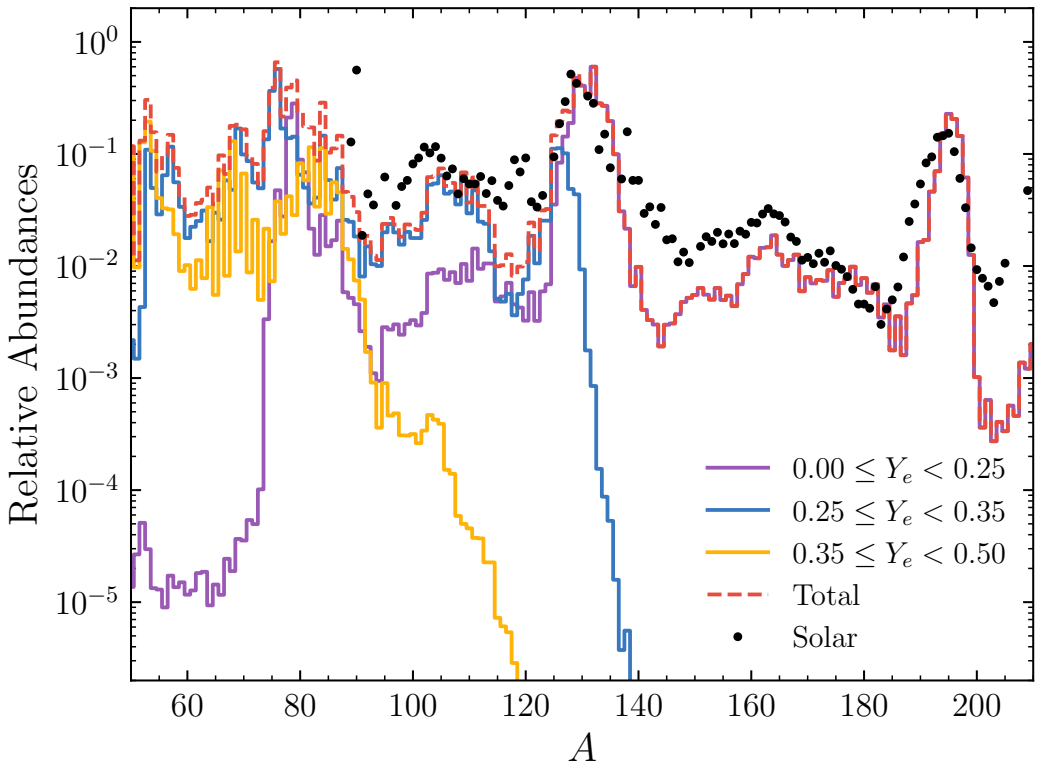

Figure 2.4: Contributions of different ranges of Y_e values to r-process nucleosynthesis in the ejecta. Predicted yields utilize the actual thermodynamic trajectories measured by Lagrangian tracer particles in the M0 simulation. Solar abundances are added for reference.

contribute almost exclusively to the lighter r-process nuclei. When viewed in comparison with the Y_e distributions in fig. 2.2 and the yields in fig. 2.3, it follows that the overall less neutron-rich ejecta in the M0 simulation produces a larger abundance of these lighter r-process nuclei than the more neutron-rich material in ejecta in the leakage simulation.

2.2 Core-Collapse Supernovae

Core-collapse supernovae simulations also provide examples of sensitivities to neutrino treatment. While simulations of NSMs require a fully general relativistic treatment of the neutrinos and hydrodynamics due to highly dynamic spacetimes, CCSNe simulations can typically approximate these as gravitational effects through either Newtonian or general relativistic

effective potentials (Rampp & Janka, 2002; O'Connor & Couch, 2018a). However, the presence of relativistic velocities, i.e, velocities that are significant fractions of the speed of light, can still impact how neutrinos evolve and interact with matter.

Neutrinos and their interactions with matter are most easily described in the co-moving frame of the fluid. Large-scale three-dimensional simulations typically operate with a fixed Eulerian computational frame for evolving neutrinos and the fluid. At relativistic velocities, the differences in how neutrino properties, such as their energy and momentum, are measured in each frame cannot be ignored, and the transformation of these quantities to and from each frame become velocity-dependent. Additionally, when resolving a spectrum of neutrino frequencies, acceleration of the fluid, as well as gravitational effects, cause redshifting effects that impact the number and energy densities of neutrinos at different frequencies.

The inclusion or omission of a velocity-dependent neutrino treatment can produce different results for the evolution of CCSNe. O'Connor & Couch (2018b) compare two-moment neutrino radiation transport methods with and without velocity-dependence in CCSNe simulations; these results are reproduced here for reference in fig. 2.5. Their simulations all start from same progenitor model: the $20M_{\odot}$ produced by the Modules for Experiments in Stellar Astrophysics (MESA) code and presented in Farmer et al. (2016). These models were then evolved through collapse and up until 15 ms post-bounce with the `GR1D` one-dimensional general relativistic hydrodynamics code (O'Connor, 2015), and subsequently evolved in full three-dimensional simulations performed with `FLASH` (Fryxell et al., 2000; Dubey et al., 2009). Two sets of simulations were ran, ones with a fully velocity-dependent neutrino treatment labeled as `mesa20_v_LR` and `mesa20_v_LR_oct` in fig. 2.5, and a separate neutrino treatment where all velocity-dependent terms were set to zero and the effects of velocity-dependent redshifting were neglected, `mesa20_LR` and `mesa20_LR_oct` in fig. 2.5; `LR` and `oct` refers to

low-resolution and octant-symmetry, respectively.

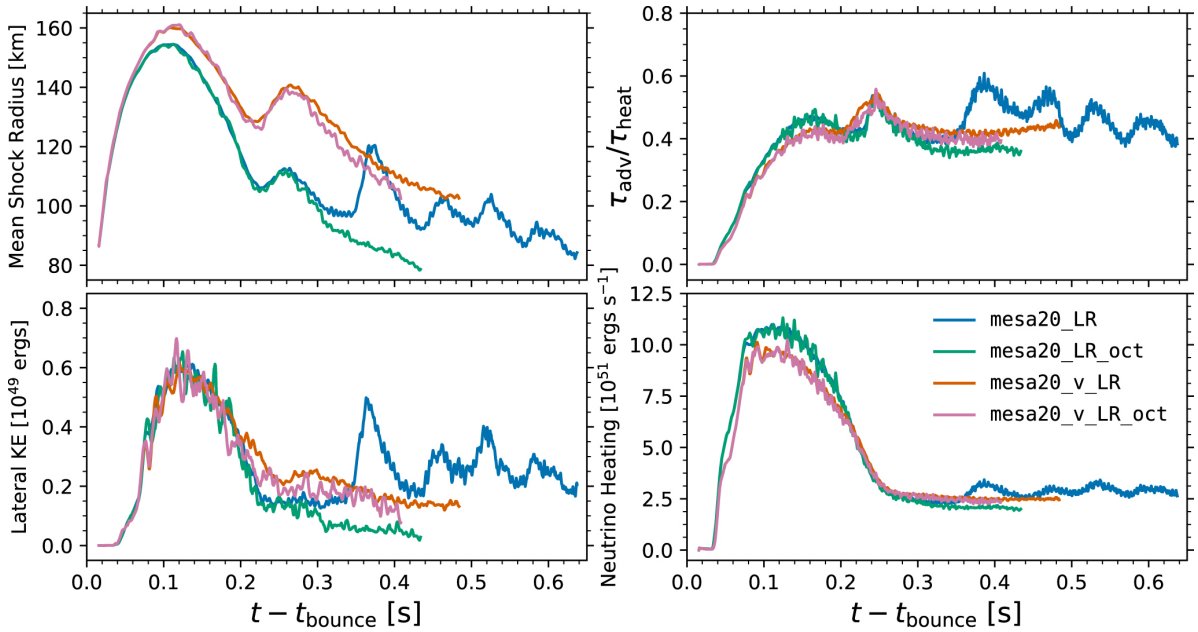

Figure 2.5: Comparison of velocity dependence in neutrino treatment for CCSNe. Figure from O'Connor & Couch (2018b). Individual plots show the mean shock radius (top-left), lateral kinetic energy (bottom-left), ratio of advection and neutrino heating timescales (top-right), and the neutrino heating rate (bottom-right) at times after bounce. Simulations used either a velocity-dependent (red and magenta lines) or non-velocity-dependent (blue and green lines) neutrino treatment.

The results for these simulations performed in O'Connor & Couch (2018b) (displayed here in fig. 2.5) highlight the effects of the velocity-dependence in the neutrino treatment on the dynamics of the evolving CCSN. Most noticeably in the top-left plot, the velocity-dependent neutrino treatment produces a faster- and farther-moving shock that subsequently recedes more slowly. They also note that some of the other differences seen, such as lower neutrino heating rates for the velocity-dependent treatment in the lower-right plot, result from differences in how energy exchange is defined in each treatment: the velocity-dependent treatment accounts for the rate of total energy between neutrinos and the fluid, while the treatment with no velocity-dependence accounts for the rate of internal energy exchange.

Chapter 3

General Relativistic Radiation Transport

Neutrinos play a crucial role in NSMs and CCNSe, influencing the evolution of the composition of the matter and providing a mechanism for energy transport. In these highly dynamic systems, neutrinos cannot be assumed to be in equilibrium with the surrounding matter, which necessitates describing the full evolution of each species distribution and their interactions. Along a trajectory through spacetime characterized by an affine parameter λ, the evolution of the neutrino distribution $f \equiv f(x^a, p^a)$ can be described by the general-relativistic Boltzmann equation (Thorne, 1981)

$$\frac{\mathrm{d}f}{\mathrm{d}\lambda} = \frac{\mathrm{d}x^a}{\mathrm{d}\lambda}\frac{\partial f}{\partial x^a} + \frac{\mathrm{d}p^i}{\mathrm{d}\lambda}\frac{\partial f}{\partial p^i} = \left[\frac{\mathrm{d}f}{\mathrm{d}t}\right]_{\text{coll.}}, \tag{3.1}$$

with spacetime coordinates x^a and momentum coordinates p^i (where only the spatial components are necessary since $p_a p^a = 0 \implies p_0 p^0 = -p_i p^i$). The right-hand side of Eq. (3.1) is referred to as the collision integral, and represents the temporal rate-of-change in the neutrino distribution function due to collisions, i.e. interactions between the neutrinos and the matter.

To better highlight the momentum-dependence in Eq. (3.1), the affine parameter can be defined as

$$\frac{\mathrm{d}}{\mathrm{d}\lambda} = \frac{p^b}{(-u_c p^c)}\frac{\partial}{\partial x^b}, \tag{3.2}$$

where $-u_c p^c = \nu$ are specific values of the neutrino frequencies (or equivalently energies

in natural units) ν as measured by a co-moving observer in the frame of the fluid with a four-velocity u^c. The change in the spacetime coordinates x^a along this affine trajectory immediately follows from Eq. (3.2) as

$$\frac{\mathrm{d}x^a}{\mathrm{d}\lambda} = \frac{p^a}{(-u_c p^c)}. \tag{3.3}$$

A similar change in the momentum coordinates p^a can be found by comparison to the geodesic described by

$$\frac{\mathrm{d}^2 x^a}{\mathrm{d}\lambda^2} + \Gamma^a{}_{bc} \frac{\mathrm{d}x^b}{\mathrm{d}\lambda} \frac{\mathrm{d}x^c}{\mathrm{d}\lambda} = 0, \tag{3.4}$$

where $\Gamma^a{}_{bc}$ are the connection coefficients of the covariant derivative operator (more commonly referred to as the Christoffel symbols when working with a coordinate basis). Applying Eq. (3.2) to the momentum coordinates p^a as defined in Eq. (3.3) then shows that

$$\frac{\mathrm{d}p^a}{\mathrm{d}\lambda} = (-u_c p^c) \frac{\mathrm{d}^2 x^a}{\mathrm{d}\lambda^2} = -\frac{1}{(-u_c p^c)} \Gamma^a{}_{bc} p^b p^c. \tag{3.5}$$

Making use of Eq. (3.3) and Eq. (3.5) in Eq. (3.1), the evolution of the distribution function becomes

$$p^a \frac{\partial f}{\partial x^a} - \Gamma^i{}_{ab} p^a p^b \frac{\partial f}{\partial p^i} = (-u_c p^c) \left[\frac{\mathrm{d}f}{\mathrm{d}t}\right]_{\text{coll.}}. \tag{3.6}$$

With seven degrees of freedom (1 time + 3 space + 3 momentum) for each neutrino species, tackling Eq. (3.6) presents a significant challenge. As discussed earlier in chapter 1, methods capable of directly solving Eq. (3.6), e.g. Monte-Carlo particle methods, are

computationally intensive and can be difficult to evolve consistently with the fluid and the spacetime. One effective way of reducing these degrees of freedom to something more practical is performing a moment expansion of the neutrino distribution function (Thorne, 1981; Shibata et al., 2011). The remainder of this chapter will describe the formulation of and the numerical methods used for a two-moment evolution scheme. The fully general relativistic implementation of this scheme in `Flash-X` will be presented later in chapter 6.

3.1 Moment Formalism

3.1.1 Unprojected Moment Expansion

For specific values of neutrino frequencies ν, the neutrino distribution function can be approximated by a monochromatic moment expansion by integrating over the four-momenta degrees of freedom. As applied to photons in Thorne (1981) and later to neutrinos in Shibata et al. (2011), this monochromatic expansion for any arbitrary-rank unprojected moment of the distribution function takes the form

$$M^{A_k}_{(\nu)} \equiv M^{A_k}_{(\nu)}\left(x^b\right) = \int \mathrm{d}V'_p \, \frac{\delta\left(\nu + u_c p'^c\right)}{\left(-u_c p'^c\right)^{k-2}} \left(\prod_{a_k \in A_k} p'^{a_k} \right) f\left(x^b, p'^a\right), \tag{3.7}$$

where $\mathrm{d}V_p$ is the invariant volume element of the momentum coordinates p^a, and $A_k = \{a_1, \ldots, a_k\}$ is the set of indices for the k-th rank moment. This expansion holds only for the monochromatic case, i.e. only for the specific values of the frequency ν chosen for each moment. For this particular formalism, ν will always be the frequency measured by co-moving observer in the fluid frame, as this is the most practical frame to describe the interaction of neutrinos with the matter. It is important to note that Eq. (3.7) forms a fully-

symmetric tensor for any arbitrary rank that the expansion is specialized to. At times, it will be necessary to measure the total contribution from all neutrino frequencies, and Eq. (3.7) can be integrated over ν (separately for each species)

$$M^{A_k} = \int_0^\infty \mathrm{d}\nu \, M^{A_k}_{(\nu)}. \tag{3.8}$$

The evolution of the neutrino distribution function can now be described in terms of this moment expansion. One approach involves applying the moment expansion directly to the Boltzmann equation (Cardall et al., 2013). A second approach proposed in Shibata et al. (2011) relates the covariant divergence of the moment expansion to the Boltzmann equation; this is the approach adopted for this work. By taking the covariant divergence of the moment expansion, the evolution of any arbitrary $(k+1)$-th rank moment takes the form (see appendix D for the derivation)

$$\boldsymbol{\nabla}_b M^{A_k b}_{(\nu)} - \frac{\partial}{\partial \nu}\Big(\nu M^{A_k bc}_{(\nu)} \boldsymbol{\nabla}_b u_c\Big) - (k-1)\nu M^{A_k bc}_{(\nu)} \boldsymbol{\nabla}_b u_c = S^{A_k}_{(\nu)}, \tag{3.9}$$

where $\boldsymbol{\nabla}_b$ is the covariant derivative operator compatible with the spacetime metric g_{ab}. The right-hand side of Eq. (3.9) represents the moment expansion of the collision integral from the Boltzmann equation

$$S^{A_k}_{(\nu)} = \int \mathrm{d}V'_p \, \frac{\delta\big(\nu + u_c p'^c\big)}{\big(-u_c p'^c\big)^{k-2}} \left(\prod_{a_k \in A_k} p'^{a_k} \right) \left[\frac{\mathrm{d}f}{\mathrm{d}t}\right]_{\text{coll.}}. \tag{3.10}$$

The evolution of the frequency-integrated moments follows directly as

$$\boldsymbol{\nabla}_b M^{A_k b} - (k-1)\nu M^{A_k bc} \boldsymbol{\nabla}_b u_c = S^{A_k}, \tag{3.11}$$

where the frequency-derivative term vanishes under integration since its argument goes to zero at the endpoints $\nu = 0, \infty$.

While Eq. (3.9) and Eq. (3.11) permit evolving any arbitrary-rank moment of the distribution function, a choice of k must be made to further develop a moment evolution formulation. A practical choice is $k = 1$, leading to a set of evolution equations in terms of the second-rank moment expansion; this moment represents the neutrino stress-energy tensor. In this case, the monochromatic evolution equation takes the form

$$\boldsymbol{\nabla}_b M^{ab}_{(\nu)} - \frac{\partial}{\partial \nu}\Big(\nu M^{abc}_{(\nu)} \boldsymbol{\nabla}_b u_c\Big) = S^a_{(\nu)}, \tag{3.12}$$

and its frequency-integrated counterpart takes the even simpler form

$$\boldsymbol{\nabla}_b M^{ab} = S^a. \tag{3.13}$$

3.1.2 Fluid Frame Projections of the Moment Expansion

The next step in developing a moment formalism practical for numerical implementation involves decomposing the moment expansion into its irreducible parts. The projections of the moment expansion required for this decomposition will correspond to physically-relatable quantities measured by a co-moving observer in the frame the expansion is defined. In this case, these projections will most easily be made for the fluid frame, but since the moment-expansion and its evolution equation are covariant expressions, these projections can be directly related to the projections in any other frame of reference.

To project the moment expansion into the fluid frame, the neutrino four-momentum is decomposed into components parallel and orthogonal to the co-moving observer's four-

velocity, taking the form

$$p^a = (-u_c p^c)(u^a + \ell^a), \tag{3.14}$$

where ℓ^a is a unit normal vector orthogonal to the four-velocity u^a, i.e. $\ell^a \ell_a = 1$ and $u_a \ell^a = 0$. With this choice of four-momentum, the invariant volume element in Eq. (3.7) takes the form (Thorne, 1981)

$$\mathrm{d}V_p = (-u_c p^c)\, \mathrm{d}(-u_c p^c)\, \mathrm{d}\Omega\,, \tag{3.15}$$

where $\mathrm{d}\Omega$ is the differential solid-angle of the momentum-coordinates. With these, Eq. (3.7) reduces to

$$M^{A_k}_{(\nu)}\left(x^b\right) = \nu^3 \int \mathrm{d}\Omega \left[\prod_{a_k \in A_k} (u^{a_k} + \ell^{a_k})\right] f_{(\nu)}, \tag{3.16}$$

where $f_{(\nu)} \equiv f(x^b, \nu, \Omega)$.

For this form of the moment expansion, the first few projected moments are defined as

$$J_{(\nu)} = \nu^3 \int \mathrm{d}\Omega\, f_{(\nu)} \tag{3.17}$$

$$H^a_{(\nu)} = \nu^3 \int \mathrm{d}\Omega\, \ell^a f_{(\nu)} \tag{3.18}$$

$$L^{ab}_{(\nu)} = \nu^3 \int \mathrm{d}\Omega\, \ell^a \ell^b f_{(\nu)} \tag{3.19}$$

$$N^{abc}_{(\nu)} = \nu^3 \int \mathrm{d}\Omega\, \ell^a \ell^b \ell^c f_{(\nu)}. \tag{3.20}$$

The first three projected moments correspond to the radiation energy density, momentum

density, and pressure tensor associated with neutrinos of a monochromatic frequency ν. Specializing Eq. (3.16) to the $k = 2$ case, the second-rank moment can be written in terms of the the projections in Eqns. (3.17)–(3.20) as

$$\begin{aligned} M^{ab}_{(\nu)} &= \nu^3 \int \mathrm{d}\Omega \left[u^a u^b + u^a \ell^b + u^b \ell^a + \ell^a \ell^b \right] f_{(\nu)} \\ &= J_{(\nu)} u^a u^b + H^a_{(\nu)} u^b + H^b_{(\nu)} u^a + L^{ab}_{(\nu)}. \end{aligned} \tag{3.21}$$

By a similar process, the third-rank moment expansion can be expressed as

$$\begin{aligned} M^{abc}_{(\nu)} &= J_{(\nu)} u^a u^b u^c + H^a_{(\nu)} u^b u^c + H^b_{(\nu)} u^a u^c + H^c_{(\nu)} u^a u^b \\ &\quad + L^{ab}_{(\nu)} u^c + L^{ac}_{(\nu)} u^b + L^{bc}_{(\nu)} u^a + N^{abc}_{(\nu)}. \end{aligned} \tag{3.22}$$

Examining the structures of Eq. (3.16) Eqns. (3.21)–(3.22) shows that each higher-rank moment fully contains each lower-rank moment in the hierarchy. This fact can be exploited to obtain the projected moment of any rank from the unprojected moment of equal or higher rank. To demonstrate this, first note that the projected moments of rank greater than zero are orthogonal to the co-moving observer's four-velocity u^a since $u_a \ell^a = 0$, e.g. $u_a L^{ab} = 0$, and thus are spacelike in this local frame. Using this in Eq. (3.16) then shows that

$$u_b M^{A_k b}_{(\nu)} = -M^{A_k}_{(\nu)}. \tag{3.23}$$

Next, define the projection operator orthogonal to u^a as

$$h_{ab} = g_{ab} + u_a u_b, \tag{3.24}$$

where g_{ab} is the spacetime metric; this operator can also be viewed as the local spatial metric in the observer's restframe. With Eq. (3.24), the projected components and trace of Eq. (3.16) relate to lower-rank moments in the expansion as

$$h^b{}_c M^{A_k c}_{(\nu)} = M^{A_k b}_{(\nu)} - u^c M^{A_k}_{(\nu)} \tag{3.25}$$

$$h_{bc} M^{A_k bc}_{(\nu)} = M^{A_k}_{(\nu)} \tag{3.26}$$

The projected moments can now be obtained directly from the unprojected moments by making use of the identities in Eq. (3.23) and Eq. (3.25) in comparison to the expansion in Eq. (3.16) and the projected moments in Eqns. (3.17)–(3.20) as

$$J_{(\nu)} = M_{(\nu)} = -u_a M^a_{(\nu)} = u_a u_b M^{ab}_{(\nu)} = -u_a u_b u_c M^{abc}_{(\nu)} \tag{3.27}$$

$$H^a_{(\nu)} = h^a{}_b M^b_{(\nu)} = -h^a{}_b u_c M^{bc}_{(\nu)} = h^a{}_b u_c u_d M^{bcd}_{(\nu)} \tag{3.28}$$

$$L^{ab}_{(\nu)} = h^a{}_c h^b{}_d M^{cd}_{(\nu)} = -h^a{}_c h^b{}_d u_e M^{cde}_{(\nu)} \tag{3.29}$$

$$N^{abc}_{(\nu)} = h^a{}_d h^b{}_e h^c{}_f M^{def}_{(\nu)} . \tag{3.30}$$

Making similar projections of the moment evolution equation into components parallel and orthogonal to the co-moving observer's four-velocity will lead to a system of evolution equations for the projected moments. Unfortunately, using the decomposition of the moment expansion in terms of the co-moving frame projected moments from Eq. (3.21) directly in Eq. (3.12) does not produce flux-conservative evolution equations, even in the absence of neutrino-matter interactions. A flux-conservative formulation is a desirable property for maintaining energy and momentum conservation, facilitating consistent and stable numerical implementations. As such, the co-moving frame projections of the evolution equations will

not be considered any further in this chapter; please refer to Shibata et al. (2011) for a more detailed examination of these equations.

3.1.3 Eulerian Frame Projections of the Moment Expansion

The decomposition of the moment expansion and the moment evolution equations are covariant tensor expressions and are valid in any frame of reference for any choice of coordinate system (although the frequency is specific to the frame that the expansion is defined in). This allows freedom in choosing a frame of reference that is suitable for numerical computations. A choice widely used for dynamic spacetime and fluid evolution is a fixed Eulerian frame as viewed by a infinitely distant observer. In this frame, a '3+1' decomposition of the spacetime is made into spacelike hypersurfaces at constant coordinate times. Please refer to appendix C for an overview of this decomposition that will be used throughout the remainder of this chapter.

In a similar manner as the projections made in the co-moving frame, the unprojected moment expansion can be decomposed into components parallel and orthogonal to the four-velocity of an Eulerian observer – this four-velocity corresponds to the timelike normal direction of the spacelike hypersurfaces. Let n^a be the Eulerian four-velocity, such that $n_a n^a = -1$. A projection operator onto the spacelike hypersurfaces immediately follows by comparison to Eq. (3.24)

$$\gamma_{ab} = g_{ab} + n_a n_b, \tag{3.31}$$

which also plays the role of the metric on the spacelike hypersurfaces. With these, the second-rank unprojected moment can be decomposed in terms of n^a and Eulerian projections of the

moments (which are defined in terms of the following expression)

$$M^{ab}_{(\nu)} = E_{(\nu)} n^a n^b + F^a_{(\nu)} n^b + F^b_{(\nu)} n^a + P^{ab}_{(\nu)}. \tag{3.32}$$

As was the case for the co-moving frame projections, the Eulerian projections correspond to the energy density, momentum density, and pressure tensor of the neutrinos as measured by the Eulerian observer. Making use of the Eulerian four-velocity and projection operator, the projected moments can be found from Eq. (3.32)

$$E_{(\nu)} = n_a n_b M^{ab}_{(\nu)} \tag{3.33}$$

$$F^a_{(\nu)} = -\gamma^a{}_b n_c M^{bc}_{(\nu)} \tag{3.34}$$

$$P^{ab}_{(\nu)} = \gamma^a{}_c \gamma^b{}_d M^{cd}_{(\nu)}, \tag{3.35}$$

which follows from $n_a F^a_{(\nu)} = n_a P^{ab}_{(\nu)} = 0$.

An immediate complication posed by the Eulerian projections is that the neutrino frequency ν that these moments are specific to is not, in general, the same as the frequency measured by the Eulerian observer. The neutrino-matter interactions and redshifting effects, such as those caused by acceleration of the fluid, in the evolution equations are most easily defined in the fluid frame, so it is necessary to relate the co-moving frame projected moments to their Eulerian counterparts. To accomplish this, the co-moving observer's four-velocity can be decomposed into components parallel to the fluid's spatial velocity v^a and the Eulerian four-velocity n_a

$$u^a = W(n^a + v^a), \tag{3.36}$$

where $W = -u_a n^a$, commonly referred to as the Lorentz factor, quantifies the relative velocity difference between the two observers. From the normalization and orthogonality conditions $u_a u^a = n_a n^a = -1$ and $n_a v^a = 0$, the usual definition of the Lorentz factor $W = 1/\sqrt{1 - v_i v^i}$ is recovered from Eq. (3.36). Using Eq. (3.32) and Eq. (3.36) in Eqns. (3.27)–(3.28) then gives the relation of the co-moving frame energy and momentum densities to the Eulerian frame projected moments

$$J_{(\nu)} = W^2\Big(E_{(\nu)} - 2v_i F^i + v_i v_j P^{ij}_{(\nu)}\Big) \tag{3.37}$$

$$H^a_{(\nu)} = W F^a_{(\nu)} + W\Big(E_{(\nu)} + v_i F^i_{(\nu)}\Big)n^a - J_{(\nu)}u^a - W v_i P^{ia}_{(\nu)}. \tag{3.38}$$

Now the moment evolution equation can be brought into a more tractable form. Inserting the Eulerian decomposition of Eq. (3.32) into Eq. (3.12) and taking projections parallel and orthogonal to the Eulerian observer's four-velocity yields (see appendix D for a full derivation)

$$\begin{aligned}\frac{\partial}{\partial t}\Big[\tilde{E}_{(\nu)}\Big] + \frac{\partial}{\partial x^j}\Big[\alpha\tilde{F}^j_{(\nu)} - \beta^j\tilde{E}_{(\nu)}\Big] + \frac{\partial}{\partial \nu}\Big[\nu\alpha n_a \tilde{M}^{abc}_{(\nu)}\boldsymbol{\nabla}_b u_c\Big] \\ = \alpha\left[\tilde{P}^{ij}_{(\nu)}K_{ij} - \tilde{F}^i_{(\nu)}\frac{\partial \ln\alpha}{\partial x^i} - n_a\tilde{S}^a_{(\nu)}\right]\end{aligned} \tag{3.39}$$

and

$$\begin{aligned}\frac{\partial}{\partial t}\Big[\tilde{F}_{i,(\nu)}\Big] + \frac{\partial}{\partial x^j}\Big[\alpha\tilde{P}^j{}_{i,(\nu)} - \beta^j\tilde{F}_{i,(\nu)}\Big] - \frac{\partial}{\partial \nu}\Big[\nu\alpha\gamma_{ia}\tilde{M}^{abc}_{(\nu)}\boldsymbol{\nabla}_b u_c\Big] \\ = \alpha\left[\frac{\tilde{P}^{jk}_{(\nu)}}{2}\frac{\partial\gamma_{jk}}{\partial x^i} + \frac{\tilde{F}_{k,(\nu)}}{\alpha}\frac{\partial\beta^k}{\partial x^i} - \tilde{E}_{(\nu)}\frac{\partial\ln\alpha}{\partial x^i} + n_a\tilde{S}^a_{(\nu)}\right],\end{aligned} \tag{3.40}$$

where the 3+1 split spacetime quantities are the lapse function α, shift vector β^i, and the

extrinsic curvature of the spacelike hypersurfaces K_{ij}. In the preceding equations, the tilde-quantities represent the densitized form of the corresponding moment, e.g. $\tilde{E}_{(\nu)} \equiv \sqrt{\gamma} E_{(\nu)}$, where $\gamma = \det \gamma_{ij}$ is the determinant of the spatial metric and quantifies the local volume element in the Eulerian frame. These evolution equations form a hyperbolic system in a flux-conservative form, and must be solved for each neutrino species and frequency. In both Eqns. (3.39)–(3.40), the ν-derivative terms encapsulate the effects of velocity- and gravitational-dependent redshifting, and represent a "flux" along the neutrino frequency degree-of-freedom. These terms, along with some types of neutrino-matter interactions such as inelastic scattering on electrons, serve to tightly couple the systems of evolution equations between the neutrino frequencies.

Since the Eulerian projection of the moments and the evolution equation are for specific frequencies defined in the co-moving frame, their frequency-integrated forms follow in the exact same manner as

$$E = \int_0^\infty \mathrm{d}\nu \, E_{(\nu)} \tag{3.41}$$

$$F^a = \int_0^\infty \mathrm{d}\nu \, F^a_{(\nu)} \tag{3.42}$$

$$P^{ab} = \int_0^\infty \mathrm{d}\nu \, P^{ab}_{(\nu)} \tag{3.43}$$

and

$$\frac{\partial}{\partial t}\left[\tilde{E}\right] + \frac{\partial}{\partial x^j}\left[\alpha \tilde{F}^j - \beta^j \tilde{E}\right] = \alpha\left[\tilde{P}^{ij} K_{ij} - \tilde{F}^i \frac{\partial \ln \alpha}{\partial x^i} - n_a \tilde{S}^a\right] \tag{3.44}$$

$$\frac{\partial}{\partial t}\left[\tilde{F}_i\right] + \frac{\partial}{\partial x^j}\left[\alpha \tilde{P}^j{}_i - \beta^j \tilde{F}_i\right] = \alpha\left[\frac{\tilde{P}^{jk}}{2}\frac{\partial \gamma_{jk}}{\partial x^i} + \frac{\tilde{F}_k}{\alpha}\frac{\partial \beta^k}{\partial x^i} - \tilde{E}\frac{\partial \ln \alpha}{\partial x^i} + n_a \tilde{S}^a\right]. \tag{3.45}$$

As was the case with the frequency-integrated moment evolution equation in Eq. (3.11), the

terms containing frequency-derivatives vanish under integration. This frequency-integrated form of the evolution equations will not be considered any further, and the explicit ν-subscript will be omitted for the remainder of this chapter in order to simplify notation. All neutrino quantities will be assumed to be frequency-dependent unless otherwise stated.

3.1.4 Closure Relation

Since the evolution equations for the energy density and momentum density result from projections of the second-rank moment's evolution equation, knowledge of the higher-rank projections are necessary. The higher-rank moments are also necessary when relating the co-moving frame projections to the evolved Eulerian frame projections. Since the moments of second-rank and higher are not directly evolved (these would also always require the next-highest rank moments), the pressure tensor P^{ab} must be provided to close the system in Eqns. (3.39)–(3.40).

The common approach in this type of two-moment evolution scheme, also referred to as an M1 scheme, is to approximate the pressure tensor by interpolation between the optically thin and thick limiting forms that can be specified in terms of the energy and momentum densities. Since neutrino-matter interactions ultimately determine how optically thin or thick the fluid is, this determination must be made in the co-moving frame. For the M1 formulation, the interpolation takes the form

$$L^{ab} = \frac{3\chi(\xi) - 1}{2} L^{ab}_{\text{thin}} + \frac{3(1 - \chi(\xi))}{2} L^{ab}_{\text{thick}}, \tag{3.46}$$

where $\chi(\xi)$ is a closure function of the parameter ξ that characterizes how optically thin or thick the fluid is, and L^{ab}_{thin} and L^{ab}_{thick} are the optically thin and thick, respectively,

limiting forms of the co-moving frame pressure tensor. A closure function may be freely chosen (see Murchikova et al. (2017) for a comprehensive list and comparison of commonly used closures), but it must take on the following limiting values to correctly reproduce these limiting forms

$$\begin{aligned} \lim_{\xi\to 0}\chi(\xi) &= \frac{1}{3} \quad \text{(Optically Thick)} \\ \lim_{\xi\to 1}\chi(\xi) &= 1 \quad \text{(Optically Thin)}, \end{aligned} \tag{3.47}$$

where the parameter ξ ranges from zero to one in the optically thick and thin limits, respectively. One effective to define ξ such that these limits are met is by the ratio of the spatial momentum density and energy density in the co-moving frame (Shibata et al., 2011)

$$\xi^2 = \frac{h_{ab}H^a H^b}{J^2}. \tag{3.48}$$

In the co-moving frame, this form correctly sets $\xi = 0$ in the optically thick limit where the radiation is isotropic. Defining ξ in terms of the Eulerian quantities cannot guarantee this limit since completely completely trapped neutrinos in a moving fluid will not necessarily be observed as isotropic in the Eulerian frame, i.e. $\gamma_{ij}F^iF^j \neq 0$ as the neutrinos will have non-zero momentum in this frame due to advection of the fluid.

The structure of the moment expansion in terms of the projected moments allows the interpolation in Eq. (3.46) to be used for the other projected moments as well. By using Eq. (3.46) in Eq. (3.21) and then taking the projection Eq. (3.35), the Eulerian frame pressure tensor can also be expressed as

$$P^{ab} = \frac{3\chi(\xi)-1}{2}P^{ab}_{\text{thin}} + \frac{3(1-\chi(\xi))}{2}P^{ab}_{\text{thick}}. \tag{3.49}$$

By making use of both Eq. (3.46) and Eq. (3.49), the co-moving frame energy and moment densities can then be found in terms of E, F^i, and ξ via Eqns. (3.37)–(3.38). Higher-rank moments, such as N^{abc}, can also be expressed as interpolations between their optically thin and thick limiting forms.

The remainder of this section will focus on obtaining the limiting forms of the pressure tensor in the co-moving and Eulerian frames. A list of practical expressions of the co-moving frame energy and momentum densities suitable for numerical implementation are provided in appendix E.

3.1.4.1 Optically Thin Limit

In the optically thin limit, neutrinos are free-streaming in a vacuum or non-interacting fluid, and are assumed to be traveling at the speed of light. As massless particles (to good approximation), free-streaming neutrinos will follow null trajectories, i.e. their four-momenta obey $p^a p_a = 0$. As such, the magnitude of the neutrino momentum will always equal the neutrino energy, from which it follows

$$J = H = \sqrt{h_{ab} H^a H^b}, \quad H^a = J \frac{H^a}{H} = J \hat{h}^a \tag{3.50}$$

$$E = F = \sqrt{\gamma_{ab} F^a F^b}, \quad F^a = E \frac{F^a}{F} = E \hat{f}^a, \tag{3.51}$$

where H and F are the magnitudes of the neutrino momentum in the co-moving and Eulerian frames, respectively, and $\hat{h}^a$ and $\hat{f}^a$ are unit vectors in the streaming direction of the neutrinos in the co-moving and Eulerian frames, respectively.

Based on the relations in Eq. (3.50), any expression for the optically thin pressure tensor in the co-moving frame can only depend on the energy density and the streaming direction.

Using Eq. (3.21) in Eq. (3.26) further constrains the trace of the pressure tensor to be

$$h_{ab}L^{ab} = J \tag{3.52}$$

. With both of these constraints, a suitable form of the pressure tensor is

$$L^{ab}_{\text{thin}} = J\hat{h}^a\hat{h}^b. \tag{3.53}$$

By a similar argument, since the neutrino momentum as observed in the Eulerian frame does not contain any contribution from the motion of the fluid in the optically thin limit, the Eulerian frame pressure tensor can also be written as

$$P^{ab}_{\text{thin}} = E\hat{f}^a\hat{f}^b. \tag{3.54}$$

Alternative formulations of the Eulerian frame pressure tensor exist; see Shibata et al. (2011) for a further discussion of these forms (including the one used here) and potential issues posed when using these in the closure interpolation.

The third-rank projected moment can be found by using Eq. (3.53) in Eq. (3.22) and using the trace condition in Eq. (3.26) to show that

$$h_{bc}N^{abc}_{\text{thin}} = J\hat{h}^a. \tag{3.55}$$

As with the second-rank moment, it will be necessary to express the third-rank projection only in terms of the energy density and momentum density. Making use of the identity $h_{ab}\hat{h}^a\hat{h}^b = 1$, a straightforward extension of the procedure used to obtain the second-rank

projected moment gives the third-rank projected moment of the form

$$N^{abc}_{\text{thin}} = J\hat{h}^a\hat{h}^b\hat{h}^c. \tag{3.56}$$

3.1.4.2 Optically Thick Limit

In the optically thick limit, the neutrinos are completely trapped by the fluid, leading to an isotropic distribution. To satisfy the trace condition in Eq. (3.26) and Eq. (3.52), the optically thick limit of the co-moving frame pressure tensor must take the form

$$L^{ab}_{\text{thick}} = \frac{J}{3}h^{ab}. \tag{3.57}$$

Applying the trace condition in Eq. (3.26) to the third-rank moment expansion in Eq. (3.22) using the form of L^{ab} given in Eq. (3.57) shows that the third-rank projection's trace must satisfy

$$h_{bc}N^{abc}_{\text{thick}} = H^a. \tag{3.58}$$

To preserve the isotropic character of the neutrino radiation, a suitable form for the third-rank projected moment in the optically thick limit is

$$N^{abc}_{\text{thick}} = \frac{1}{5}\Big(H^a h^{bc} + H^b h^{ac} + H^c h^{ab}\Big). \tag{3.59}$$

Unlike the optically thin limit, the Eulerian frame pressure tensor cannot be specified in as straightforward of a manner, as the radiation is not necessarily isotropic in this frame. Instead, it must be obtained from the projection of the optically thick limiting form of the

second-rank moment expansion

$$M^{ab}_{\text{thick}} = Ju^a u^b + H^a u^b + H^b u^a + \frac{J}{3}h^{ab}$$
$$= \frac{J}{3}\Big(g^{ab} + 4u^a u^b\Big) + H^a u^b + H^b u^a. \tag{3.60}$$

By using Eq. (3.60) in the projection in Eq. (3.35), the optically thick form of the Eulerian pressure tensor is

$$P^{ij}_{\text{thick}} = \frac{J}{3}\Big(\gamma^{ij} + 4W^2 v^i v^j\Big) + W\Big(\bar{H}^i v^j + \bar{H}^j v^i\Big), \tag{3.61}$$

where $\bar{H}^i = \gamma^i{}_a H^a$ is the Eulerian spatial projection of the co-moving frame momentum density[1], and Eq. (3.31) and Eq. (3.36) were used to expand g^{ab} and u^a, respectively.

An immediate complication with this form of the Eulerian frame projection is the energy and moment densities in the co-moving frame must now be expressed in their optically thick limiting forms in terms of their known Eulerian counterparts. To find these expressions, Eq. (3.61) can be used in the projections in Eqns. (3.27)–(3.28) to show that

$$E = \frac{1}{3}\Big(4W^2 - 1\Big)J_{\text{thick}} + 2Wv_i H^i_{\text{thick}} \tag{3.62}$$
$$F^i = \left(\frac{4}{3}W^2 J_{\text{thick}} + Wv_i H^i_{\text{thick}}\right)v^i + W\bar{H}^i_{\text{thick}}, \tag{3.63}$$

where J_{thick} and H^a_{thick} are the optically thick limiting forms of the co-moving frame energy and momentum densities, respectively. Inverting these expressions for the co-moving frame projections continues by contracting Eq. (3.63) with the spatial velocity v^i, then solving for

[1]Barred co-moving frame quantities will always represent their Eulerian frame spatial projections

the Wv_iH^a term, and finally using the result in Eq. (3.62) to find

$$J_{\text{thick}} = \frac{3}{2W^2+1}\Big[\Big(2W^2-1\Big)E - 2W^2 v_i F^i\Big] \tag{3.64}$$

$$\bar{H}^i_{\text{thick}} = \frac{F^i}{W} - \frac{W}{2W^2+1}\Big[4W^2E - \Big(4W^2+1\Big)v_k F^k\Big]v^i. \tag{3.65}$$

Using these expressions in Eq. (3.61) then gives the optically thick limiting form of the Eulerian frame pressure tensor as

$$\begin{aligned} P^{ij}_{\text{thick}} = {} & \frac{1}{2W^2+1}\Big[\Big(2W^2-1\Big)E - 2W^2 v_k F^k\Big]\gamma^{ij} \\ & - \frac{2W^2}{2W^2+1}\Big[2E - v_k F^k\Big]v^i v^j + F^i v^j + F^j v^i. \end{aligned} \tag{3.66}$$

3.1.5 Redshifting Effects

The frequency of the neutrinos in the co-moving frame can change due to both the acceleration and shear of the fluid, as well as the curvature of the spacetime. These velocity- and gravitational-induced redshifting effects are captured in frequency-derivatives in the evolution equations by the projection of the third-rank moment expansion onto the covariant derivative of the co-moving observer's four-velocity. While this can be done in the co-moving frame, it is simpler to perform these calculations in the Eulerian frame where the evolved neutrino energy and momentum densities, fluid velocity, and metric quantities are defined. By making use of the Eulerian decomposition of the co-moving observer's four-velocity in Eq. (3.36), its covariant derivative takes the form

$$\boldsymbol{\nabla}_b u_c = \boldsymbol{\nabla}_b[W(n_c + v_c)] = W\boldsymbol{\nabla}_b n_c + n_c\frac{\partial W}{\partial x^b} + \boldsymbol{\nabla}_n(Wv_c). \tag{3.67}$$

Unlike the co-moving frame moment-expansion which can readily be defined by its irreducible parts, e.g. J is the zeroth-rank part of both the second- and third-rank expansions, the same hierarchy does not generally hold for the Eulerian frame decompositions. Instead, following the method presented in Cardall et al. (2013), the third-rank moment expansion can be defined as

$$\begin{aligned} M^{abc} = Qn^a n^b n^c + R^a n^b n^c + R^b n^a n^c + R^c n^a n^b \\ S^{ab} n^c + S^{ac} n^b + S^{bc} n^a + T^{abc}, \end{aligned} \tag{3.68}$$

where, similarly to the projections of the Eulerian second-rank moment-expansion, the projections of first-rank and higher are purely spatial, i.e. $n_a R^a = n_a S^{ab} = n_a T^{abc} = 0$. With this definition, the projected moments are found in terms of the co-moving frame projections

$$Q = -n_a n_b n_c M^{abc} = W^3 J + 3W^2 v_i \bar{H}^i + 3W v_i v_j \bar{L}^{ij} + v_i v_j v_k N^{ijk} \tag{3.69}$$

$$\begin{aligned} R^i = \gamma^i{}_a n_b n_c M^{abc} &= \left[W^3 J + 2W^2 v_j \bar{H}^j + W v_j v_k \bar{L}^{jk}\right] v^i + W^2 \bar{H}^i \\ &\quad + 2W v_j \bar{L}^{ij} + v_j v_k \bar{N}^{ijk} \end{aligned} \tag{3.70}$$

$$\begin{aligned} S^{ij} = -\gamma^i{}_a \gamma^j{}_b n_c M^{abc} &= \left[W^3 J + W^2 v_k \bar{H}^k\right] v^i v^j + \left[W^2 \bar{H}^i + W v_k \bar{L}^{ik}\right] v^j \\ &\quad + \left[W^2 \bar{H}^j + W v_k \bar{L}^{jk}\right] v^i + W \bar{L}^{ij} + v_k \bar{N}^{ijk} \end{aligned} \tag{3.71}$$

$$\begin{aligned} T^{ijk} = \gamma^i{}_a \gamma^j{}_b \gamma^k{}_c M^{abc} &= W^3 J v^i v^j v^k + W^2 \left[\bar{H}^i v^j v^k + \bar{H}^j v^i v^k + \bar{H}^k v^i v^j\right] \\ &\quad + W\left[\bar{L}^{ij} v^k + \bar{L}^{ik} v^j + \bar{L}^{jk} v^i\right] + \bar{N}^{ijk}, \end{aligned} \tag{3.72}$$

where the barred-quantities are again the Eulerian spatial projections of the co-moving frame moments. In general, Q, R^i, and S^{ij} are not the same as the similar-rank projections of the second-rank moment expansion. They are only equivalent for the trivial case of $v^i = 0$,

such that both observers have the same four-velocity, $u^a = n^a$, and the projected moments in each frame take on the same values.

With the Eulerian decompositions of the covariant derivative of the four-velocity in Eq. (3.67) and the third-rank moment in Eq. (3.68), the contraction at the core of the redshifting term in the evolution equations becomes a straightforward (but lengthy) exercise of tensor algebra; please refer to appendix D.3 for the step-by-step details. In its final form, this term can now be expressed in the form

$$M^{abc}\boldsymbol{\nabla}_b u_c = -C^{\{0\}}(Qn^a + R^a) - C_i^{\{1\}}\Big(R^i n^a + S^{ai}\Big) - C_{ij}^{\{2\}}\Big(S^{ij} n^a + T^{aij}\Big), \tag{3.73}$$

where the tensor coefficients are

$$C^{\{0\}} = \left[n^a \frac{\partial W}{\partial x^a}\right] + \left[W v^k \frac{\partial \ln \alpha}{\partial x^k}\right] \tag{3.74}$$

$$C_i^{\{1\}} = \left[\frac{\partial W}{\partial x^i} - n^a \frac{\partial W v_i}{\partial x^a}\right] + \left[-W \frac{\partial \ln \alpha}{\partial x^i} - \frac{W v^k}{\alpha} \frac{\partial \beta_i}{\partial x^k}\right] \tag{3.75}$$

$$C_{ij}^{\{2\}} = \left[-\frac{\partial W v_i}{\partial x^j}\right] + \left[W K_{ij} - \frac{W v^k}{2} \frac{\partial \gamma_{ij}}{\partial x^k}\right], \tag{3.76}$$

where the terms in the left brackets characterize the acceleration and shear of the co-moving observer, and the terms in the right brackets characterize the curvature of the spacetime. The timelike and spacelike projections in the evolution equations, Eqns. (3.39)–(3.40) trivially follow as

$$n_a M^{abc}\boldsymbol{\nabla}_b u_c = C^{\{0\}} Q + C_i^{\{1\}} R^i + C_{ij}^{\{2\}} S^{ij} \tag{3.77}$$

$$-\gamma_{ia} M^{abc}\boldsymbol{\nabla}_b u_c = C^{\{0\}} R_i + C_j^{\{1\}} S^j{}_i + C_{jk}^{\{2\}} T^{jk}{}_i. \tag{3.78}$$

When the spatial and temporal gradients of the velocity are zero, a non-flat spacetime will still lead to a "flux" between the neutrino frequencies. This effect is even present in vacuum or when there is no finite velocity, as the contraction of the higher-rank projections with the extrinsic curvature K_{ij} may still be non-zero. The redshifting effects will only vanish in the frequency-dependent evolution equations for the trivial case of a flat spacetime and a zero background fluid velocity.

3.1.6 Neutrino-Matter Interactions

The collision integral in the Boltzmann equation, and consequently its moment-expansion form, determines the rate-of-change in the underlying neutrino distribution. In the second-rank evolution equations, Eqns. (3.39)–(3.40), the moment-expansion source vector $S^a_{(\nu)}$ characterizes the transfer of energy and momentum from neutrinos of frequency ν to the fluid. To maintain energy and momentum conservation, the net transfer of energy and momentum of all neutrino species over the full frequency spectrum must also be accounted for in the evolution of the fluids energy and momentum densities. The total interaction source vector is then

$$S^a_{\text{tot.}} = \sum_{\sigma} \int_0^{\infty} \mathrm{d}\nu \, S^a_{(\sigma,\nu)}, \tag{3.79}$$

where the additional subscript σ from the summation is over all neutrino species. For a hydrodynamics formulation similar in structure to the moment evolution equations, i.e. a formulation based on the covariant divergence of the fluid's stress-energy tensor, the fluid's energy and momentum evolution equations will contain the timelike and spacelike projections, respectively, of Eq. (3.79), but with an opposite sign as those in Eqns. (3.39)–(3.40).

The source vector takes on different forms for different types of neutrino-matter interactions. To simplify the analysis and facilitate adding new types of interactions, the interaction source will be separated into terms representing different processes,

$$S^a = S^a_{\rm eq} + S^a_{\rm iso}, \tag{3.80}$$

where "eq" denotes emission and absorption of neutrinos by the fluid, and "iso" denotes isoenergetic scattering of neutrinos off of nucleons and nuclei. The remainder of this section will describe the form that each of these terms takes. The notation used in Rampp & Janka (2002); Shibata et al. (2011) $B \equiv [\mathrm{d}f/\mathrm{d}t]_{\rm coll.}$ for the collision integral will be used throughout, where the explicit subscripts for the neutrino frequency and species will be suppressed for readability.

3.1.6.1 Emission and Absorption

Emission and absorption of neutrinos result from electron-capture type processes such as $\nu_e + n \leftrightharpoons e^- + p$. The total rate of the sum of these processes can be characterized by an emissivity j and the mean-free path of the neutrinos λ. Since neutrinos are fermions, the reaction rates for all of these processes will be subject to final-state Pauli blocking, also referred to as stimulated absorption (Rampp & Janka, 2002; Burrows et al., 2006), and the collision integral takes the form (Bruenn, 1985)

$$B_{\rm ea} = j(1-f) - \frac{f}{\lambda}, \tag{3.81}$$

where $f \equiv f(\nu, \Omega)$ is the frequency-dependent neutrino distribution function that appears in the Boltzmann equation and moment-expansion. The emissivity and mean-free path are

assumed to only depend on the frequency ν of the neutrinos, but not on the direction of the momentum Ω. This form of the collision integral suggests a total absorption opacity (Rampp & Janka, 2002)

$$\kappa_{\text{abs}} = j + \frac{1}{\lambda}, \tag{3.82}$$

such that the collision integral takes the form

$$B_{\text{ea}} = j - \kappa_{\text{abs}} f. \tag{3.83}$$

The emissivity can now be brought into a form more suitable for use in the moment-expansion of the collision integral. When $B_{\text{ea}} = 0$, such that absorption and emission rates are balanced, the neutrinos will be in thermal equilibrium with the fluid, described by an equilibrium Fermi-Dirac distribution (Burrows et al., 2006)

$$f_{\text{eq}} = \frac{1}{e^{(\nu - \mu_\nu)}/T + 1}, \tag{3.84}$$

where μ_ν is the chemical potential of the neutrinos (which goes to zero in the β-equilibrium case). It immediately follows the the emissivity can be expressed as

$$j = \kappa_{\text{abs}} f_{\text{eq}}, \tag{3.85}$$

and the collision integral then takes the form

$$B_{\text{ea}} = \kappa_{\text{abs}} \big(f_{\text{eq}} - f \big). \tag{3.86}$$

The form of the collision integral is aptly suited for use in the moment-expansion of the collision integral in Eq. (3.10). By defining the equilibrium energy density in the co-moving frame by using Eq. (3.84) in Eq. (3.17) as

$$J_{\rm eq} = \nu^3 \int {\rm d}\Omega \, f_{\rm eq}, \tag{3.87}$$

and assuming that the neutrino radiation is completely isotropic such that $H^a_{\rm eq} = 0$, performing the first-rank expansion of the collision integral for emission and absorption then yields

$$\begin{aligned} S^a_{\rm ea} &= \kappa_{\rm abs} J_{\rm eq} u^a - \kappa_{\rm abs}(J u^a + H^a) \\ &= \kappa_{\rm abs}\big(J_{\rm eq} - J\big) u^a - \kappa_{\rm abs} H^a. \end{aligned} \tag{3.88}$$

3.1.6.2 Isoenergetic Scattering

When neutrinos scatter off of much heavier nucleons and nuclei, there is no assumed energy exchange (Bruenn, 1985), so these scattering processes are elastic. To analyze these processes, it will be useful to simplify the integrals over the angular degrees-of-freedom by defining the unit vector in the spatial direction of the momentum as $\ell^i = (\cos\phi \sin\theta, \sin\phi \sin\theta, \cos\theta)$, for polar and azimuthal angles θ and ϕ, respectively, such that the integrals take the form

$$\int {\rm d}\Omega \quad \rightarrow \quad \int_{-1}^{1} {\rm d}\mu \int_0^{2\pi} {\rm d}\phi \, , \tag{3.89}$$

where $\mu = \cos\theta$.

The total isoenergetic scattering rate must account for all possible incoming and outgoing

scattering angles, and when accounting for initial and final state blocking take the form (Bruenn, 1985; Shibata et al., 2011)

$$B_{\rm iso} = \nu^2 \int_{-1}^{1} \mathrm{d}\mu' \int_{0}^{2\pi} \mathrm{d}\phi' \left[f' - f\right] R_{\rm iso}(\omega), \tag{3.90}$$

where $f' \equiv f(\nu, \Omega')$ and $R_{\rm iso}(\omega)$ is the isoenergetic scattering kernel as function of the angle between the incoming and outgoing neutrino directions

$$\omega = \mu\mu' + \cos\left(\phi - \phi'\right)\sqrt{\left(1-\mu^2\right)\left(1-\mu'^2\right)}. \tag{3.91}$$

A common approach taken is to expand the scattering kernel in powers of ω, retaining only up to terms linear in ω, and the distribution function in powers of ℓ^a up to second-rank (Shibata et al., 2011)

$$R_{\rm iso} \approx R^0_{\rm iso} + \omega R^1_{\rm iso}, \tag{3.92}$$

and

$$f \approx f^0 + f^1_a \ell^a + f^2_{ab} \ell^a \ell^b, \tag{3.93}$$

where the coefficients with numeric superscripts are all independent of the angle. With these definitions, using Eqns. (3.91)–(3.93) in Eq. (3.90) shows that

$$B_{\rm iso} = 4\pi\nu^2 \left[\frac{1}{3} R^1_{\rm iso} f^1_a \ell^a - R^0_{\rm iso}\left(f^1_a \ell^a + f^2_{ab} \ell^a \ell^b\right)\right]. \tag{3.94}$$

With this form of the isoenergetic scattering collision integral, its first-rank moment expansion evaluates to

$$S^a_{\rm iso} = 4\pi\nu^2\left[\frac{1}{3}R^1_{\rm iso} - R^0_{\rm iso}\right]H^a, \tag{3.95}$$

where the terms proportional to ℓ^a and $\ell^a\ell^b\ell^c$ go to zero under integration over $\mathrm{d}\Omega$ in the expansion. Defining the isoenergetic scattering opacity as

$$\kappa_{\rm iso} = 4\pi\nu^2\left[R^0_{\rm iso} - \frac{1}{3}R^1_{\rm iso}\right], \tag{3.96}$$

Eq. (3.95) can be simply written as

$$S^a_{\rm iso} = -\kappa_{\rm iso}H^a. \tag{3.97}$$

3.1.6.3 Full Form of the Neutrino-Matter Interaction Source Terms

With the expansions of the collision integral given Eq. (3.88) and Eq. (3.97), the moment expansion of the full collision integral is then

$$S^a = \kappa_{\rm abs}\big(J_{\rm eq} - J\big)u^a - (\kappa_{abs} + \kappa_{\rm iso})H^a. \tag{3.98}$$

The projections necessary for the evolution equations, Eqns. (3.39)–(3.40) are then trivially found as

$$-n_aS^a = W\kappa_{\rm abs}\big(J_{\rm eq} - J\big) + (\kappa_{\rm abs} + \kappa_{\rm iso})n_aH^a \tag{3.99}$$

$$\gamma_{ia}S^a = W\kappa_{\rm abs}\big(J_{\rm eq} - J\big)v_i - (\kappa_{\rm abs} + \kappa_{\rm iso})\bar{H}_i, \tag{3.100}$$

where again $\bar{H}_i$ is the Eulerian frame spatial projection of the co-moving frame momentum density. With this formulation, all details of the reaction rates based on the properties of the fluid are encapsulated in the absorption and isoenergetic scattering opacities. This allows greater flexibility in choosing which interactions are included without having to change the form of these source terms in the evolution equations.

3.2 Numerical Methods

This section will describe the numerical methods used for solving moment evolution equations. Implementation details of the solver developed for Flash-X will be covered later in chapter 6. To better illustrate these methods, Eqns. (3.39)–(3.40) will be written in the compact form

$$\partial_t \mathbf{U} + \partial_j \mathbf{F}^j(\mathbf{U}) + \partial_\nu \mathbf{R}(\mathbf{U}) = \mathbf{G}(\mathbf{U}) + \mathbf{S}(\mathbf{U}), \tag{3.101}$$

where the derivative operators are $\partial_t \equiv \partial/\partial t$, $\partial_j \equiv \partial/\partial x^j$, and $\partial_\nu \equiv \partial/\partial \nu$. The evolved variables are represented by the vector

$$\mathbf{U} = \begin{bmatrix} \tilde{E} \\ \\ \tilde{F}_i \end{bmatrix}. \tag{3.102}$$

The remaining terms are all treated as functions of the evolved variables, with

$$\mathbf{F}^j(\mathbf{U}) = \begin{bmatrix} \alpha \tilde{F}^j - \beta^j \tilde{E} \\ \\ \alpha \tilde{P}^j_{\ i} - \beta^j \tilde{F}_i \end{bmatrix} \tag{3.103}$$

representing advection in the j-th spatial direction,

$$\mathbf{R}(\mathbf{U}) = \nu\alpha \begin{bmatrix} n_a \tilde{M}^{abc} \boldsymbol{\nabla}_b u_c \\ -\gamma_{ia} \tilde{M}^{abc} \boldsymbol{\nabla}_b u_c \end{bmatrix} \tag{3.104}$$

representing advection along the frequency "axis" of the neutrinos,

$$\mathbf{G}(\mathbf{U}) = \alpha \begin{bmatrix} \tilde{P}^{jk} K_{jk} - \tilde{F}^i \partial_i \ln \alpha \\ \frac{1}{2} \tilde{P}^{jk} \partial_i \gamma_{jk} + \alpha^{-1} \tilde{F}_j \partial_i \beta^j - \tilde{E} \partial_i \ln \alpha \end{bmatrix} \tag{3.105}$$

representing geometric source terms resulting from the covariant divergence of the second-rank moment expansion, and finally

$$\mathbf{S}(\mathbf{U}) = \alpha \begin{bmatrix} -n_a \tilde{S}^a \\ \gamma_{ia} \tilde{S}^a \end{bmatrix} \tag{3.106}$$

representing the energy and momentum exchange due to interactions between neutrinos and the fluid. The remainder of this section will cover the methods used for solving the closure relation and performing temporal, spatial, and frequency discretizations of each of the terms in Eq. (3.101).

3.2.1 M1 Closure

Closing the evolution equations and interpolating the co-moving frame quantities requires specifying a closure function $\chi(\xi)$. As previously mentioned earlier in this chapter, the closure function must be able to reproduce both the optically thin and thick limiting forms of the co-moving frame pressure tensor, but is otherwise freely specifiable. There are many different

closure functions commonly used in two-moment evolution schemes, which all take different approaches to arrive at a functional form of $\chi(\xi)$, ranging from simple weighted averages to multi-parameter fits of analytic models or direct solutions of the Boltzmann equation (e.g. from Monte Carlo particle methods). For a comprehensive list and comparison of widely used closure functions, please refer to Murchikova et al. (2017).

The implementation described in this chapter will make use of the maximum-entropy Minerbo closure (Minerbo, 1978; Cernohorsky & Bludman, 1994)

$$\chi(\xi) = \frac{1}{3} + \frac{2\xi^2}{15}\Big(3\xi^2 - \xi + 3\Big). \tag{3.107}$$

This form of the closure results from maximizing the entropy as a functional of the distribution function, and has an efficient-to-compute polynomial form. The Minerbo closure is widely used in general-relativistic neutrino radiation transport codes; see Foucart et al. (2015); Radice et al. (2022); Cheong et al. (2023) for examples.

Obtaining the pressure tensor and various co-moving frame projected moments from their closure interpolations requires solving Eq. (3.48) for the closure factor ξ. Unfortunately, Eq. (3.48) is implicit in the closure factor, as both J and H^a must be interpolated via the closure from the optically thin and thick limiting forms. As such, ξ must be solved for numerically. First, Eq. (3.48) is transformed into the function

$$\mathcal{G}(\xi) = [J(\xi)]^2\xi^2 - h_{ab}H^a(\xi)H^b(\xi), \tag{3.108}$$

where the dependence of the co-moving frame energy and momentum densities on the closure factor is shown explicitly. Solving for the closure factor now reduces to finding the numerical

root of $\mathcal{G}(\xi) = 0$. Since the closure factor is always in the range $0 \leq \xi \leq 1$, the bracketed Brent root-finding algorithm (Brent, 2002) is used to solve Eq. (3.108). A Newton-Raphson iterative method will typically converge on a solution faster with an adequate initial guess, e.g. a previous known value of ξ or the zero-velocity limit F/E, but in practice can struggle to converge in the limit of relativistic velocities near the boundaries $\xi = 0, 1$. This root-finding procedure is performed for every new set of E and F_i.

3.2.2 Spatial Advection

The spatial advection operator $\partial_j \mathbf{F}(\mathbf{U})$ makes use of either a finite-volume or finite-difference discretization. The finite-volume discretization is well suited for the flux-conservative form of Eq. (3.101), and works well with the cell-centered mesh utilized by Flash-X. The alternative finite-difference discretization seeks parity with the finite-difference hydrodynamics scheme (see chapter 4), and performs comparably to its finite-volume counterpart.

This section will describe the eigenstructure of the spatial advection operator and the finite-volume and finite-difference discretizations.. Each discretization will divide the domain into a mesh of cells, and the evolved variables will be located at the cell-centers. Both will be described along a single spatial direction, but will generalize to all directions. In the following, the notation $\mathbf{U}_i$ will represent cell-centered quantities in the i-th along an axis in the mesh, and $\mathbf{U}_{i\pm1/2}$ will represent the upper (+) and lower (-) face-centered quantities in the i-th cell.

3.2.2.1 Eigenstructure

The eigenstructure of the spatial advection operator describes the characteristic trajectories and speeds of the solution variables in space and time. These characteristics are necessary for

determining the upwind direction of the flow in order to maintain numerical stability when discretizing this operator. To examine these characteristics, the spatial advection operator is first written in the form (suppressing the direction superscript j for readability)

$$\partial_j \mathbf{F}(\mathbf{U}) = \frac{\partial \mathbf{F}}{\partial \mathbf{U}} \partial_j \mathbf{U} = \mathbf{A} \partial_j \mathbf{U}, \tag{3.109}$$

where A is the jacobian matrix of the flux vector $\mathbf{F}$. Using Eqns. (3.102)–(3.103) in Eq. (3.109), the jacobian takes the form

$$\mathbf{A} = \begin{bmatrix} -\beta^j & \alpha\gamma^{jk} \\ \alpha \dfrac{\partial P^j}{\partial E} & \alpha \dfrac{\partial P^j}{\partial F_k} - \beta^j \gamma^k \end{bmatrix}, \tag{3.110}$$

where i and k represent the row and column indices, respectively; j is fixed for each direction. The characteristic speeds are then simply the eigenvalues of $\mathbf{A}$ obtained from the solution of

$$\det\left(\mathbf{A} - \lambda \mathbf{I}\right) = 0, \tag{3.111}$$

λ is an eigenvalue of $\mathbf{A}$, and $\mathbf{I}$ is the identity matrix. However, in this most general form, the derivatives of the pressure tensor with respect to the energy and momentum densities are complicated by the closure factor ξ. An equivalent approach to find the characteristic speeds is to solve Eq. (3.111) separately for the optically thin and thick limits and interpolate the characteristic speeds in the same way as the pressure tensor (Shibata et al., 2011).

For the optically thin limit, taking the derivatives of the pressure tensor in Eq. (3.54)

with respect to the energy and moment densities yields

$$\frac{\partial}{\partial E}\Big[P^j{}_i\Big]_{\text{thin}} = \hat{f}^j \hat{f}_i \tag{3.112}$$

$$\frac{\partial}{\partial F_k}\Big[P^j{}_i\Big]_{\text{thin}} = \frac{E}{F}\Big[\Big(\gamma^{jk} - 2\hat{f}^j \hat{f}^k\Big)\hat{f}_i + \gamma^k{}_i \hat{f}^j\Big]. \tag{3.113}$$

Using these derivatives in the Eq. (3.110) leads to the optically thin limiting forms of the characteristic speeds in the j-th direction

$$\lambda^{\pm}_{\text{thin}} = -\beta^j \pm \alpha \hat{f}^j \tag{3.114}$$

$$\lambda^{0}_{\text{thin}} = -\beta^j + \alpha \frac{E}{F}\hat{f}^j, \tag{3.115}$$

where the characteristic speed λ^0_{thin} is doubly degenerate. Similarly for the optically thick limit, taking the derivatives of the pressure tensor in Eq. (3.66) with respect to the energy and moment densities yields

$$\frac{\partial}{\partial E}\Big[P^j{}_i\Big]_{\text{thick}} = \frac{1}{2W^2+1}\Big[\Big(2W^2-1\Big)\gamma^j{}_i - 4W^2 v^j v_i\Big] \tag{3.116}$$

$$\frac{\partial}{\partial F_k}\Big[P^j{}_i\Big]_{\text{thick}} = \frac{2W^2}{2W^2+1}\Big(v^j v_i - \gamma^j{}_i\Big)v^k + \gamma^{jk}v_i + \gamma^k{}_i v^j. \tag{3.117}$$

Using these derivatives in the Eq. (3.110) leads to the optically thick limiting forms of the characteristic speeds in the j-th direction

$$\lambda^{\pm}_{\text{thick}} = -\beta^j + \frac{2W^2 p^j \pm \sqrt{\alpha^2\big(2W^2+1\big)\gamma^{jj} - 2W^2 p^j p^j}}{2W^2+1} \tag{3.118}$$

$$\lambda^{0}_{\text{thick}} = -\beta^j + p^j, \tag{3.119}$$

where $p^j = \alpha\gamma^{jk}u_k/W = \alpha v^j$[2], and once again the characteristic speed $\lambda^0_{\rm thick}$ is doubly degenerate. The closure interpolated characteristic speeds are then given as

$$\lambda^{\pm,0} = \frac{3\chi(\xi)-1}{2}\lambda^{\pm,0}_{\rm thin} + \frac{3[1-\chi(\xi)]}{2}\lambda^{\pm,0}_{\rm thick}. \tag{3.120}$$

3.2.2.2 Finite-Volume Discretization

For the finite-volume discretization, the cell-centered $\mathbf{U}_i$ are treated as the volume-averages over the i-th cell. The spatial flux of the evolved variables in and out of each cell are approximated at the interfaces between the cells. The values of the evolved variables needed in evaluating Eq. (3.103) are not known at the cell-interfaces, so they must be reconstructed from their cell-centered values.

In each cell, the values at the upper (+) and lower (-) interface are reconstructed using a total-variation diminishing piecewise linear interpolation over a stencil including the adjacent cells (Kurganov & Tadmor, 2002; Toro, 2009)

$$\mathbf{U}^{\pm}_i = \mathbf{U}_i \pm \frac{1}{2}\phi(r,\theta)\Delta\mathbf{U}_{i+1}, \tag{3.121}$$

where $\phi(r,\theta)$ is the generalized minmod slope limiter

$$\phi(r,\theta) = \max\left[0, \min\left(r\theta, \frac{r+1}{2}m\theta\right)\right], \quad 1 \leq \theta \leq 2, \tag{3.122}$$

[2]Most works citing the optically thick characteristic speeds of Eq. (6.35) in Shibata et al. (2011) incorrectly assume that $p^j = \alpha V^j/W \equiv \alpha v^j/W$. However, in section 6.2 of Shibata et al. (2011), V^j is defined as the spatially projected co-moving frame four-velocity $V^j = \gamma^{jk}u_k = Wv^j$.

and the slopes and their ratio are defined as

$$r = \frac{\Delta \mathbf{U}_i}{\Delta \mathbf{U}_{i+1}}, \quad \Delta \mathbf{U}_i = \mathbf{U}_i - \mathbf{U}_{i-1}. \tag{3.123}$$

Use of a generalized minmod limiter provides a means of varying the amount of dissipation in the reconstructed slopes via the parameter θ. The limiter is most dissipative when $\theta = 1$, and least dissipative when $\theta = 2$. This flexibility allows more dissipation to be added when necessary, e.g. to smooth out potential overshoots near shocks and rarefactions in the fluid.

In practice, the neutrino moments E and F_i/E are reconstructed instead of the evolved densitized moments $\tilde{E}$ and $\tilde{F}_i$. To minimize errors in reconstruction, the densitization factor $\sqrt{\gamma}$ is removed since linear interpolation may not always be suitable for spacetime quantities. For instance, with a simple spherically symmetric Minkowski spacetime, the factor $\sqrt{\gamma} = r^2$ will not be correctly reconstructed when the reconstruction stencil overlaps the reflective boundary at the origin. Reconstructing the ratio F_i/E prevents the superluminal momentum densities that can occur when reconstructing F_i directly (Foucart et al., 2015; O'Connor, 2015). Additionally, the fluid velocity v^i is reconstructed in a similar manner, but the spacetime quantities α, β^i, and γ_{ij}, which are smooth in space, are reconstructed with a fourth-order Lagrange interpolating polynomial. All derived quantities, including the closure factor ξ, pressure tensor P^{ij}, characteristic speeds, inverse spatial metric γ^{ij}, its determinant γ, and the fluid's Lorentz factor W, are calculated in terms of the reconstructed interface values.

There are now two reconstructed states for each interface, the upper face from the cell to the left and the lower face from the cell to the right. In general, these two states are not necessarily the same, and form a local Riemann problem that must be solved to resolve the

discontinuity. To proceed, the left (L) and right (R) states at the $i+1/2$ interface are chosen as

$$\mathbf{U}^{\mathrm{L}}_{i+1/2} = \mathbf{U}^{+}_{i}, \quad \mathbf{U}^{\mathrm{R}}_{i+1/2} = \mathbf{U}^{-}_{i+1}, \tag{3.124}$$

and then used to compute the spatial fluxes via Eq. (3.103) as

$$\Big[\mathbf{F}^{j}\Big]^{\mathrm{L}}_{i+1/2} = \mathbf{F}^{j}\Big(\mathbf{U}^{\mathrm{L}}_{i+1/2}\Big), \quad \Big[\mathbf{F}^{j}\Big]^{\mathrm{R}}_{i+1/2} = \mathbf{F}^{j}\Big(\mathbf{U}^{\mathrm{R}}_{i+1/2}\Big). \tag{3.125}$$

With these states and their fluxes, the local Riemann problem is solved by using the Harten, Lax, van Leer (HLL) approximate Riemann solver (LeVeque, 2002; Toro, 2009), giving the flux across the interface

$$\Big[\mathbf{F}^{j}\Big]_{i+1/2} = \frac{s^{+}[\mathbf{F}^{j}]^{\mathrm{L}}_{i+1/2} - s^{-}[\mathbf{F}^{j}]^{\mathrm{R}}_{i+1/2} + \varepsilon s^{+}s^{-}\Big(\mathbf{U}^{\mathrm{R}}_{i+1/2} - \mathbf{U}^{\mathrm{L}}_{i+1/2}\Big)}{s^{+} - s^{-}}, \tag{3.126}$$

where the fastest characteristic speeds in the left-to-right (+) and right-to-left (-) directions are

$$\begin{aligned} s^{+} &= \max\left(\Big\{\lambda^{\pm,0}\Big\}^{\mathrm{L}}, \Big\{\lambda^{\pm,0}\Big\}^{\mathrm{R}}\right) \\ s^{-} &= \min\left(\Big\{\lambda^{\pm,0}\Big\}^{\mathrm{L}}, \Big\{\lambda^{\pm,0}\Big\}^{\mathrm{R}}\right), \end{aligned} \tag{3.127}$$

and the factor ε corrects for the optically thick limit. In this limit, the neutrinos behave diffusively and the evolution equations are no longer hyperbolic. The HLL Riemann solver fails to produce the correct form of the flux when there are large gradients between the left and right states. The correction method utilized here follows similar approaches in Skinner et al. (2019); Radice et al. (2022); Cheong et al. (2023), and decreases the contribution from the final term in Eq. (3.126) as the optical depth increases. For this purpose, the correction

factor is defined as

$$\varepsilon = \min\left(1, \frac{1}{\tau^j_{i+1/2}}\right), \tag{3.128}$$

where $\tau^j_{i+1/2}$ is the geometric average of the optical depths across each of the adjacent cells in the j-th direction

$$\tau^j_{i+1/2} = \sqrt{\tau^j_i \tau^j_{i+1}}. \tag{3.129}$$

Each cell's optical depth is approximated by

$$\tau^j_i = \left[\kappa_{\text{abs}} + \kappa_{\text{iso}}\right]_i \mathrm{d}x^j_i\,, \tag{3.130}$$

where $\mathrm{d}x^j_i$ is the proper distance across the i-th cell in the j-th direction.

Finally, the second-order approximation of the divergence of the spatial flux in can be found from the interface fluxes as

$$\left[\partial_j \mathbf{F}^j(\mathbf{U})\right]_i = \frac{\left[\mathbf{F}^j\right]_{i+1/2} - \left[\mathbf{F}^j\right]_{i-1/2}}{\left[\Delta x^j\right]_i}, \tag{3.131}$$

where $\left[\Delta x^j\right]_i$ is the width of the i-th cell in the j-th direction.

3.2.2.3 Finite-Difference Discretization

The finite-difference discretization is based on the one presented in Radice et al. (2022), and the key elements are described below. This discretization assumes that the evolved variables $\mathbf{U}_i$ are point-like values at the cell-centers. Again, the spatial advection operator is evaluated

across each cell by Eq. (3.131), but in contrast with the finite-volume discretization, the fluxes at the cell interfaces are approximated directly from the cell-centered values. While this allows for a more computationally-efficient implementation, this discretization is more dissipative than both the previous finite-volume discretization and the high-order finite-difference discretization presented in chapter 4.

Since this discretization only makes use of the cell-centered states $\mathbf{U}_i$, Eq. (3.103) is evaluated for each cell

$$\left[\mathbf{F}^j\right]_i = \mathbf{F}^j\big(\mathbf{U}_j\big), \tag{3.132}$$

along with its characteristic speeds via Eq. (3.120). These cell-centered fluxes are then used to make high-order and low-order approximations of the flux at the interface of two adjacent cells. A Lax-Friedrichs approximation is used for the low-order (LO) flux

$$\left[\mathbf{F}^j\right]^{\mathrm{LO}}_{i+1/2} = \frac{1}{2}\left(\left[\mathbf{F}^j\right]_i + \left[\mathbf{F}^j\right]_{i+1}\right) - \frac{1}{2}\lambda_{\max}\left(\mathbf{U}_{i+1} - \mathbf{U}_i\right), \tag{3.133}$$

where $\lambda_{\max}$ is the fastest characteristic speed in the adjacent cells

$$\lambda_{\max} = \max\left(\left\{\left|\lambda^{\pm,0}\right|\right\}_i, \left\{\left|\lambda^{\pm,0}\right|\right\}_{i+1}\right). \tag{3.134}$$

A second-order approximation is used for the high-order (HO) flux

$$\left[\mathbf{F}^j\right]^{\mathrm{HO}}_{i+1/2} = \frac{1}{2}\left(\left[\mathbf{F}^j\right]_i + \left[\mathbf{F}^j\right]_{i+1}\right). \tag{3.135}$$

Alternatively, higher-order approximations can be used, such as the WENO scheme discussed

in chapter 4 for the hydrodynamics solver, but will incur an increased computational cost. Finally, the flux at the interface is approximated by a variant flux-limiter approach

$$\Big[\mathbf{F}^j\Big]_{i+1/2} = \Big[\mathbf{F}^j\Big]^{\mathrm{HO}}_{i+1/2} - \varepsilon(1-\phi)\bigg(\Big[\mathbf{F}^j\Big]^{\mathrm{HO}}_{i+1/2} - \Big[\mathbf{F}^j\Big]^{\mathrm{LO}}_{i+1/2}\bigg), \tag{3.136}$$

where similarly to the finite-volume discretization, ε is the corrective factor for the optically thick limit, and ϕ is the minmod limited slope from the states in the surrounding cells. These interface fluxes are then used to evaluate the spatial advection operator via Eq. (3.131).

3.2.3 Frequency Advection

Velocity- and gravitational-dependent redshifting effects lead to a change in the number density, and consequently the energy density, of neutrinos with a frequency ν, as measured by a co-moving observer. In Eq. (3.101), these changes in frequency are described by the term $\partial_\nu \mathbf{R}(\mathbf{U})$, which characterizes advection along the ν-axis (the remaining momentum degree-of-freedom after performing the moment expansion). Advection along this ν-axis effectively couples the evolution equations for each frequency of neutrinos being evolved. This section will present a new method for discretizing this operator that aims to be more computationally- and memory-efficient to implement.

Since $\partial_\nu \mathbf{R}(\mathbf{U})$ accounts for the flux of neutrinos between different frequencies, it must vanish when integrating the evolution equations over all neutrino frequencies, i.e.

$$\int_0^\infty \mathrm{d}\nu\, \partial_\nu \mathbf{R}(\mathbf{U}) = 0, \tag{3.137}$$

in order for to conserve the neutrino number and energy densities (Müller et al., 2010). For

Eq. (3.137) to hold, $\mathbf{R}(\mathbf{U})$ must vanish at the endpoints $\nu = 0, \infty$. The discretization of the frequency-advection operator will need to enforce these boundary conditions on both $\mathbf{R}(\mathbf{U})$ and its derivative.

A common approach to discretizing the frequency-advection operator uses a flux conservative finite-volume method that breaks up the ν-axis into frequency-bins. The moment-expansion and its projection are now represented as by bin-integrated values

$$\tilde{M}^{A_k}_{(\bar{\nu}_i)} = \int_{\bar{\nu}_{i-1/2}}^{\bar{\nu}_{i+1/2}} \mathrm{d}\nu \, \tilde{M}^{A_k}_{(\nu)}, \tag{3.138}$$

where $\bar{\nu}_i$ and $\bar{\nu}_{i+1/2}$ the bin-center and bin-interface values of the frequency in the i-th bin. Simply using the bin-integrated values directly in Eq. (3.104) will not guarantee that Eq. (3.137) holds, so the monochromatic values at the bin-interfaces must be reconstructed while accounting for the boundary conditions. These interface values are typically reconstructed from both directions, and then a weighted contribution from both states is used to approximate the flux across the bin-interface. For examples of this type of flux-conservative discretization, please refer Müller et al. (2010); O'Connor (2015); Cheong et al. (2023).

While these frequency-bin discretizations can be constructed to guarantee number and energy conservation, there are drawbacks to their use. The frequency-bins are usually logarithmically spaced, and typical approximations of the bin-center values made from the bin-totals such as $\tilde{E}_{(\nu_i)} = \tilde{E}_{(\bar{\nu}_i)}/\Delta\bar{\nu}_i$ may not accurately reflect the actual bin-centered monochromatic values in larger bins. Additionally, since a infinitely-wide bin at the upper boundary is not possible, an arbitrary upper boundary where the moments are expected to be negligible must be chosen; this can be problematic if the contributions near this boundary become non-negligible during evolution. Finally, these discretizations commonly rely on

small stencils of bins when approximating the ν-derivatives, and can require a large number of bins to achieve sufficient resolution over a desired range of frequencies. Since there are four equations to solve for every neutrino frequency and species, this quickly leads to soaring computational costs as the number of evolved neutrino frequencies are increased.

Instead, this implementation will makes use of a pseudospectral discretization of the ν-axis. In a pseudospectral discretization, evolved quantities are projected on to a set orthogonal polynomial basis functions to build interpolants of these quantities and their derivatives in their original coordinate basis (Boyd, 2013). Spectral and pseudospectral methods are commonly used in numerical relativity, as they can achieve an accuracy comparable to finite-difference and finite volume methods while using a much smaller set of discretization points (Baumgarte & Shapiro, 2010). Applying a pseudospectral discretization to the frequency advection operator allows the monochromatic projected moments to be evolved directly, and quadrature rules associated with a chosen polynomial basis will provide more accurate approximations to the frequency-integrated quantities than those made with simple Riemann sums in frequency-bin discretizations.

3.2.3.1 Basis Polynomials

Choosing an appropriate set of basis polynomials is absolutely critical. Commonly used Chebyshev polynomials prove useful for almost all non-periodic problems (Boyd, 2013), but an ideal set of basis polynomials will automatically meet the boundary conditions and have a domain that easily maps to the ν-axis. Since neutrinos are fermions, one such option are the set of polynomials that are orthogonal with respect to the Fermi-Dirac distribution. This section will describe a set of Fermi-Dirac weighted orthogonal polynomials and their associated quadrature rule, based on a method for calculating moments of an electron distribution

function presented in Oettinger et al. (2013).

Let $\mathcal{F}_n(x)$ be the set of polynomials that are orthogonal with respect to the inner product

$$\langle \mathcal{F}_m | \mathcal{F}_n \rangle = \int_0^\infty \mathrm{d}x \, f_k(x) \mathcal{F}_m(x) \mathcal{F}_n(x) = D_m \delta_{mn}, \tag{3.139}$$

where D_m is a normalization constant, δ_{mn} is the Kronecker delta, and the weight $f_k(x)$ is a generalized Fermi-Dirac distribution

$$f_k(x) = \frac{x^k}{e^x + 1}. \tag{3.140}$$

The polynomials $\mathcal{F}_n(x)$ are constructed via the Gram-Schmidt process (Roman, 2007) applied to the set of polynomials $\mathcal{X}_n(x) = x^n$

$$\mathcal{F}_n(x) = \mathcal{X}_n(x) - \sum_{m=0}^{n-1} \frac{\langle \mathcal{X}_n | \mathcal{F}_m \rangle}{\langle \mathcal{F}_m | \mathcal{F}_m \rangle} \mathcal{F}_m(x). \tag{3.141}$$

These basis polynomials are then used to form an N-point quadrature rule with nodes x_i and weights w_i (Press et al., 2007). The N nodes and weights are

$$\begin{aligned} \mathcal{F}_N(x_i) &= 0 \\ w_i = \frac{1}{\mathcal{F}'_N(x_i)} \int_0^\infty \mathrm{d}x \, \frac{f_k(x) \mathcal{F}_N(x)}{x - x_i} \quad &, \qquad 0 \le i < N. \end{aligned} \tag{3.142}$$

For some function $g(x) = f_k(x) h(x)$, the quadrature rule can now be used to approximate the integrals

$$\int_0^\infty \mathrm{d}x \, g(x) = \int_0^\infty \mathrm{d}x \, f_k(x) h(x) \approx \sum_{n=0}^{N-1} w_n h(x_n) = \sum_{n=0}^{N-1} \frac{w_n}{f_k(x_n)} g(x_n), \tag{3.143}$$

where the factor $f_k(x_n)$ must be removed if using $g(x)$ directly in the summation.

3.2.3.2 Derivatives and Integrals of Frequency-Dependent Quantities

The basis polynomials are used to build an interpolating polynomial along the ν-axis. In order to make use of the quadrature rules in the previous section, the frequencies ν mapped to quadrature nodes x_i, i.e. $\nu_i = \nu(x_i)$, where in the simple case of a equilibrium distribution, the map takes the form

$$\nu(x) = Tx, \tag{3.144}$$

where T is the equilibrium temperature; this is the direct result of making the substitution $\nu/T \to x$ to arrive at Eq. (3.140).

Next, frequency-dependent quantities must be projected unto the polynomial basis. For some arbitrary frequency-dependent $M_{(\nu)}$, it is projected on to the basis polynomials via inner product (of the same form as Eq. (3.139))

$$m_i = \frac{1}{D_i}\int_0^\infty \mathrm{d}x\, f_k(x) M_{(\nu(x))}\mathcal{F}_i(x) = \frac{1}{D_i}\sum_{m=0}^{N-1} w_n M_{(\nu_n)}\mathcal{F}_i(x_n). \tag{3.145}$$

With these coefficients, the frequency-dependent quantities and their derivatives for an arbitrary value of ν are

$$M_{(\nu(x))} = \sum_{i=0}^{N-1} m_i[\mathcal{F}_i(x) f_k(x)] \tag{3.146}$$

$$\partial_\nu M_{(\nu(x))} = \frac{1}{\nu'(x)}\sum_{i=0}^{N-1} m_i[\mathcal{F}_i(x) f_k(x)]', \tag{3.147}$$

where the primed quantities are derivatives with respect to x. The x-dependence (and consequently ν-dependence) are now completely contained in the basis polynomial and weight functions and can be computed analytically.

When working strictly with frequency-dependent quantities located at the N quadrature nodes, all the of x-dependent quantities can be precomputed, and calculating Eqns. (3.145)–(3.147) for all N points reduces to matrix-vector multiplications. For this case, the frequency-integrated quantities reduce to the vector dot-product

$$M = \int_0^\infty \mathrm{d}\nu \, M_{(\nu)} = \int_0^\infty \mathrm{d}x \, \nu'(x) M_{(\nu)} = \sum_{i=0}^{N-1} \frac{w_i}{f_k(x_i)} \nu'(x_i) M_{(\nu_i)}. \tag{3.148}$$

In application to the frequency advection operator, $\mathbf{R}(\mathbf{U})$ is computed first for each neutrino frequency ν_i. Next, these values are projected on to the basis functions via Eq. (3.145). Finally, Eq. (3.147) is used to evaluate the full operator $\partial_\nu \mathbf{R}(\mathbf{U})$ for all frequencies ν_i. When frequency-integrated quantities are required, e.g. for frequency-integrated full collision source term in Eq. (3.79), or for outputting the frequency-integrated projected moments, Eq. (3.148) is used directly.

3.2.4 Geometric Sources

The geometric source terms $\mathbf{G}(\mathbf{U})$ are evaluated directly in terms of the cell-centered neutrino and spacetime variables. Computation of the spatial derivatives of the spacetime quantities all use a centered fourth-order finite difference operator

$$\frac{\partial g}{\partial x^j} \approx \frac{1}{12\Delta x^j}[g_{i-2} - 8g_{i-1} + 8g_{i+1} - g_{i+2}], \tag{3.149}$$

where g is replaced with each the lapse α, the shift components β^i, and the spatial metric components γ_{ij}. This operator is applied separately along each direction j. These same derivatives, along with the derivatives of the spatial velocity and Lorentz factor, present in the frequency advection terms are also computed using Eq. (3.149). While the spatial accuracy for both the finite-volume and finite-difference discretizations is only second-order, this fourth-order approximation for the derivatives spacetime quantities mirrors the operators used in the spacetime solvers described in chapter 5.

3.2.5 Time Integration

The disparate timescales associated with each of the operators in Eq. (3.101) complicate the time integration of the evolution equations. The advective timescale, which is at worst the light-crossing time of the smallest cell, can be used to stably evolve most terms in Eq. (3.101) with explicit time integration methods. However, the timescales of the neutrino-matter interactions are $\mathcal{O}\big(1/\kappa_{\mathrm{ea,iso}}\big)$ (Burrows et al., 2006), which quickly become orders of magnitude smaller than the advective timescale with increasing optical depths. These timescales cannot be stably evolved with explicit methods for any practical size of the time-step. While fully-implicit time integration methods can be used to stably evolve Eq. (3.101) for a more-practical choice of time-step, the computational and memory costs rapidly become prohibitive for large numbers of neutrino species and frequencies since the spatial- and frequency-advection operators are non-local along their respective axes.

Instead, Eq. (3.101) will be integrated using a hybrid implicit-explicit (IMEX) method in a method-of-lines (MoL) discretization (see chapter 6 for more about the time-integrators). In this method, an explicit integration method applied to the advective and geometric sources couples to an implicit integration method used for the neutrino-matter interactions. Extend-

ing the notation in Eq. (3.101) as $\mathbf{U}^n$ to now include a superscript n referring to the state at the n-th time-step, the evolution equation is written in the form

$$\begin{aligned} \partial_t \mathbf{U}^{n\to n+1} &= \mathbb{F}_{\mathrm{E}}\Big(\mathbf{U}^n\Big) + \mathbb{F}_{\mathrm{I}}\Big(\mathbf{U}^{n+1}\Big) \\ \mathbb{F}_{\mathrm{E}}\Big(\mathbf{U}^n\Big) = -\partial_j \mathbf{F}^j\Big(\mathbf{U}^n\Big) - \partial_\nu \mathbf{R}\Big(\mathbf{U}^n\Big) + \mathbf{G}\Big(\mathbf{U}^n\Big), &\qquad \mathbb{F}_{\mathrm{I}}\Big(\mathbf{U}^{n+1}\Big) = \mathbf{S}\Big(\mathbf{U}^{n+1}\Big), \end{aligned} \tag{3.150}$$

for advancing the evolved variables from $\mathbf{U}^n$ to $\mathbf{U}^{n+1}$, where $\mathbb{F}_{\mathrm{E}}$ and $\mathbb{F}_{\mathrm{I}}$ are used to update the evolved variables explicitly and implicitly, respectively.

The interactions sources are updated implicitly during each stage of time integration. Since the included interactions described earlier in the chapter do not couple the neutrinos between frequencies, the implicit-update equation takes a block-diagonal matrix form, where there is one 4×4 block for each neutrino species and frequency. This greatly simplifies the update procedure, allowing each block in the equation to be solved separately. As these terms are highly non-linear due to the closure interpolation, they cannot be directly inverted, so Newton-Raphson iterations are used to find the root of the function

$$\mathcal{G}\Big(\mathbf{U}^{\mathbf{n+1}}\Big) = \mathbf{U}^{n+1} - \Big[\mathbf{U}^n + \Delta t \mathbb{F}_{\mathrm{E}}\Big(\mathbf{U}^n\Big)\Big] - \Delta t \mathbb{F}_{\mathrm{I}}\Big(\mathbf{U}^{n+1}\Big) = 0. \tag{3.151}$$

While the preceding equation is for the full time-step, this procedure takes the same form for finding the intermediate states during every integration stage. The incoming state (the terms in the brackets), is used as the initial guess. The Newton-Raphson iterations then take the form (Press et al., 2007)

$$\mathbf{U}^{\mathbf{k+1}} = \mathbf{U}^{\mathbf{k}} - \Big[\mathcal{G}'\Big(\mathbf{U}^{\mathbf{k}}\Big)\Big]^{-1} \mathcal{G}\Big(\mathbf{U}^{\mathbf{k}}\Big), \tag{3.152}$$

where $\mathcal{G}'$ is the jacobian of Eq. (3.151)

$$\mathcal{G}'(\mathbf{U}) = \mathbf{I} - \Delta t \frac{\partial \mathbb{F}_{\mathrm{I}}}{\partial \mathbf{U}}, \tag{3.153}$$

where $\mathbf{I}$ is the identity matrix. See appendix F for the full form of the derivatives of the interaction source terms in Eq. (3.153).

A common approach to simplifying the implicit update procedure is linearizing the interaction source terms, allowing for a direct inversion of Eq. (3.151). This can be accomplished by by assuming the direction of the momentum remains constant, i.e. $\hat{f}^i = \text{const.}$ or $v_k F^k = \text{const.}$, or that the closure does not change as a result of the implicit update, i.e. $\xi = \text{const.}$. See Foucart et al. (2015); Weih et al. (2020) for examples of these linearizations. However, these types of approximations can be problematic for maintaining energy and momentum conservation (Radice et al., 2022). Instead, following the methods in Radice et al. (2022); Cheong et al. (2023), the closure is computed for every iteration and all non-linear terms in the interaction sources are retained.

Chapter 4

General Relativistic Hydrodynamics

The general relativistic neutrino radiation transport solver presented in chapter 3 is of limited use on its own when applied to problems in vacuum or with fixed-background fluids. Coupling the evolution of the neutrinos with the fluid requires a compatible hydrodynamics formulation in order to enforce energy and momentum conservation and maintain consistency and stability. In the dynamic curved spacetimes present during neutron star mergers and near the compact objects formed during core-collapse supernovae, the effects of general relativity on the evolution of the fluid and neutrinos cannot be ignored. This chapter will present the formulation of and the numerical methods used for a general relativistic hydrodynamics solver, and its implementation will be presented later in chapter 6.

4.1 Mathematical Formalism

For the method presented here, only an ideal fluid will be considered, with the stress-energy tensor defined in the fluid's restframe

$$T^{ab} = \rho h u^a u^b + p g^{ab}, \tag{4.1}$$

where u^a is the four-velocity of a co-moving observer, g^{ab} the full spacetime metric, ρ and p are the fluid density and pressure, respectively, and the specific enthalpy is

$$h = 1 + \epsilon + \frac{p}{\rho} = \frac{e + p}{\rho}, \tag{4.2}$$

where $e = \rho(1+\epsilon)$ is the fluid energy density accounting both for the rest-mass energy and the internal energy ϵ. Similarly to the neutrino radiation transport formulation in chapter 3, it will be necessary to define the stress-energy tensor in the Eulerian frame of the 3+1 split of the spacetime. In this frame, the stress-energy tensor takes the form

$$T^{ab} = \mathcal{E} n^a n^b + \mathcal{S}^a n^b + \mathcal{S}^b n^a + \mathcal{S}^{ab}, \tag{4.3}$$

where n^a is the four-velocity of the Eulerian observer, or equivalently the normal vector to each spacelike hypersurface in the spacetime decomposition. In analogy to the neutrino moment-expansion projected moments, the Eulerian projections of the fluid stress-energy tensor are found as

$$\mathcal{E} = n_a n_b T^{ab} \quad = \rho h W^2 - p \tag{4.4}$$

$$\mathcal{S}^i = -\gamma^i{}_a n_b T^{ab} = \rho h W^2 v^i \tag{4.5}$$

$$\mathcal{S}^{ij} = \gamma^i{}_a \gamma^j{}_b T^{ab} \; = \rho h W^2 v^i v^j + p\gamma^{ij}, \tag{4.6}$$

where v^i is the fluid velocity with Lorentz factor $W = 1/\sqrt{1 - v_i v^i}$ – these result from using the Eulerian decomposition of the four-velocity in Eq. (3.36).

Energy and momentum conservation are guaranteed when the total stress-energy tensor of the fluid and neutrinos is divergence free,

$$\boldsymbol{\nabla}_b \left[T^{ab} + \sum_\sigma M^{ab}_{(\sigma)} \right] = 0, \tag{4.7}$$

where $\boldsymbol{\nabla}_b$ is the covariant derivative operator, and the summation is over the frequency-

integrated neutrino stress-energy tensor for all neutrino species. In comparison with the frequency-integrated second-rank moment expansion equations in Eq. (3.13), it follows that the fluid energy and momentum densities must obey

$$\boldsymbol{\nabla}_b T^{ab} = -S^a_{\text{tot.}}, \tag{4.8}$$

where $S^a_{\text{tot.}}$ is the full sum-over-species integrated-over-frequencies neutrino-matter interaction source term defined in Eq. (3.79). Mass conservation is enforced by requiring the rest-mass density along the trajectory of the co-moving observer to remain divergence-free

$$\boldsymbol{\nabla}_a(\rho u^a) = 0. \tag{4.9}$$

This equation is commonly referred to the mass-current continuity equation.

The evolution equations for the energy and momentum densities result from the timelike and spacelike projections of Eq. (4.8) and take the exact same form as the neutrino moment evolution equations

$$\frac{\partial \tilde{\mathcal{E}}}{\partial t} + \frac{\partial}{\partial x^j}\Big(\alpha \tilde{\mathcal{S}}^j - \beta^j \tilde{\mathcal{E}}\Big) = \alpha\left[\tilde{\mathcal{S}}^{ij} K_{ij} - \tilde{\mathcal{S}}^i \frac{\partial \ln \alpha}{\partial x^i} + n_a \tilde{S}^a_{\text{tot.}}\right] \tag{4.10}$$

$$\frac{\partial \tilde{\mathcal{S}}_i}{\partial t} + \frac{\partial}{\partial x^j}\Big(\alpha \tilde{\mathcal{S}}^j{}_i - \beta^j \tilde{\mathcal{S}}_i\Big) = \alpha\left[\frac{1}{2}\tilde{\mathcal{S}}^{jk}\frac{\partial \gamma_{jk}}{\partial x^i} - \frac{\tilde{\mathcal{S}}_j}{\alpha}\frac{\partial \beta^j}{\partial x^i} - \tilde{\mathcal{E}}\frac{\partial \ln \alpha}{\partial x^i} - \gamma_{ia}\tilde{S}^a_{\text{tot.}}\right]. \tag{4.11}$$

The evolution equation for the rest-mass density follows directly from using the Eulerian decomposition of the co-moving four-velocity, Eq. (3.36), in Eq. (4.9) to find

$$\frac{\partial \tilde{\mathcal{D}}}{\partial t} + \frac{\partial}{\partial x^j}\Big[\tilde{\mathcal{D}}\Big(\alpha v^j - \beta^j\Big)\Big] = 0, \tag{4.12}$$

where $\mathcal{D} = \rho W$ is the rest-mass density measured in the Eulerian frame. In the preceding evolution equations, all tilde-quantities include the densitization factor $\sqrt{\gamma}$, where $\gamma = \det \gamma_{ij}$. A common modification made to the energy density evolution equation to facilitate numerical implementations is to separate the contribution from the rest-mass energy by defining the new variable $\tau = \mathcal{E} - \mathcal{D}$ (Rezzolla & Zanotti, 2013). Substituting this new variable into Eq. (4.10) and using Eq. (4.12) to eliminate the terms proportional to $\mathcal{D}$ gives the modified energy density evolution equation

$$\frac{\partial \tilde{\tau}}{\partial t} + \frac{\partial}{\partial x^j}\left[\alpha\left(\tilde{\mathcal{S}}^j - \tilde{\mathcal{D}}v^j\right) - \beta^j \tilde{\tau}\right] = \alpha\left[\tilde{\mathcal{S}}^{ij} K_{ij} - \tilde{\mathcal{S}}^i \frac{\partial \ln \alpha}{\partial x^i} + n_a \tilde{S}^a_{\rm tot.}\right]. \tag{4.13}$$

This flux-conservative form of the evolution equations in Eqns. (4.11)–(4.13) without the neutrino-matter interaction sources terms is commonly referred to as the Valencia formulation and was first proposed in Banyuls et al. (1997).

4.2 Numerical Methods

This section will detail the high-order finite-difference discretization used to solve the fluid evolution equations. A similar notation to the one used in chapter 3 will be used to separate Eqns. (4.11)–(4.13) into an operator form

$$\partial_t \mathsf{U} + \partial_j \mathsf{F}^j(\mathsf{U}) = \mathsf{G}(\mathsf{U}) + \mathsf{S}(\mathsf{U}), \tag{4.14}$$

where the vector of evolved conserved variables is

$$\mathsf{U} = \begin{bmatrix} \tilde{\mathcal{D}} \\ \tilde{\mathcal{S}}_j \\ \tilde{\tau} \end{bmatrix}, \tag{4.15}$$

the flux vector in the spatial advection operator for the j-th direction is

$$\mathsf{F}^{\mathsf{j}}(\mathsf{U}) = \begin{bmatrix} \tilde{\mathcal{D}}\big(\alpha v^j - \beta^j\big) \\ \alpha \tilde{\mathcal{S}}^j{}_i - \beta^j \tilde{\mathcal{S}}_i \\ \alpha\Big(\tilde{\mathcal{S}}^j - \tilde{\mathcal{D}} v^j\Big) - \beta^j \tilde{\tau} \end{bmatrix}, \tag{4.16}$$

the geometric source vector is

$$\mathsf{G}(\mathsf{U}) = \alpha \begin{bmatrix} 0 \\ \frac{1}{2}\tilde{\mathcal{S}}^{jk}\partial_i \gamma_{jk} - \alpha^{-1}\tilde{\mathcal{S}}_j \partial_i \beta^j - \tilde{\mathcal{E}}\partial_i \ln\alpha \\ \tilde{\mathcal{S}}^{ij} K_{ij} - \tilde{\mathcal{S}}^i \partial_i \ln\alpha \end{bmatrix}, \tag{4.17}$$

and the neutrino-matter interaction source vector is

$$\mathsf{S}(\mathsf{U}) = \alpha \begin{bmatrix} 0 \\ -\gamma_{ia}\tilde{S}^a_{\text{tot.}} \\ n_a \tilde{S}^a_{\text{tot.}} \end{bmatrix}. \tag{4.18}$$

The source terms are treated similarly to those in the neutrino moment evolution equations, so will only briefly be described here. The geometric sources take the exact same form as those in the neutrino moment evolution equations, and are calculated in the same way as

described in chapter 3. The neutrino-matter interaction sources will be handled in a simplified manner: the opacities characterizing the interactions will be held fixed when implicitly updating the evolved neutrino variables, and then will subsequently be used to explicitly update the fluid variables. This method is commonly used in radiation transport solvers and works well so long as the fluid does not rapidly change over the course of a single time-step; see Foucart et al. (2015); O'Connor (2015); Radice et al. (2022); Cheong et al. (2023) for examples of this method. Implicitly updating the fluid variables alongside the neutrino variables would fully-couple all neutrino frequencies, vastly increasing the computational cost by requiring an iterative solution of the much-larger $\left(5 + 4 \times N_{\text{freq}} \times N_{\text{species}}\right) \times \left(5 + 4 \times N_{\text{freq}} \times N_{\text{species}}\right)$ matrix equation (which is no longer strictly block-diagonal) for the full fluid plus neutrino system of evolution equations.

The remainder of this section will be dedicated to describing the eigenstructure and the discretization of the spatial advection operator, as well as a method used to recover the primitive variables, such as the density, pressure, and velocity. Similarly to the presentation of the discretization of the neutrino moment evolution equations, the following will presented for a single spatial direction, but generalize to all spatial directions. Quantities will be labeled with a subscript i will refer to cell-centered values in the i-th cell along the chosen direction, with cell-interface values labeled by $i \pm 1/2$ for the upper (+) and lower (-) faces of the cell.

4.2.1 Eigenstructure

The eigenstructure of the spatial advection operator is used to determine its characteristic trajectories and speeds. The speeds are immediately useful for determining the maximum stable time-step size, although the neutrino characteristic speeds typically require a smaller time-step size. The characteristic trajectories provide a decoupled form of the advection

operator that can robustly capture shocks that form in the fluid. To examine this eigenstructure, the spatial advection operator in the i-th direction will first be written as

$$\partial_i \mathsf{F}(\mathsf{U}) = \frac{\partial \mathsf{F}^i}{\partial \mathsf{U}} \partial_i \mathsf{U} = \mathsf{A}^i \partial_i \mathsf{U}, \tag{4.19}$$

where the flux operator A^i is the jacobian of the spatial flux vector F^i with each row corresponding to the elements of F^i and each column the derivative with respect to elements of U. The flux operator can be diagonalized by its eigendecomposition in the form

$$\mathsf{A}^i = \mathsf{R}^{-1} \mathsf{\Lambda} \mathsf{R}, \tag{4.20}$$

where $\mathsf{\Lambda}$ is a diagonal matrix of the eigenvalues of A^i, and R is the matrix of right-eigenvectors, with each column corresponding representing the eigenvector associated with the same column in $\mathsf{\Lambda}$.

The characteristic speeds for the evolution equations are simply the eigenvalues of A^i. For the Valencia formulation, the unique eigenvalues associated with the flux in the i-th direction are $\lambda_\pm$ which characterizes acoustic waves, and the triply-degenerate λ_0 which characterizes matter waves (Banyuls et al., 1997)

$$\lambda_\pm = -\beta^i + \frac{\left(1 - c_s^2\right) p^i \pm c_s \sqrt{\left(1 - v^2\right)\left[\alpha^2 \left(1 - v^2 c_s^2\right) \gamma^{ii} - \left(1 - c_s^2\right) p^i p^i\right]}}{1 - v^2 c_s^2} \tag{4.21}$$

$$\lambda_0 = -\beta^i + p^i, \tag{4.22}$$

where $p^i = \alpha v^i$, and c_s is the relativistic sound speed of the fluid defined in terms of the of

the relation

$$hc_s^2 = \left(\frac{\partial p}{\partial \rho}\right)_\epsilon + \frac{p}{\rho^2}\left(\frac{\partial p}{\partial \epsilon}\right)_\rho. \tag{4.23}$$

In normal ordering, the eigenvalue matrix is

$$\Lambda = \mathrm{diag}(\lambda_-,\, \lambda_0,\, \lambda_0,\, \lambda_0,\, \lambda_+). \tag{4.24}$$

The analytic forms of the eigenvectors for the Valencia formulation are also known, and will prove useful in projecting the evolved variables and their fluxes on to their characteristic trajectories. The right-eigenvectors for this system are presented in Ibanez et al. (1999); Rezzolla & Zanotti (2013) and are reproduced here for reference. For the i-th direction, the components of the eigenvectors $\mathbf{r}^{(\pm)} = \left[r_0^{(\pm)}, r_1^{(\pm)}, r_2^{(\pm)}, r_3^{(\pm)}, r_4^{(\pm)}\right]^{\mathrm{T}}$ associated with $\lambda_\pm$ are:

$$\begin{aligned} r_0^{(\pm)} &= 1 \\ r_i^{(\pm)} &= hW\mathcal{V}_\pm^i \\ r_j^{(\pm)} &= hWv_j \\ r_k^{(\pm)} &= hWv_k \\ r_4^{(\pm)} &= hW\mathcal{A}_\pm^i - 1 \end{aligned} \tag{4.25}$$

with

$$\mathcal{V}_\pm^i = \frac{v^i - \Lambda_\pm^i}{\gamma^{ii} - v^i\Lambda_\pm^i}, \qquad \mathcal{A}_\pm^i = \frac{\gamma^{ii} - v^iv^i}{\gamma^{ii} - v^i\Lambda_\pm^i}, \qquad \Lambda_\pm^i = \frac{\lambda_\pm + \beta^i}{\alpha}, \tag{4.26}$$

and the eigenvectors $\mathbf{r}^{(0,n)} = \left[r_0^{(0,n)}, r_1^{(0,n)}, r_2^{(0,n)}, r_3^{(0,n)}, r_4^{(0,n)}\right]^{\mathrm{T}}$ associated with λ_0 are

$$
\begin{aligned}
r_0^{(0,1)} &= \frac{\mathcal{K}}{hW}, & r_0^{(0,2)} &= W v_j, & r_0^{(0,3)} &= W v_k \\
r_i^{(0,1)} &= v_i, & r_i^{(0,2)} &= h\Big(\gamma_{ij} + 2W^2 v_i v_j\Big), & r_i^{(0,3)} &= h\Big(\gamma_{ik} + 2W^2 v_i v_k\Big) \\
r_j^{(0,1)} &= v_j, & r_j^{(0,2)} &= h\Big(\gamma_{jj} + 2W^2 v_j v_j\Big), & r_j^{(0,3)} &= h\Big(\gamma_{jk} + 2W^2 v_j v_k\Big) \\
r_k^{(0,1)} &= v_k, & r_k^{(0,2)} &= h\Big(\gamma_{kj} + 2W^2 v_k v_j\Big), & r_k^{(0,3)} &= h\Big(\gamma_{kk} + 2W^2 v_k v_k\Big) \\
r_4^{(0,1)} &= 1 - \frac{\mathcal{K}}{hW}, & r_4^{(0,2)} &= W v_j (2hW - 1), & r_4^{(0,3)} &= W v_k (2hW - 1),
\end{aligned}
\tag{4.27}
$$

where

$$
\mathcal{K} = \frac{\left(\partial p/\partial \epsilon\right)_\rho}{\left(\partial p/\partial \epsilon\right)_\rho - \rho c_s^2}. \tag{4.28}
$$

The explicit component notation for the five components of each eigenvector allows a straightforward specification for each spatial dimension, where the i, j, k components are easily permuted for different directions. For each direction $i = 1, 2, 3$, the corresponding values of i, j, k for the components and their ordering take on the cyclic permutations of $i, j, k = 1, 2, 3$. With these, the right-eigenvector matrix corresponding the the eigenvalue matrix $\mathsf{\Lambda}$ in Eq. (4.24) takes the form

$$
\mathsf{R} = \Big[\mathbf{r}^{(-)},\, \mathbf{r}^{(0,1)},\, \mathbf{r}^{(0,2)},\, \mathbf{r}^{(0,3)},\, \mathbf{r}^{(+)}\Big]. \tag{4.29}
$$

The inverse of the right-eigenvector matrix, also referred to as the left-eigenvector matrix, has rows that correspond to the left-eigenvector for the eigenvalue in the same row in $\mathsf{\Lambda}$. These also have known analytic forms for the Valencia formulation (Ibanez et al., 1999; Rezzolla &

Zanotti, 2013), and are reproduced here for reference. Using a similar component notation, the left-eigenvectors associated with $\lambda_\pm$ in the i-th direction are

$$
\begin{aligned}
l_0^{(\pm)} &= hW\zeta\mathcal{V}_\mp^i + l_4^{(\pm)} \\
l_i^{(\pm)} &= \Big(1-\mathcal{K}\mathcal{A}_\mp^i\Big)\Gamma_{ii} + \Big(2\mathcal{K}-1\Big)\Big(W^2\zeta v^i - \Gamma_{ii}v^i\Big)\mathcal{V}_\mp^i \\
l_j^{(\pm)} &= \Big(1-\mathcal{K}\mathcal{A}_\mp^i\Big)\Gamma_{ij} + \Big(2\mathcal{K}-1\Big)\Big(W^2\zeta v^j - \Gamma_{ij}v^i\Big)\mathcal{V}_\mp^i \\
l_k^{(\pm)} &= \Big(1-\mathcal{K}\mathcal{A}_\mp^i\Big)\Gamma_{ik} + \Big(2\mathcal{K}-1\Big)\Big(W^2\zeta v^k - \Gamma_{ik}v^i\Big)\mathcal{V}_\mp^i \\
l_4^{(\pm)} &= \Big(\mathcal{K}-1\Big)\Big[\Big(W^2\zeta-\Gamma_{ii}\Big)\mathcal{V}_\mp^i - \gamma v^i\Big] - \mathcal{K}W^2\zeta\mathcal{V}_\mp^i
\end{aligned}
\tag{4.30}
$$

$$
\mathbf{l}^{(\pm)} = \mp\frac{h^2}{\Delta}\Big[l_0^{(\pm)},\, l_1^{(\pm)},\, l_2^{(\pm)},\, l_3^{(\pm)},\, l_4^{(\pm)}\Big], \tag{4.31}
$$

and the left-eigenvectors associated with the triply-degenerate λ_0 in the i-th direction are

$$
\begin{aligned}
&l_0^{(0,1)} = h - W, && l_0^{(0,2)} = \gamma_{jk}v_k - \gamma_{kk}v_j, && l_0^{(0,3)} = \gamma_{kj}v_j - \gamma_{jj}v_k \\
&l_i^{(0,1)} = Wv^i, && l_i^{(0,2)} = \Big(\gamma_{kk}v_j - \gamma_{jk}v_k\Big)v^i, && l_i^{(0,3)} = \Big(\gamma_{jj}v_k - \gamma_{kj}v_j\Big)v^i \\
&l_j^{(0,1)} = Wv^j, && l_j^{(0,2)} = \mathcal{B}^i\gamma_{kk} + \gamma_{ik}v_kv^i, && l_j^{(0,3)} = -\mathcal{B}^i - \gamma_{ij}v_kv^i \\
&l_k^{(0,1)} = Wv^k, && l_k^{(0,2)} = -\mathcal{B}^i - \gamma_{ik}v_jv^i, && l_k^{(0,3)} = \mathcal{B}^i\gamma_{jj} + \gamma_{ij}v_jv^i \\
&l_4^{(0,1)} = -W, && l_4^{(0,2)} = l_0^{(0,2)}, && l_4^{(0,3)} = l_0^{(0,3)},
\end{aligned}
\tag{4.32}
$$

$$
\mathbf{l}^{(0,1)} = \frac{W}{\mathcal{K}-1}\Big[l_0^{(0,1)},\, l_1^{(0,1)},\, l_2^{(0,1)},\, l_3^{(0,1)},\, l_4^{(0,1)}\Big] \tag{4.33}
$$

$$
\mathbf{l}^{(0,2)} = \frac{1}{h\zeta}\Big[l_0^{(0,2)},\, l_1^{(0,2)},\, l_2^{(0,2)},\, l_3^{(0,2)},\, l_4^{(0,n)}\Big] \tag{4.34}
$$

$$
\mathbf{l}^{(0,3)} = \frac{1}{h\zeta}\Big[l_0^{(0,3)},\, l_1^{(0,3)},\, l_2^{(0,3)},\, l_3^{(0,3)},\, l_4^{(0,3)}\Big], \tag{4.35}
$$

where γ is the determinant of the spatial metric, and

$$\Gamma_{ii} = \gamma_{jj}\gamma_{kk} - \gamma_{jk}\gamma_{jk}, \quad \Gamma_{ij} = \gamma_{ik}\gamma_{jk} - \gamma_{ij}\gamma_{kk}, \quad \Gamma_{ik} = \gamma_{ij}\gamma_{jk} - \gamma_{jj}\gamma_{ik},$$
$$\zeta = \Gamma_{ii} - \gamma v^i v^i, \quad \Delta = h^3 W \zeta \Big(\mathcal{K} - 1\Big)\Big(\mathcal{C}^i_+ - \mathcal{C}^i_-\Big), \tag{4.36}$$
$$\mathcal{B}^i = 1 - v_i v^i, \quad \mathcal{C}^i_\pm = v^i - \Lambda^i_\pm.$$

At times it will be necessary to renormalize the right-eigenvectors, e.g. for a barotropic equation of state the parameter $\mathcal{K}$ in Eq. (4.28) is singular since $c_s^2 = (1/\rho)(\,\partial p/\partial \epsilon\,)_\rho$ (Rezzolla & Zanotti, 2013), and the left-eigenvectors are no longer given by the analytic expressions in Eq. (4.31) and Eqns. (4.33)–(4.35). In these cases, the left-eigenvectors are obtained from the numerical inverse of the renormalized right-eigenvector matrix.

4.2.2 Spatial Advection

In the finite-difference discretization, the evolved conserved fluid variables are represented by the point-like values located at the cell-centers in the mesh. The spatial advection operator, $\partial_j \mathsf{F}^j(\mathsf{U})$, is evaluated by the difference of the spatial fluxes approximated at the interfaces of each cell. These interface fluxes are reconstructed from the characteristic projections of the variables and their spatial fluxes. Typically, finite-difference methods applied to hydrodynamics are unable to capture shocks that form in the solution without either using a finite-volume type Riemann solver to approximate the interface fluxes, or by attempting to smooth discontinuities by methods such as artificial viscosity. However, when evaluating the spatial advection operator along the characteristic trajectories that the fluid evolves along, shock and rarefaction waves that form in the solution can be resolved without relying on more approximate methods.

Following a similar method presented in Rezzolla & Zanotti (2013); Radice et al. (2014), the right- and left-eigenvector matrices are first computed at each interface using an average of the adjacent cells

$$\bar{\mathsf{R}} \equiv \mathsf{R}\Big(\bar{\mathsf{U}}_{i+1/2}\Big), \quad \bar{\mathsf{L}} \equiv \bar{\mathsf{R}}^{-1} \tag{4.37}$$

$$\bar{\mathsf{U}}_{i+1/2} = \frac{1}{2}\Big(\mathsf{U}_i + \mathsf{U}_{i+1}\Big). \tag{4.38}$$

Next, the cell-centered conserved variables U_i and their spatial fluxes $\mathsf{F}_i \equiv \mathsf{F}^j(\mathsf{U}_i)$, where i is limited to the range of the local reconstruction stencil $\mathbb{S}$, are projected on to their characteristic trajectories via the eigenvector matrices (Rezzolla & Zanotti, 2013)

$$\mathsf{W}_i = \bar{\mathsf{L}}\mathsf{U}_i \tag{4.39}$$

$$\mathsf{Q}_i = \bar{\mathsf{L}}\mathsf{F}_i. \tag{4.40}$$

Then, a Lax-Friedrichs flux-splitting separates the characteristic fluxes into their rightward (+) and leftward (-) components (Radice et al., 2014)

$$\mathsf{Q}_i^{\pm} = \mathsf{Q}_i \pm c\mathsf{U}_i, \tag{4.41}$$

where

$$c = \max_{i\in\mathbb{S}} \left|\lambda_i^{\{\pm,0\}}\right| \tag{4.42}$$

is the fastest characteristic speed in the stencil $\mathbb{S}$.

A weighted essentially non-oscillatory (WENO) method directly reconstructs the split

characteristic fluxes at each interface. The specific reconstruction scheme here is based on the fifth-order method presented in Jiang & Shu (1996) and a symmetric variant presented in Martín et al. (2006). Fifth-order WENO methods are commonly used for both finite-volume and finite-difference discretizations in general relativistic hydrodynamics; see Mösta et al. (2014) and Radice et al. (2014) for examples of each, respectively. In a finite-difference discretization, WENO methods use weighted combinations of the reconstructed flux from different stencils surrounding the interface. For fifth-order methods, each stencil consists of three cells surrounding the interface that is being reconstructed. The standard method presented in Jiang & Shu (1996) uses three sets of stencils biased in the upwind direction, while the symmetric variant in Martín et al. (2006) adds a fourth stencil such that there are an equal number of cells to the left- and right-sides of the interface are present in the combined overall stencil.

The following is adapted from the symmetric method presented in Martín et al. (2006); this also includes the standard method when working with the reduced set of stencils. Let the full reconstruction stencil for the $i + 1/2$ interface be

$$\begin{aligned} \mathbb{S} &= \{\mathbb{S}_0, \mathbb{S}_1, \mathbb{S}_2, \mathbb{S}_3\} \\ \mathbb{S}_k &= \{i + k - 2, i + k - 1, i + k\}. \end{aligned} \tag{4.43}$$

The split characteristic fluxes are interpolated to the $i + 1/2$ interface over each stencil as

$$q_k^{\pm} = \sum_{l=0}^{2} a_{kl} \mathsf{Q}^{\pm}_{i+k+l-2}, \tag{4.44}$$

where $a_{k,l}$ are the interpolating coefficients for the k-th stencil. The weighted approximations

at the interface are then

$$\mathsf{Q}^{\pm}_{i+1/2} = \sum_{k=0}^{3} \omega_k q_k^{\pm}, \tag{4.45}$$

where the weights ω_k associated with each stencil are the normalized non-oscillatory weights

$$\omega_k = \frac{\varpi_k}{\sum_{l=0}^{3} \varpi_l}. \tag{4.46}$$

The non-oscillatory weights are

$$\varpi_k = \frac{C_k}{\mathrm{IS}_k + \epsilon}, \tag{4.47}$$

where ϵ is a small-parameter to avoid division by zero, C_k are the optimal weights, and IS_k are the oscillation indicators

$$\mathrm{IS}_k = \sum_{m=1}^{2} \left(\sum_{l=0}^{3} d_{kl}^{m} \mathsf{Q}^{\pm}_{i+k+l-2} \right)^2. \tag{4.48}$$

For the symmetric set of stencils used here, the final stencil $\mathbb{S}_3$ is fully downwind of the $i+1/2$ interface, so the final stencil's oscillation indicator is set to the maximum of all values following the procedure in Martín et al. (2006)

$$\mathrm{IS}_3 = \max_{0 \le k \le 3} \mathrm{IS}_k. \tag{4.49}$$

The coefficients a_{kl}, d_{kl}^{m}, and optimal weights C_k are from the fifth-order WENO-SYMOO

scheme in Martín et al. (2006) and are provided here for reference:

$$a_{kl} = \begin{bmatrix} \frac{1}{3} & -\frac{7}{6} & \frac{11}{6} \\ -\frac{1}{6} & \frac{5}{6} & \frac{1}{3} \\ \frac{1}{3} & \frac{5}{6} & -\frac{1}{6} \\ \frac{11}{6} & -\frac{7}{6} & \frac{1}{3} \end{bmatrix}, \quad d^1_{kl} = \begin{bmatrix} \frac{1}{2} & -2 & \frac{3}{2} \\ -\frac{1}{2} & 0 & \frac{1}{2} \\ -\frac{3}{2} & 2 & -\frac{1}{2} \\ -\frac{5}{2} & 4 & -\frac{3}{2} \end{bmatrix}, \quad d^2_{kl} = \begin{bmatrix} b & -2b & b \\ b & -2b & b \\ b & -2b & b \\ b & -2b & b \end{bmatrix}, \tag{4.50}$$

where $b = \sqrt{13/12}$, and

$$C_k = \left[\frac{1}{20}, \frac{9}{20}, \frac{9}{20}, \frac{1}{20}\right]. \tag{4.51}$$

The reconstructed split characteristic fluxes at the interfaces are then used to approximate the unsplit characteristic flux as

$$\mathsf{Q}_{i+1/2} = \mathsf{Q}^+_{i+1/2} + \mathsf{Q}^-_{i+1/2}. \tag{4.52}$$

Next, these are projected back to the spatial fluxes at the interface via the right-eigenvector matrix

$$\mathsf{F}^j_{i+1/2} = \bar{\mathsf{R}}\mathsf{Q}_{i+1/2}. \tag{4.53}$$

Finally, the spatial advection operator for the j-th direction is evaluated in the i-th cell by using the reconstructed interface fluxes as

$$\partial_j \mathsf{F}^j(\mathsf{U}_i) = \frac{\mathsf{F}^j_{i+1/2} - \mathsf{F}^j_{i-1/2}}{\Delta x}, \tag{4.54}$$

where Δx is the width of the cell.

4.2.3 Recovering the Primitive Variables

Similarly to the neutrino moment evolution equations, the fluid evolution equations must also be closed by specifying the pressure. While the pressure is readily obtained from an equation of state $p(\rho, \epsilon)$, the primitive variables necessary for evaluating the equation of state and the evolution equations, such as the rest-mass density ρ, internal energy ϵ, and the fluid velocity v^i, are not as straightforward to obtain. They must be inverted from the conserved variables, Eq. (4.15), which is not a simple algebraic process in general relativistic hydrodynamics like it is in its Newtonian formulation. The conserved-to-primitive conversion is complicated by the presence of the Lorentz factor W which leads to a highly non-linear relation between these sets of variables. As such, the only effective means of recovering the primitives is through numerical root-finding algorithms.

A one-dimensional bracketed root-finding procedure is used to recover the primitive variables

$$\mathsf{V} = \begin{bmatrix} \rho \\ v^i \\ \epsilon \end{bmatrix} \tag{4.55}$$

from the conserved variables U in Eq. (4.15). This procedure is based on the method presented in Rezzolla & Zanotti (2013), and parametrizes the relation between the conserved and primitive variables in terms of the magnitude of the spatial projection of the fluid four-

velocity

$$z = \sqrt{\gamma_{ij} u^i u^j} = Wv. \tag{4.56}$$

To recover the variables in V as a function of the parameter z requires the Lorentz factor W, which can be found from z by using its definition $W^2 = 1/\big(1 - v^2\big)$ as

$$z^2 = W^2 v^2 = W^2 - 1 \quad \Longrightarrow \quad W(z) = \sqrt{1 + z^2}. \tag{4.57}$$

Noting that $\mathcal{D} = \rho W$, the rest-mass density is trivially obtained as

$$\rho = \frac{\mathcal{D}}{W(z)}. \tag{4.58}$$

Next, the internal energy will require solving Eq. (4.2) for ϵ and eliminating h and p in terms of the conserved variables and z. It follows from Eq. (4.5) that

$$z = \frac{\mathcal{S}}{h\mathcal{D}} \equiv \frac{\sqrt{\gamma_{ij}\mathcal{S}^i\mathcal{S}^j}}{h\mathcal{D}} \quad \Longrightarrow \quad h = \frac{\mathcal{S}}{z\mathcal{D}}. \tag{4.59}$$

Using Eq. (4.4) in $\tau = \mathcal{E} - \mathcal{D}$ shows that

$$\tau = \mathcal{D}(hW - 1) - p \quad \Longrightarrow \quad p = \mathcal{D}(hW - 1) - \tau. \tag{4.60}$$

Finally, Eqns. (4.56)–(4.60) are used in Eq. (4.2) to express the internal energy as

$$\epsilon(z) = h - \frac{p}{\rho} - 1 = W(z)\Big[\frac{\tau}{\mathcal{D}} + 1\Big] - z\frac{\mathcal{S}}{\mathcal{D}} - 1. \tag{4.61}$$

The pressure is then obtained from the equation of state as

$$p(z) = p[\rho(z), \epsilon(z)] \tag{4.62}$$

Additionally, the fluid velocity is obtained from comparison of Eq. (4.59) to Eq. (4.56) as

$$v^i(z) = \frac{\mathcal{S}^i}{W(z)h(z)D}, \tag{4.63}$$

where $h(z)$ results from using Eq. (4.58) and Eqns. (4.61)–(4.62) in Eq. (4.2).

The value of z that holds for the previous relations must now be solved for numerically. First, Eq. (4.59) is rewritten as the function

$$\mathsf{P}(z) = z - \frac{\mathcal{S}}{h(z)\mathcal{D}} = 0. \tag{4.64}$$

For bracketed root-finding methods, a tightly-bounded range of possible z values will increase the convergence-rate of the solution. These bounds can be determined from the bounding values of the the magnitude of the fluid velocity, which can be found from the ratio of $\mathcal{S}$ and $\mathcal{E} = \tau + \mathcal{D}$ as

$$k = \frac{\mathcal{S}}{\tau + \mathcal{D}} = \frac{\rho h W^2 v}{\rho h W^2 - p}. \tag{4.65}$$

The smallest value v can take occurs in the limit $p/e = 1$, while the largest value occurs when $p = 0$. Using these in Eq. (4.65) leads to (Rezzolla & Zanotti, 2013)

$$\frac{1}{2}k \leq v \leq k < 1. \tag{4.66}$$

Using these limits in Eqns. (4.56)–(4.57) leads to the valid range of z values

$$\frac{k}{\sqrt{4-k^2}} \leq z \leq \frac{k}{\sqrt{1-k^2}}. \tag{4.67}$$

This provides the tight bracket around the root of Eq. (4.64), which is then solved using the bracketed Brent method (Brent, 2002). From this obtained value of z and the conserved variables U, all primitive variables V and the pressure can be found from Eqns. (4.57)–(4.63) and the equation of state.

Chapter 5

Dynamic Spacetime Evolution

In the presence of compact objects, such as black holes and neutron stars, the underlying geometry of spacetime can no longer be assumed to be flat. Newtonian and special-relativistic treatments of the evolving neutrino radiation and fluid are not capable of fulling capturing the effects of a curved spacetime through simplistic use of a gravitational potential. Only a fully general relativistic descriptions of the spacetime and treatment of the evolution equations are capable of fully capturing the effects of the spacetime's curvature and the influence of matter and energy on the spacetime. While stationary background spacetimes, such as the exterior of a non-rotating stationary black hole, may be suitable for some scenarios, an evolving spacetime will be necessary for studying neutron star mergers and the formation of compact remnants in core-collapse supernovae.

An arbitrary spacetime is described by Einstein's field equations of general relativity (Misner et al., 2017; Baumgarte & Shapiro, 2010)

$$R_{ab} - \frac{1}{2} g_{ab} R = 8\pi T_{ab}\,, \tag{5.1}$$

where T_{ab} is the stress-energy tensor accounting for all matter and energy (fluid, neutrinos, etc.), g_{ab} is the spacetime metric, and the Ricci tensor and scalar are

$$R_{ab} = R^{c}{}_{acb}, \tag{5.2}$$

$$R = R^{a}{}_{a}, \tag{5.3}$$

and describe the intrinsic curvature of the spacetime in terms of the Riemann curvature tensor

$$R^{a}{}_{bcd} = \partial_c \Gamma^{a}{}_{bd} - \partial_d \Gamma^{a}{}_{bc} + \Gamma^{a}{}_{ec}\Gamma^{e}{}_{bd} - \Gamma^{a}{}_{ed}\Gamma^{e}{}_{bc}, \tag{5.4}$$

where $\Gamma^{a}{}_{bc}$ are the Christoffel symbols defined in Eq. (C.12). Both Eq. (5.4) and the Christoffel symbols assume that the spacetime is described by a coordinate basis (which will always be the case in this work). Taking the trace of Eq. (5.1) and using Eq. (5.3) also shows that

$$R = -8\pi T, \tag{5.5}$$

where T is the trace of the stress-energy tensor.

Directly solving Eq. (5.1) is far from practical. A common approach is to decompose the spacetime into spacelike hypersurfaces at constant coordinate times, the so-called '3+1 split' used in numerical relativity; see Baumgarte & Shapiro (2010) for a complete presentation of this method, and refer to appendix C for an overview of the notation and concepts used in this work. In this splitting, the lapse function α characterizes the distance between the hypersurfaces in time, the shift vector β^i characterizes the shift in coordinates along the proper-timelike direction from one hypersurface to the next, the spatial metric γ_{ab} measures distance in and projects onto the hypersurfaces, and the extrinsic curvature K_{ab} describes the curvature of the hypersurfaces relative to the full spacetime manifold, such that the invariant line element associated with the spacetime takes the form

$$\mathrm{d}s^2 = g_{ab}\,\mathrm{d}x^a\,\mathrm{d}x^b = -\alpha^2\,\mathrm{d}t^2 + \gamma_{ij}\Big(\mathrm{d}x^i + \beta^i\,\mathrm{d}t\Big)\Big(\mathrm{d}x^j + \beta^j\,\mathrm{d}t\Big). \tag{5.6}$$

Applying this splitting to Eq. (5.1) leads to a set of evolution equations for the spatial metric and extrinsic curvature, and the Hamiltonian and momentum constraint equations for energy and momentum conservation, respectively. The resulting equations, first proposed in Arnowitt et al. (1962), are referred to the Arnowitt, Deser and Misner (ADM) equations. Unfortunately, they are of limited practical use: this formalism assumes that the Hamiltonian and momentum constraints of the Einstein equations hold exactly, but outside of spherically symmetric or axisymmetric spacetimes, this can lead to numerical instabilities due to numerical errors causing violations of the constraints (Baumgarte & Shapiro, 2010). Instead, fully-constrained formulations or constraint-damping schemes can be used to stably evolve the spacetime. This chapter will describe two variations of a constraint-damping formalism for evolving the spacetime alongside the fluid and neutrinos, and the code generation utility developed to implement the resulting equations in numerical solvers. Their implementations in `Flash-X` will be described later in chapter 6.

5.1 Mathematical Formalism

The elliptic form that the constraint equations take in the standard 3+1 splitting of Eq. (5.1) proves challenging to stably solve numerically. The Z4 formulation of the Einstein equations reduce the constraints to a more manageable system of first-order equations by introducing a new four-vector field Z_a, that when allowing for damping of constraint violations, puts Eq. (5.1) into the form Gundlach et al. (2005)

$$R_{ab} - \frac{1}{2} g_{ab} R + \boldsymbol{\nabla}_a Z_b + \boldsymbol{\nabla}_b Z_a - g_{ab} \boldsymbol{\nabla}_c Z^c - \kappa_1 \big(n_a Z_b + n_b Z^a + \kappa_2 g_{ab} n_c Z^c \big) = 8\pi T_{ab} \,, \tag{5.7}$$

where n_a is a normal to the spacelike hypersurface, and $\kappa_1, \kappa_2 \geq 0$ are constants that determine the strength of the damping.

This section will present the two decomposition of the Z4 formulation used in this work: the Z4c conformal decomposition as described in Bernuzzi & Hilditch (2010); Cao & Hilditch (2012), and the CCZ4 conformal and covariant decomposition as described in Alic et al. (2012); Radia et al. (2022). Both formulations will use the conformally-related variables

$$\chi = \gamma^{-\frac{1}{3}}, \tag{5.8}$$

$$\tilde{\gamma}_{ij} = \chi \gamma_{ij}, \tag{5.9}$$

$$\tilde{A}_{ij} = \chi \left(K_{ij} - \frac{1}{3} \gamma_{ij} K \right), \tag{5.10}$$

$$\Theta = -n_a Z^a, \tag{5.11}$$

$$\hat{\Gamma}^i = \tilde{\Gamma}^i + 2\tilde{\gamma}^{ij} Z_j, \tag{5.12}$$

where χ is the conformal factor, $\tilde{\gamma}_{ij}$ is the conformally-related spatial metric, $\tilde{A}_{ij}$ is the trace-free part of the conformally-related extrinsic curvature, Θ is the timelike projection of the Z4 vector Z^a, and $\hat{\Gamma}^i$ is a modification to the conformally-related connection function

$$\tilde{\Gamma}^i = \tilde{\gamma}^{jk} \tilde{\Gamma}^i{}_{jk}, \tag{5.13}$$

where $\tilde{\Gamma}^i{}_{jk}$ are the Christoffel symbols for the conformally-related spatial metric $\tilde{\gamma}_{ij}$ found by using Eq. (5.9) in Eq. (C.12). The conformally-related quantities, indicated with a tilde, are purely spatial, i.e. orthogonal to the hypersurface normal n^a, and have indices raised and lowered by the conformally-related spatial metric. The spatial metric and extrinsic curvature

relate to these new variables as

$$\gamma_{ij} = \frac{1}{\chi}\tilde{\gamma}_{ij}, \tag{5.14}$$

$$K_{ij} = \frac{1}{\chi}\tilde{A}_{ij} + \frac{1}{3}\gamma_{ij}K. \tag{5.15}$$

The evolution and constraint equations will all make use of the Eulerian projections of the full stress-energy tensor, including contributions from all matter and energy sources,

$$\mathcal{E} = n_a n_b T^{ab}, \tag{5.16}$$

$$\mathcal{S}^i = -\gamma^i{}_a n_b T^{ab}, \tag{5.17}$$

$$\mathcal{S}^{ij} = \gamma^i{}_a \gamma^j{}_b T^{ab}, \tag{5.18}$$

With these variables, the Hamiltonian and momentum constraints, $\mathcal{H}$ and $\mathcal{M}^i$, respectively, take the form (Radia et al., 2022)

$$\mathcal{H} = R + \frac{2}{3}K^2 - \tilde{A}_{ij}\tilde{A}^{ij} - 16\pi\mathcal{E} \tag{5.19}$$

$$\mathcal{M}^i = \tilde{\gamma}^{jk}\Big(\partial_j \tilde{A}_{ki} - \tilde{\Gamma}^l{}_{ki}\tilde{A}_{jl} - \tilde{\Gamma}^l{}_{kj}\tilde{A}_{il} - 3\tilde{A}_{ij}\partial_k \ln\chi\Big) - \frac{2}{3}\partial_i K - 8\pi\mathcal{S}_i\,. \tag{5.20}$$

Two additional algebraic constraints also follow from the definitions of the conformally-related spatial metric and trace-free extrinsic curvature:

$$\det\tilde{\gamma}_{ij} = 1, \tag{5.21}$$

$$\tilde{\gamma}^{ij}\tilde{A}_{ij} = 0. \tag{5.22}$$

The remainder of this section will present the two different Z4 formulations and the slicing and gauge conditions used in this work. Both formulations will define their evolution equations using the operator

$$\partial_0 \equiv \partial_t - \beta^k \partial_k, \tag{5.23}$$

which acts as the time-derivative in the direction of the hypersurface normal n^a. Covariant derivatives with respect to the spatial metrics will be computed as

$$D_i X^j{}_k = \partial_i X^j{}_k + \Gamma^j{}_{il} X^l{}_k - \Gamma^l{}_{ik} X^j{}_l, \tag{5.24}$$

$$\tilde{D}_i \tilde{X}^j{}_k = \partial_i \tilde{X}^j{}_k + \tilde{\Gamma}^j{}_{il} \tilde{X}^l{}_k - \tilde{\Gamma}^l{}_{ik} \tilde{X}^j{}_l, \tag{5.25}$$

for some arbitrary tensor $X^j{}_k$, where $\tilde{\Gamma}^i{}_{jk}$ are the Christoffel symbols of the conformally-related spatial metric found from using Eq. (5.9) in Eq. (C.12). Additionally, both formulations will also make use of trace-free components of various rank-two tensors; these quantities will be denoted by the superscript "tf" and represent the operations, again for an arbitrary rank-two tensor X_{ij} and its conformally-related counterpart,

$$X^{\mathrm{tf}}_{ij} = X_{ij} - \frac{1}{3}\gamma_{ij}\left(\gamma^{kl} X_{kl}\right), \tag{5.26}$$

$$\tilde{X}^{\mathrm{tf}}_{ij} = \tilde{X}_{ij} - \frac{1}{3}\tilde{\gamma}_{ij}\left(\tilde{\gamma}^{kl} \tilde{X}_{kl}\right). \tag{5.27}$$

5.1.1 Z4c

The Z4c formulation results from the conformal 3+1 decomposition of the Z4 system in Eq. (5.7) while discarding non-primary non-damping terms, and is based on the formulations

presented in Bernuzzi & Hilditch (2010); Cao & Hilditch (2012). This formulation directly evolves the conformally-related χ, $\tilde{\gamma}_{ij}$, $\tilde{A}_{ij}$, Θ, and $\tilde{\Gamma}_i$, as well as the modified trace of the extrinsic curvature

$$\hat{K} = K - 2\Theta. \tag{5.28}$$

The evolution equations for the variables are

$$\partial_0 \chi = \frac{2}{3}\Big[\alpha\Big(\hat{K} + 2\Theta\Big) - \partial_k \beta^k\Big], \tag{5.29}$$

$$\partial_0 \tilde{\gamma}_{ij} = -2\alpha\tilde{A}_{ij} + \tilde{\gamma}_{ki}\partial_j\beta^k + \tilde{\gamma}_{kj}\partial_i\beta^k - \frac{2}{3}\tilde{\gamma}_{ij}\partial_k\beta^k, \tag{5.30}$$

$$\partial_0 \hat{K} = -D^k D_k \alpha + \alpha\left[\tilde{A}^{jk}\tilde{A}_{jk} + \frac{1}{3}\Big(\hat{K} + 2\Theta\Big)^2 + \kappa_1(1+\kappa_2)\Theta\right] + 4\pi\alpha(\mathcal{E} + \mathcal{S}), \tag{5.31}$$

$$\begin{aligned}\partial_0 \tilde{A}_{ij} = {} & \chi\Big[-D_i D_j \alpha + \alpha\Big(R_{ij} - 8\pi\mathcal{S}_{ij}\Big)\Big]^{\mathrm{tf}} + \alpha\Big[\Big(\hat{K} + 2\Theta\Big)\tilde{A}_{ij} - 2\tilde{\gamma}^{kl}\tilde{A}_{ik}\tilde{A}_{lj}\Big] \\ & + \tilde{A}_{ki}\partial_j\beta^k + \tilde{A}_{kj}\partial_i\beta^k - \frac{2}{3}\tilde{A}_{ij}\partial_k\beta^k,\end{aligned} \tag{5.32}$$

$$\partial_0 \Theta = \frac{1}{2}\alpha\left[R - \tilde{A}_{ij}\tilde{A}^{ij} + \frac{2}{3}\Big(\hat{K} + 2\Theta\Big)^2 - 16\pi\mathcal{E} - 2\kappa_1(1+\kappa_2)\Theta\right], \tag{5.33}$$

$$\begin{aligned}\partial_0 \hat{\Gamma}^i = {} & 2\alpha\left[\tilde{\Gamma}^i{}_{jk}\tilde{A}_{jk} - \frac{3}{2}\partial_j \ln\chi - \frac{1}{3}\tilde{\gamma}^{ij}\partial_j\Big(\hat{K} + 2\Theta\Big) - \kappa_1\Big(\hat{\Gamma}^i - \tilde{\Gamma}^i\Big) - 8\pi\tilde{\gamma}^{ij}\mathcal{S}_j\right] \\ & - 2\tilde{A}^{ij}\partial_j\alpha + \frac{2}{3}\tilde{\Gamma}^i\partial_k\beta^k - \tilde{\Gamma}^j\partial_j\beta^i + \tilde{\gamma}^{jk}\partial_j\partial_k\beta^i + \frac{1}{3}\tilde{\gamma}^{ij}\partial_j\partial_k\beta^k,\end{aligned} \tag{5.34}$$

where $\mathcal{S} = \gamma_{ij}\mathcal{S}^{ij}$.

The Ricci tensor is separated into terms proportional to the derivatives of the conformal factor and the derivatives of the conformally-related spatial metric

$$R_{ij} = R^{\chi}_{ij} + \tilde{R}_{ij}, \tag{5.35}$$

where

$$R^{\chi}_{ij} = \frac{1}{2\chi}\Big[\tilde{D}_i\tilde{D}_j\chi + \tilde{\gamma}_{ij}\tilde{\gamma}^{kl}\tilde{D}_k\tilde{D}_l\chi\Big] - \frac{1}{4\chi^2}\Big[\partial_i\chi\partial_j\chi + 3\tilde{\gamma}_{ij}\tilde{\gamma}^{kl}\partial_k\chi\partial_l\chi\Big] \tag{5.36}$$

$$\begin{aligned}\tilde{R}_{ij} = &\frac{1}{2}\Big[\tilde{\gamma}_{ki}\partial_j\hat{\Gamma}^k + \tilde{\gamma}_{kj}\partial_i\hat{\Gamma}^k - \tilde{\gamma}^{kl}\partial_k\partial_l\tilde{\gamma}_{ij} + \tilde{\Gamma}^k\Big(\tilde{\Gamma}_{ijk} + \tilde{\Gamma}_{jik}\Big)\Big] \\ &+ \tilde{\gamma}^{lm}\Big[\tilde{\Gamma}^k{}_{li}\tilde{\Gamma}_{jkm} + \tilde{\Gamma}^k{}_{lj}\tilde{\Gamma}_{ikm} + \tilde{\Gamma}^k{}_{im}\tilde{\Gamma}_{klj}\Big].\end{aligned} \tag{5.37}$$

5.1.2 CCZ4

The CCZ4 formulation similarly results from the conformal 3+1 decomposition of Eq. (5.7), but unlike Z4c, its evolution equations remain fully-covariant (Alic et al., 2012). This section will describe a CCZ4 formulation based on the one presented in Radia et al. (2022). Similarly to Z4c, this formulation also evolves the conformally-related χ, $\tilde{\gamma}_{ij}$, $\tilde{A}_{ij}$, Θ, and $\tilde{\Gamma}_i$, but directly evolves K, the trace of the extrinsic curvature. The evolution equations presented here, which are based on the ones in Radia et al. (2022), differ from the original formulation in Alic et al. (2012) primarily by the modification of the Ricci tensor and scalar to absorb terms proportional to the covariant derivatives of Z^i that appear in the CCZ4 equations. This modification allows the conformally-related Ricci tensor to be written in terms of the evolved $\hat{\Gamma}^i$ and its derivatives instead of the derived connection functions $\tilde{\Gamma}^i$. The modified Ricci tensor is

$$\hat{R}_{ij} = R_{ij} + D_iZ_j + D_jZ_i = \tilde{R}_{ij} + R^{\chi}_{ij} + R^{Z}_{ij}, \tag{5.38}$$

where $\tilde{R}_{ij}$ and R^{χ}_{ij} are still defined as in Eqns. (5.37)–(5.36), and

$$R^{Z}_{ij} = \frac{Z^k}{\chi^2}\Big(\tilde{\gamma}_{ik}\partial_j\chi + \tilde{\gamma}_{jk}\partial_i\chi - \tilde{\gamma}_{ij}\partial_k\chi\Big), \tag{5.39}$$

where $Z^i = \gamma^i{}_a Z^a$ is the spatially-projected Z4 vector.

The evolution equations for the conformal factor and the conformally-related metric are exactly the same as their Z4c equations in Eqns. (5.29)–(5.30). The new evolution equation for CCZ4 are

$$\partial_0 K = -D^k D_k\alpha + \alpha\Big[\hat{R} + K(K-2\Theta) - 3\alpha\kappa_1(1+\kappa_2)\Theta\Big] + 4\pi\alpha(\mathcal{S}-3\mathcal{E}) \tag{5.40}$$

$$\begin{aligned}\partial_0\tilde{A}_{ij} = {}& \chi\Big[-D_iD_j\alpha + \alpha\Big(\hat{R}_{ij} - 8\pi\mathcal{S}_{ij}\Big)\Big]^{\mathrm{tf}} + \alpha\Big[(K-2\Theta)\tilde{A}_{ij} - 2\tilde{\gamma}^{kl}\tilde{A}_{ik}\tilde{A}_{lj}\Big] \\ & + \tilde{A}_{ki}\partial_j\beta^k + \tilde{A}_{kj}\partial_i\beta^k - \frac{2}{3}\tilde{A}_{ij}\partial_k\beta^k,\end{aligned} \tag{5.41}$$

$$\partial_0\Theta = \frac{1}{2}\alpha\left[\hat{R} - \tilde{A}_{ij}\tilde{A}^{ij} + \frac{2}{3}K^2 - 2\Theta K - 16\pi\mathcal{E} - 2\kappa_1(2+\kappa_2)\Theta\right] - Z^k\partial_k\alpha, \tag{5.42}$$

$$\begin{aligned}\partial_0\hat{\Gamma}^i = {}& \frac{2}{3}\left[\partial_k\beta^k\left(\tilde{\Gamma}^i + 2\kappa_3\frac{Z^i}{\chi}\right) - \frac{3}{2}\left(\tilde{\Gamma}^k + 2\kappa_3\frac{Z^k}{\chi}\right)\partial_k\beta^i\right] - 2\alpha\left[\frac{2}{3}K + \kappa_1\right]\frac{Z^i}{\chi} \\ & - \alpha\left[\frac{4}{3}\tilde{\gamma}^{ij}\partial_j K + 3\tilde{A}^{ij}\partial_j\ln\chi\right] - 2\tilde{A}^{ij}\partial_j\alpha + 2\alpha\tilde{\Gamma}^i{}_{jk}\tilde{A}^{jk} - 16\pi\alpha\tilde{\gamma}^{ij}\mathcal{S}_j \\ & + 2\tilde{\gamma}^{ij}\Big[\alpha\partial_j\Theta - \Theta\partial_j\alpha\Big] + \tilde{\gamma}^{jk}\partial_j\partial_k\beta^i + \frac{1}{3}\tilde{\gamma}^{ij}\partial_j\partial_k\beta^k,\end{aligned} \tag{5.43}$$

where κ_3 is an additional constant not present in the Z4c equations, and determines the full covariance of the evolution equations. Setting $\kappa_3 = 1$ results in a fully-covariant system, but this can lead to numerical instabilities when there is non-linear coupling between the various damping terms; these effects can be present in black hole spacetimes, where it is beneficial to set $\kappa_3 = 1/2$ (Alic et al., 2012). Alternatively, the replacement $\alpha\kappa_1 \to \kappa_1$ permits stable evolution of black hole spacetimes when setting $\kappa_3 = 1$ (Radia et al., 2022).

5.1.3 Slicing and Gauge Conditions

The choice of slicing and gauge conditions determine the evolution of the lapse and shift. At present, both formulations make use of the moving puncture gauge conditions, which are a combination of '1+log' slicing for the lapse and a hyperbolic Γ-driver shift, and are given in Baumgarte & Shapiro (2010) as

$$\partial_0 \alpha = -2\alpha K, \tag{5.44}$$

$$\partial_0 \beta^i = \frac{3}{4} B^i, \tag{5.45}$$

$$\partial_0 B^i = \partial_0 \hat{\Gamma}^i - \eta B^i, \tag{5.46}$$

where η is a damping parameter; for spacetimes outside of a compact object of mass M, this parameter will be of the order $\eta = \mathcal{O}(1/(2M))$. Different variations of these gauge conditions, e.g. a harmonic lapse slicing and a non-hyperbolic Γ-driver shift presented in Cao & Hilditch (2012), can also be used alongside the different Z4 formulations, but Eqns. (5.44)–(5.46) will prove useful in testing these formulations in vacuum and black hole spacetimes.

5.2 Numerical Methods

For use in the description of the numerical methods applied to the Z4 formulations in the preceding section, the evolution equations will be written as

$$\partial_t \mathbf{V} - \beta^j \partial_j \mathbf{V} = \mathbf{E}\Big(\mathbf{V},\, \partial_i \mathbf{V},\, \partial_i \partial_j \mathbf{V}\Big), \tag{5.47}$$

where the vector evolved variables is

$$\mathbf{V} = \begin{cases} \left[\chi, \tilde{\gamma}_{ij}, \hat{K}, \tilde{A}_{ij}, \Theta, \hat{\Gamma}^i, \alpha, \beta^i, B^i\right]^{\mathrm{T}} & \text{(Z4c)} \\ \left[\chi, \tilde{\gamma}_{ij}, K, \tilde{A}_{ij}, \Theta, \hat{\Gamma}^i, \alpha, \beta^i, B^i\right]^{\mathrm{T}} & \text{(CCZ4)} \end{cases}, \tag{5.48}$$

and the function $\mathbf{E}$ represents the the right-hand sides of the either the Z4c or CCZ4 evolution equations (omitted here for brevity) and notes the explicit dependence on the evolved variables and their spatial first- and second-derivatives.

Both the Z4c and CCZ4 formulations utilize a finite-difference spatial discretization. For compatibility with the neutrino radiation and hydrodynamics solvers described in chapters 3–4 and the mesh structures available in `Flash-X`, the evolved metric and curvature variables are located at the cell-centers. All derivatives in the right-hand side function $\mathbf{E}$ use centered fourth-order accurate differences of the form

$$\partial_i \mathbf{V} \approx \mathbf{D}_i \mathbf{V}, \tag{5.49}$$

$$\partial_i \partial_j \mathbf{V} \approx \mathbf{D}_{ij} \mathbf{V}, \tag{5.50}$$

where the difference operators are

$$\mathbf{D}_i \mathbf{V} = \frac{\mathbf{V}_{i-2} - 8\mathbf{V}_{i-1} + 8\mathbf{V}_{i+1} - \mathbf{V}_{i+2}}{12\Delta x^i}, \tag{5.51}$$

and

$$\mathbf{D}_{ij} \mathbf{V} = \begin{cases} \dfrac{-\mathbf{V}_{i-2} + 16\mathbf{V}_{i-1} - 30\mathbf{V}_i + 16\mathbf{V}_{i+1} - \mathbf{V}_{i+2}}{12\left(\Delta x^i\right)^2} & i = j \\ \mathbf{D}_i \mathbf{D}_j \mathbf{V} & i \neq j \end{cases}, \tag{5.52}$$

where the $i \neq j$ case represents the consecutive applications of the first-derivative difference operator along each direction. The advective derivatives along the shift utilize an upwind-biased derivative for stability (Cao & Hilditch, 2012; Radia et al., 2022)

$$\beta^j \partial_j \mathbf{V} \approx \beta^j \mathbf{D}_j^+ \mathbf{V},$$
$$\mathbf{D}_j^+ \mathbf{V} = \begin{cases} \dfrac{-3\mathbf{V}_{i-1} - 10\mathbf{V}_i + 18\mathbf{V}_{i+1} - 6\mathbf{V}_{i+2} + \mathbf{V}_{i+3}}{12\Delta x^j} & \beta^j > 0 \\ \dfrac{-\mathbf{V}_{i-3} + 6\mathbf{V}_{i-2} - 18\mathbf{V}_{i-1} + 10\mathbf{V}_i + 3\mathbf{V}_{i+1}}{12\Delta x^j} & \beta^j < 0 \end{cases}. \tag{5.53}$$

Similarly to the hydrodynamics solver described in chapter 4, both the Z4c and CCZ4 implementations make use of a method-of-lines time discretization and the fourth-order accurate Runge-Kutta explicit method. When using an implicit-explicit (IMEX) method for integrating the neutrino moment evolution equations, a fourth-order accurate IMEX method will be necessary. As is common practice in numerical relativity, the time-derivative operators are modified to include Kreiss-Oliger dissipation (Baumgarte & Shapiro, 2010; Rezzolla & Zanotti, 2013). Following the methods used in Cao & Hilditch (2012); Radia et al. (2022), sixth-order dissipation is added by re-defining the time-derivative operator as

$$\partial_t \mathbf{V} \to \partial_t \mathbf{V} - \sigma \sum_i \frac{\mathbf{V}_{i-3} - 6\mathbf{V}_{i-2} + 15\mathbf{V}_{i-1} - 20\mathbf{V}_i + 15\mathbf{V}_{i+1} - 6\mathbf{V}_{i+1} + \mathbf{V}_{i+3}}{64\Delta x^i}, \tag{5.54}$$

where σ is a constant specifying the strength of the dissipation. This change is not made at the level of the time-integrator, but is accounted for by adding the dissipation term in to the evaluation of the evolution equations right-hand sides.

5.3 Code Generation

Implementing the evolution equations for either the Z4c or CCZ4 formulations is far from a straightforward exercise. Evaluation of the evolution equations and the derived, composite quantities, such as the Ricci tensor, require copious amounts of tensor algebra. Manual translation of the symbolic forms of the evolution equations presented in this chapter into component-wise equations and operations in any programming language suitable for large-scale numerical simulations is as error prone as it is tedious.

A common approach in numerical relativity is the use of code generators to translate these symbolic expressions into usable code. A number of code generation packages exist, typically targeted at specific numerical relativity codes, with some recent examples being `NRPy+` for the Einstein Toolkit (Ruchlin et al., 2018), and `STvAR` for `AMReX`-based codes (Peterson et al., 2023). While these code-generators work well alongside their intended target codes, the utilities and assumptions specific to their respective codes are not directly adaptable elsewhere. This section will describe a set of code-generation utilities that originally began as an extension of `STvAR` for use in `Flash-X` to produce Fortran code, but subsequently underwent a complete re-design to facilitate use and re-usability, and improve the quality and readability of the generated Fortran code.

The new code-generation capabilities aim to extend existing capabilities in the open-source symbolic algebra Python package `SymPy` (Meurer et al., 2017). `SymPy` includes a robust tensor algebra module (`sympy.tensor.tensor`) that unfortunately is not compatible with its existing code-generation and printing utilities. Due to this limitation, modules and data structures compatible with the existing `SymPy` code-generation utilities, e.g. `IndexedBase`, are typically used directly in loop-based calculations of tensorial equations. The approach

taken here will instead connect `SymPy`'s symbolic tensor algebra and code-generation capabilities. Extensions to the symbolic tensor algebra module will provide replacement rules for converting symbolic tensorial expressions into sets of data-types and expressions compatible with the abstract syntax tree nodes used by the code-generation and -printing utilities. New and updated abstract syntax tree nodes will then automate the application of these replacement rules when used in an updated modern Fortran code printer.

5.3.1 Symbolic Tensor Algebra

In `SymPy`'s symbolic tensor algebra module, symbolic tensors and their indices are represented by the `TensorHead` and `TensorIndex` data-types, respectively. These are then used to form three types of symbolic expressions: indexed expressions representing a specific tensor, and expressions for addition and products of other expressions. These allow for translating tensor equations directly into tensor expressions by making use of an Einstein summation notation.

Code snippet 5.1 provides an example of this procedure by constructing a symbolic tensor expression for the trace of the extrinsic curvature, $K = \gamma^{ij} K_{ij}$. Both the tensors and their indices rely on the specification of a `TensorIndexType` to match the defined tensor indices to the tensor "slots" they can be used in. Additionally, each tensor is constructed as fully-symmetric by providing a `TensorSymmetry` object. The final line demonstrates two types of tensor expressions: the indexed expressions `gamma(i,j)` and `K(-i,-j)`, and the product expression of these two indexed expressions. In this notation, `i,j` represent contravariant (raised) indices, and `-i,-j` represent covariant (lowered) indices. This expression assumes these indices are fully-contracted such that the overall product results in a scalar expression.

The individual and composite tensor expressions only specify the tensors and operations involved, but do not specialize to a specific basis, dimension, or values until they are eval-

```
# A basic type for all tensor indices
spatial = TensorIndexType("spatial")

# Symmetry specification for a fully symmetric rank-2 tensor
sym = TensorSymmetry.fully_symmetric(2)

# Symbolic tensor indices
i = TensorIndex("i", spatial)
j = TensorIndex("j", spatial)

# Spatial metric and extrinsic curvature tensors
gamma = TensorHead("gamma", [spatial]*2, sym)
K     = TensorHead("K", [spatial]*2, sym)

traceK = gamma(i,j)*K(-i,-j)
```

Code Snippet 5.1: Example symbolic tensor expressions using `sympy.tensor.tensor`

uated. This evaluation occurs by applying replacement rules to indexed tensor expressions with arrays containing values or other symbolic expressions that represent individual tensor components. The replacement array must be of the same rank as the target indexed tensor expression, and the array's dimensions will determine the tensor's dimensions. This process is demonstrated in code snippet 5.2 by evaluating the Eulerian decomposition of the fluid four-velocity in terms of two symbolic tensors for the fluid velocity v^a and hypersurface normal n^a, and the scalar Lorentz factor W represented by a `SymPy Symbol` object. The replacement rules are provided as a Python dictionary associating each indexed tensor expression with a list of symbols for each component.

While these symbolic tensor expressions provide a powerful set of tools for evaluating tensorial equations, there are a few limitations that complicate their use in code-generation. Each tensor object can be indexed with either raised or lowered indices, e.g. `n(a)` and `n(-a)`. This requires either separate replacement rules for each possible index configuration, or the association of a metric with the underlying index types. The former option is error-prone

```
# Index type for the four-vectors
spacetime = TensorIndexType("spacetime")

# Symbolic index
a = TensorIndex("a", spacetime)

# Fluid velocity and normal vectors
v = TensorHead("v", [spacetime])
n = TensorHead("n", [spacetime])

# Lorentz factor
W = Symbol("W")

# Eulerian decomposition of the four-velocity
u = W*(n(a) + v(a))

# Evaluate by replacing tensor components with SymPy Symbol objects
u.replace_with_arrays({
   n(a): symbols("nt,nx,ny,nz"),
   v(a): symbols("vt,vx,vy,vz")
})

# Produces a list with scalar expressions for each component
[  W*(nt + vt),  W*(nx + vx),  W*(ny + vy),  W*(nz + vz)  ]
```

Code Snippet 5.2: Evaluating symbolic tensor expressions with `sympy.tensor.tensor`

as each possible index configuration must be manually specified, which can become increasingly convoluted for higher-rank tensors. The latter option unfortunately only works well with simple analytic metric, e.g. Minkowski or Schwarzschild, while more generic metrics, e.g. ones with just named symbols for components representing their evolved values, do not efficiently work with the internal inversion and contractions performed when evaluating tensor expressions. Another limitation is a lack of symbolic operators compatible with the tensor expressions. `SymPy` only provides one operator, `PartialDerivative`, that cannot directly be replaced (only its operands can), and can only be evaluated analytically, e.g. provided $\gamma_{\theta\theta} = r^2$ it will be able to evaluate $\partial_r \gamma_{\theta\theta} = 2r$. The remainder of this section will

describe extensions to `SymPy`'s symbolic tensor algebra capabilities that improve on these limitations.

As previously mentioned, manually providing replacement rules for all possible index configurations can become increasingly complex, particularly when dealing with the large systems of equations present in numerical relativity. This can become even more complex when multiple representations are necessary, e.g. when a tensor component needs a scalar and grid variable representation in the final generated code. To reduce this complexity, this work extends `TensorHead` to generate a series of replacement rules based on the provided tensor name, symmetries, and index configuration. These replacement rules use an `Indexed` object formed from an `IndexedBase`, representing the tensor and its index configuration, and series of symbolic integers, representing specific components of the tensor. The index configuration, which will be specific to and enforced for each tensor, is included in the `IndexedBase` with a sequence of `L` and `U` characters for covariant and contravariant index slots, respectively. Since this is targeted at numerical relativity, tensors will all be considered four-dimensional, with an optional spatial-only flag that produces only non-zero spatial components. This new functionality is included in a new derived class `SymbolicTensor` that inherits the full tensor expression capabilities of `TensorHead`. Code snippet 5.3 provides an example of using `SymbolicTensor` to generate a `TensorHead` and its replacement rules for the spatial Christoffel symbols ${\Gamma^i}_{jk}$; these replacement rules produce `Indexed` objects for each component that represent a compact form that reduces the symmetric index slots to a single rank representing only the unique components.

Each symbolic tensor will also generate a set of replacement rules in terms of grid variables, i.e. when individual components of a tensor are stored in a larger data structure representing potentially many variables located on a computational grid. The naming rules

```
# Spatial index type
spatial = TensorIndexType("spatial")

# Tensor indices
i,j,k = tensor_indices("i,j,k", spatial)

# New wrapper around TensorSymmetry to aid in replacement-rule generation
# This example specifies that the 1,2 slots (out of 0,1,2) are symmetric
sym = Symmetries(Symmetric([1,2]))

# Christoffel symbol Gamma^i_jk
Gamma_ull = SymbolicTensor(
   "Gamma", [Contravariant,Covariant,Covariant], sym, spatial=True
)

# Apply the replacement rules
Gamma_ull(i,-j,-k).replace_with_arrays(Gamma_ull.repl)

# Produces the list (where X..Z and XX..ZZ are integer symbols)
# The underlying array that each component belongs to will
# be in a compact form as determined by the tenor's symmetry,
# i.e its shape will be (X:Z, XX:ZZ)
[[[  Gamma_ULL(X,XX), Gamma_ULL(X,XY), Gamma_ULL(X,XZ)  ],
  [  Gamma_ULL(X,XY), Gamma_ULL(X,YY), Gamma_ULL(X,YZ)  ],
  [  Gamma_ULL(X,XZ), Gamma_ULL(X,YZ), Gamma_ULL(X,ZZ)  ]],
 ...,
 [[  Gamma_ULL(Z,XX), Gamma_ULL(Z,XY), Gamma_ULL(Z,XZ)  ],
  [  Gamma_ULL(Z,XY), Gamma_ULL(Z,YY), Gamma_ULL(Z,YZ)  ],
  [  Gamma_ULL(Z,XZ), Gamma_ULL(Z,YZ), Gamma_ULL(Z,ZZ)  ]]]
```

Code Snippet 5.3: Automatic replacement rule generation with `SymbolicTensor`

used for the local variable names will be applied to a named integer constant used to index the grid data structure. To facilitate their use within loops over the grid, these replacement rules will additionally take a set of grid indices that can either take specific numeric values or represent an integer variable used as a loop counter. All grid-variable replacement rules also can specify prefixes and suffixes to allow for compatibility with naming conventions in the targeted code. See code snippet 5.4 for an example of the grid-variable replacement rules

again applied to the Christoffel symbols $\Gamma^{i}{}_{jk}$. A similar `SymbolicScalar` extends `Symbol` to also include the generation of grid-variable replacement rules for scalar symbols.

```
# Gamma_ull previously defined as Christoffel symbol

# Grid dimensions
NVAR,NX,NY,NZ = symbols("NVAR,NX,NY,NZ", integer=True)

# Grid indices
inds = symbols("I,J,K", integer=True)

# Represent array of grid variable data as an IndexedBase
vars = IndexedBase("vars", shape=(NVAR,NX,NY,NZ))

# Apply the grid replacement rules
Gamma_ull(i,-j,-k).replace_with_arrays(
   Gamma_ull.as_grid_repl(vars, inds, suffix="_VAR"))

# Produces the list of grid variables
[[[  vars(GAMMA_ULL_XXX_VAR, I, J, K), ...  ],
  [  vars(GAMMA_ULL_XXY_VAR, I, J, K), ...  ],
  [  vars(GAMMA_ULL_XXZ_VAR, I, J, K), ...  ]],
 ...,
 [[  vars(GAMMA_ULL_ZXX_VAR, I, J, K), ...  ],
  [  vars(GAMMA_ULL_ZXY_VAR, I, J, K), ...  ],
  [  vars(GAMMA_ULL_ZXZ_VAR, I, J, K), ...  ]]]
```

Code Snippet 5.4: Automatic replacement rule generation with `SymbolicTensor` for named variable indices in a grid data structure.

With the large number of different types of derivative operators in the Z4c and CCZ4 equations, manually creating symbolic tensors for each tensor and all of its different derivatives would be incredibly cumbersome. A new `TensorOperator` class alleviates these redundancies by inlining the creation of new `SymbolicTensor` objects and expressions with updated names and ranks. For example, a partial derivative operator applied to the spatial metric, i.e. $\partial_k \gamma_{ij}$, will produce a new set of replacement rules for a rank-three tensor and prepend the operator name to the tensor's base name. This procedure is demonstrated in

code snippet 5.5 for the non-inline application of a tensor operator to emphasize the necessary steps and the creation of the new symbolic tensor expression.

```
# Spatial index type
spatial = TensorIndexType("spatial")

# Tensor indices
i,j,k = tensor_indices("i,j,k", spatial)

# Fully symmetric in first two indices
sym = Symmetries(Symmetric([0,1]))

# Spatial metric gamma_ij
gamma_ll = SymbolicTensor(
   "gamma", [Covariant,Covariant], sym, spatial=True)

# Spatial derivative operator
d = TensorOperator("d", [Covariant], spatial=True)

# Apply the operator (this can also be done inline in an expression
# or during application of the replacement rules)
dgamma_lll = d(gamma_ll(-i,-j), k)

# Apply the replacement rules
dgamma_lll.replace_with_arrays(dgamma_lll.repl)

# Produces the list representing the metrics derivatives
[[[  dgamma_LLL(XX,X), dgamma_LLL(XX,Y), dgamma_LLL(XX,Z)  ],
  [  dgamma_LLL(XY,X), dgamma_LLL(XY,Y), dgamma_LLL(XY,Z)  ],
  [  dgamma_LLL(XZ,X), dgamma_LLL(XZ,Y), dgamma_LLL(XZ,Z)  ]],
   ...,
 [[  dgamma_LLL(XZ,X), dgamma_LLL(XZ,Y), dgamma_LLL(XZ,Z)  ],
  [  dgamma_LLL(YZ,X), dgamma_LLL(YZ,Y), dgamma_LLL(YZ,Z)  ],
  [  dgamma_LLL(ZZ,X), dgamma_LLL(ZZ,Y), dgamma_LLL(ZZ,Z)  ]]]
```

Code Snippet 5.5: Applying a `TensorOperator` to a `SymbolicTensor` to generate new replacement rules.

Additional utilities for working with symbolic tensors are also included. Common linear algebra operations such as taking the determinants and inverse of rank-2 tensors, are specialized to take an input symbolic tensor and generate expressions based on the associated re-

placement rules. Finite-difference weights are generated with `SymPy`'s `finite_diff_weights` utility, and then used to create finite-difference operators of arbitrary orders, directions and accuracies. These operators allow the accuracy and bias of the first- and second-order derivatives, as well as the Kreiss-Oliger dissipation terms, to easily be changed when generating the subroutines for calculating both the right-hand sides for the Z4c and CCZ4 evolution and constraint equations.

5.3.2 Code Printing

`SymPy` provides robust code generation and printing utilities that can automatically take certain symbolic expressions and transform them into usable lines of code in a target programming language. The code generators and printers both utilize an abstract syntax tree (AST) that represents features common in most programming languages, such as arithmetic operations and assignments. The general ASTs are also augmented language-specific nodes representing language-specific constructs, data-types, and library functions. `SymPy` expressions make use of a similar tree structure, and in most cases can be automatically parsed into a set of AST nodes for use in code generation.

Unfortunately, not all data-types and expressions are compatible with the ASTs and code-generation utilities. Simple expressions containing `Symbol`, `Indexed`, and similar objects are supported directly in all code generators and printers, but there is no built-in support for the symbolic tensor expressions. Additionally, while some code generators and printers are well maintained and supported, such as C and C++, others contain a hodgepodge of dated usage and formatting assumptions alongside a smattering of modern features, such as Fortran. Since Fortran is the target language for implementing the Z4c and CCZ4 equations in `Flash-X`, these concerns will be addressed alongside updates to the AST to support

symbolic tensor expressions.

The replacement rules for symbolic tensor expressions presented in the previous section all make use of code generation compatible data-types, specifically by using `Symbol` and `Indexed` objects. Once these replacements are applied, each element in the resulting arrays is an expression that can directly be converted into a series of AST nodes. In all practical cases, these expressions will be evaluated for assignment operations, which are represented by the `Assignment` AST node. Two new AST nodes, `ScalarAssignment` and `TensorAssignment`, eliminate the need to manually create these nodes for each component of a tensor expression. Each of these directly take the left- and right-hand sides of a symbolic tensor expression, automate the application of the replacement rules, and generate a set of `Assignment` nodes for the unique components. `TensorAssignment` nodes can either take indexed or non-indexed expressions; the latter will produce elemental array operations and assignments in Fortran.

As this code generator is aimed at producing Fortran code, a number of new Fortran-specific AST nodes have been added. The `ScalarDeclaration` and `TensorDeclaration` nodes provide simple ways of adding variable declarations for the new symbolic types. New attributes such as `target` and `optional` are included for variable declarations; previously only `intent`, `dimension`, `parameter`, and `allocatable` were supported. `ConditionalBlock` nodes provide a flexible way for adding multi-part conditional statements that are not easily represented by the `Piecewise` construct. `SymPy`'s Fortran code printer produces `merge(...)` statements for piecewise expressions when using standards more recent than Fortran77; the `merge` intrinsic function is intended for masked array assignments (Metcalf et al., 2018) and can be an inefficient option when the desired outcome is that only one case is evaluated, as both the true and false cases will be evaluated as inputs to `merge`. A `NestedDo` node simplifies the creation and formatting of nested `do` loops. New pre-processor nodes and

tokens for `#include`, `#define`, and `#if...#else...#endif` blocks; these are used extensively in `Flash-X` for conditionally compiling code based on the chosen spatial dimension. `CodeBlock`'s are extended to include an optional comment-string to prepend to the block for enhanced readability and formatting.

These features are all collectively used in a new `ExtendedSubroutine` node that facilitates creating new procedures by separately accepting input parameters, `use` statements, declarations, and the body of the procedure. The standard `Subroutine` node only accepted basic `SymPy` types as input parameters and placed them immediately prior the the body of the subroutine without allowing for an `implicit none` or `use` statements. This new node also parses all included tensor and scalar assignment blocks for all variable names, categorizes them by type and dimension, and creates declarations (sorted by type and dimension) for variables with no provided declarations. This greatly simplifies and reduces potential error in declaring all locally-used intermediate variables in longer sets of calculations.

A new Fortran code printer, `ModernFCodePrinter` derives functionality from `SymPy`'s `FCodePrinter`, updates formatting rules, and adds support for the new AST nodes. Dated conventions in the previous code printer are removed, such as the non-standard-conforming type specification `real*8`. The basic formatting and line-wrapping rules were updated to produce more-readable output code. Previously, lines were wrapped at the first encountered word-boundary or operator and continued at a fixed indentation on subsequent lines. For some of the longer equations present in the spacetime solvers, this produced code that was incredibly hard to inspect for correctness. A new set of rules now seeks to wrap lines at the boundary of a set of terms, and when possible avoids breaking a line inside of an innermost set of parentheses in an effort to keep array indices together on the same line. Line continuations made for assignments and declarations will now align to the start of the right-hand side in

the first line of the statement to improve readability. For the previously stated concerns, the use of `merge` was also removed for `Piecewise` input. For better compatibility with `Flash-X`, floating-point literals are now no longer expressed in a `X.YdZ` format; `Flash-X` requires use of `real` and `X.YeZ` notation and sets the default `real` type at compile time (defaults to double-precision).

The new AST nodes and code printing capabilities are demonstrated in code snippets 5.6–5.7. These examples illustrate the creation of a subroutine for removing the trace from a rank-two tensor as shown in Eq. (5.26). In code snippet 5.6, the input and output tensor are defined and added as parameters to the subroutine. A `use` statement for a `tensor_indexing` module is added next; this module will be assumed to define the named integer constants representing the numeric values of the array indices present in the tensor expression replacement rules. The body of the subroutine includes an indexed for the trace of the tensor, and a non-indexed expression removing this trace in the output tensor. Finally, this subroutine is passed to the Fortran code printer for parsing and formatting. `Trace` and `Inv` are specialized `TensorOperator` objects; `Trace` produces a `SymbolicScalar` representing the trace of the tensor, while `Inv` produces a `SymbolicTensor` for the provided tensor's inverse by flipping its index configuration.

The generated code is displayed in code snippet 5.7. The code printer generates the argument list in the provided order of the input and output variable declarations. Parsing of the subroutine's body identifies one locally-used intermediate variable that no declaration was provided for, `trX`, and generates the necessary variable declaration. The first assignment illustrates the output form of an indexed symbolic tensor expression and demonstrates the new line wrapping and indentation rules. The final assignment shows the elemental array operation and assignment generated by the non-indexed symbolic tensor expression.

```
# Tensor indices
i, j = tensor_indices("i,j", spatial)

# Symmetric in first two ranks
sym = Symmetries(Symmetric(0, 1))

# Input/Output tensors
X_ll = SymbolicTensor("X", [Covariant]*2, sym, spatial=True)

gam_ll = SymbolicTensor("gam", [Covariant]*2, sym, spatial=True)
gam_uu = Inv(gam_ll)

# Create the subroutine
subroutine = ExtendedSubroutine("remove_trace",
   # Input/output parameters
   params_decls = [
      TensorDeclaration(X_ll, real, [intent_inout]),
      TensorDeclaration(gam_ll, real, [intent_in]),
      TensorDeclaration(gam_uu, real, [intent_in]),
   ],

   # Module defining integer tensor indices
   uses = [use("tensor_indexing")],

   # Body of the subroutine
   body = [
      ScalarAssignment(
         Trace(X_ll),                    # LHS
         gam_uu(i, j) * X_ll(-i, -j),  # RHS
         comment="Trace of X"
      ),
      TensorAssignment(
         X_ll,                                              # LHS
         X_ll - Rational(1, 3) * gam_ll * Trace(X_ll), # RHS
         comment="Remove trace from X",
      ),
   ]
)

# Print the subroutine
print(fortrancode(subroutine))
```

Code Snippet 5.6: Generating a subroutine for a symbolic tensor expression

```
subroutine remove_trace(X_LL, gam_LL, gam_UU)
   use tensor_indexing

   implicit none

   real, dimension(XX:ZZ), intent(inout) :: X_LL
   real, dimension(XX:ZZ), intent(in) :: gam_LL
   real, dimension(XX:ZZ), intent(in) :: gam_UU

   real :: trX


   ! Trace of X
   trX = X_LL(XX)*gam_UU(XX) + 2*X_LL(XY)*gam_UU(XY) + &
         2*X_LL(XZ)*gam_UU(XZ) + X_LL(YY)*gam_UU(YY) + &
         2*X_LL(YZ)*gam_UU(YZ) + X_LL(ZZ)*gam_UU(ZZ)

   ! Remove trace from X
   X_LL = -(1.0/3.0)*trX*gam_LL + X_LL
end subroutine remove_trace
```

Code Snippet 5.7: Formatted Fortran output produced by code snippet 5.6

Chapter 6

General Relativistic Solvers in `Flash-X`

The general relativistic solvers presented in chapters 3–5 are each implemented in `Flash-X`, a composable multi-physics software framework ideally suited for large-scale simulation of astrophysical events (Dubey et al., 2022). The architecture `Flash-X` readily enables the addition of new physics and numerical methods modules, and provides access to many numerical capabilities, such as adaptive mesh refinement, and utilities, such as parallel input-output (I/O). A custom build system offers flexibility in composing simulations by including different implementations of physics, numerical, and utility modules through commonly-defined interfaces.

This chapter will describe the implementation and testing of the general relativistic solvers and their supporting modules. The radiation transport module `GRM1` implements the neutrino M1 formulation in chapter 3. The hydrodynamics module `GRFD` implements the high-order finite-difference method presented in chapter 4. The new `Spacetime` physics module implements the Z4c and CCZ4 formulations with the code-generator as presented in chapter 5. A new time integration module provides a method-of-lines discretization as an alternative to the operator-split method that delegated time-integration responsibility to each physics solvers.

To the extent that it is possible, the implementation of each solver is designed to separate the physics, numerical methods, and runtime controls. This mimics model-view-controller and delegation design patterns (see Gamma et al. (1995) for more information on these and other design patterns). Runtime controls interact directly with `Flash-X` and serve as the entry points into the solvers for various tasks. These controls orchestrate and delegate

responsibility to the numerical methods to perform calculations. In turn, the numerical methods rely on specific physics calculations. Each level of this design is made as agnostic to the others as much as possible. This facilitates adding, removing, updating, and debugging individual features by isolating specific sets of functionality to separate pieces of the code. This modular design also provides increased adaptability, e.g. the neutrino finite-volume and finite-difference discretization implementations can be swapped with one another while still using the same runtime controls and physics implementations.

6.1 Time Integration

The `TimeAdvance` module in `Flash-X` manages the evolution of a simulation. Previously this module only implemented an operator-split method that consecutively passed control to each physics module to perform a single full time-step update. In this method, an update from the n-th to $(n+1)$-th time-step of size Δt takes the form

$$\begin{aligned}
\mathbf{V}_{\mathrm{hy}}^{n+1} &= \mathcal{G}_{\mathrm{hy}}\Big(t_n, \Delta t, \mathbf{V}_{\mathrm{hy}}^{n}, \mathbf{V}_{\mathrm{rt}}^{n}, \mathbf{V}_{\mathrm{sp}}^{n}, \ldots\Big), \\
\mathbf{V}_{\mathrm{rt}}^{n+1} &= \mathcal{G}_{\mathrm{rt}}\Big(t_n, \Delta t, \mathbf{V}_{\mathrm{hy}}^{n+1}, \mathbf{V}_{\mathrm{rt}}^{n}, \mathbf{V}_{\mathrm{sp}}^{n}, \ldots\Big), \\
\mathbf{V}_{\mathrm{sp}}^{n+1} &= \mathcal{G}_{\mathrm{sp}}\Big(t_n, \Delta t, \mathbf{V}_{\mathrm{hy}}^{n+1}, \mathbf{V}_{\mathrm{rt}}^{n+1}, \mathbf{V}_{\mathrm{sp}}^{n}, \ldots\Big),
\end{aligned} \tag{6.1}$$

where t_n is the time at the start of the time-step, and $\mathbf{V}$ and $\mathcal{G}$ are the variables and update function, respectively, with each solver denoted by the subscripts "hy" for hydrodynamics, "rt" for radiation transport, and "sp" for spacetime. The variables $\mathbf{U}$ are also not necessarily the evolved variables, e.g. in the non-relativistic hydrodynamics solvers in `Flash-X`, only the primitive fluid variables are tracked, and the update functions are assumed to provide the updated primitive variables at the new time-step. While this operator-split method

has worked well for non-relativistic problems and solvers that are not tightly-coupled together, the mixed time-levels have lead to numerical instabilities when applied to coupled hydrodynamics and spacetime solvers in an earlier version of `FLASH` (Pajkos, 2022).

As an alternative to the operator-split method, the new method-of-lines (MoL) time-integrator instead assumes the combined system of all solvers' evolution equations takes the form

$$\partial_t \mathbf{U} = \mathcal{F}_{\mathrm{hy}}(t, \mathbf{U}) + \mathcal{F}_{\mathrm{rt}}(t, \mathbf{U}) + \mathcal{F}_{\mathrm{sp}}(t, \mathbf{U}) + \ldots, \tag{6.2}$$

where $\mathbf{U}$ represented the combined evolved variables from all solvers, and the $\mathcal{F}$ terms represent the right-hand sides (RHSs) of the evolution equations for each solver (as denoted by their subscripts). The MoL time-integrator provides separate explicit Runge-Kutta (ERK), implicit-explicit (IMEX), and multi-rate (MR) methods for discretizing Eq. (6.2).

This section will present each of these methods as used in the MoL time-integrator. All methods will describe a multi-stage process for updating the evolved variables $\mathbf{U}^n$ at a time t_n to their values $\mathbf{U}^{n+1}$ at a time $t_{n+1} = t_n + \Delta t$, where Δt is the size of the time-step.

6.1.1 Explicit Runge-Kutta

Explicit time-integration methods compute intermediate and final states exclusively in terms of the current known state of the evolved variables. The multi-stage methods used by the ERK time-integrator are based on the Runge-Kutta (RK) method (Runge, 1895; Kutta,

1901) and are described by Butcher tableau (Butcher, 1963) of the form

$$\begin{array}{c|c} \mathbf{c}^{\mathrm{E}} & \mathbf{A}^{\mathrm{E}} \\ \hline & \mathbf{b}^{\mathrm{E}} \end{array}, \tag{6.3}$$

where the superscript "E" denotes these are for explicit methods, the matrix $\mathbf{A}^{\mathrm{E}}$ with elements a_{ij}^{E} provides the weights for determining the intermediate stage updates, the column-vector $\mathbf{c}^{\mathrm{E}}$ with elements $c_i^{\mathrm{E}} = \sum_j a_{ij}^{\mathrm{E}}$ determines the offset between times t_n and t_{n_1} that the intermediate states represent, and the row-vector $\mathbf{b}^{\mathrm{E}}$ with elements b_j^{E} provides the weights for the final linear combination of the intermediate updates to integrate the the evolved variables through the full time-step. For explicit methods, the matrix $\mathbf{A}^{\mathrm{E}}$ will be strictly lower-triangular; each row i represents an intermediate state, so only currently known intermediate states with $j < i$ can be used in the update. For an arbitrary s-stage explicit method, these linear combinations are

$$\begin{aligned} \bar{\mathbf{U}}^i &= \mathbf{U}^n + \Delta t \sum_{j=1}^{i-1} a_{ij}^{\mathrm{E}} \mathcal{F}^{\mathrm{E}}\Big(t_n + c_j^{\mathrm{E}} \Delta t, \bar{\mathbf{U}}^j\Big), \\ \mathbf{U}^{n+1} &= \bar{\mathbf{U}}^n + \Delta t \sum_{j=1}^{s} b_j^{\mathrm{E}} \mathcal{F}^{\mathrm{E}}\Big(t_n + c_j^{\mathrm{E}} \Delta t, \bar{\mathbf{U}}^j\Big), \end{aligned} \tag{6.4}$$

where $\bar{U}^i$ represents the i-th intermediate state of the evolved variables, and $\mathcal{F}^{\mathrm{E}}$ represents the combined evaluation of all RHS terms of all evolution equations.

The ERK time-integrator works with all explicit methods that take this form. New methods can be added simply by providing a Butcher tableau. See appendix G.1 for more information on the currently available methods in `Flash-X`.

6.1.2 Implicit-Explicit Methods

Explicit integration methods are not always suitable for every problem. If evolution equations contain terms with disparate timescales, choosing a practical time-step size that allows for stable explicit integration may not be possible. The terms that have the smallest timescales, possible orders of magnitude smaller than the other terms, are referred to as stiff, and typically must be integrated implicitly. For example, the discretized neutrino moment evolution equations have an advective timescale of $\mathcal{O}(\Delta x/c)$ and an interaction timescale of $\mathcal{O}(1/c\kappa)$, where the speed of light c has been included explicitly; the interaction timescale can be many orders of magnitude smaller than the advective timescale in optically thick regions when the opacity becomes large. Implicit time-integration can be computationally expensive, particularly when applied to partial differential equations that contain spatial derivatives of the evolved variables. A more efficient approach is the use of mixed implicit-explicit (IMEX) methods that couple an implicit method applied to the stiff terms with an explicit method applied to the non-stiff terms.

The IMEX time-integrator makes use of the methods presented in Ascher et al. (1997) and Pareschi & Russo (2005). These methods will use a set of Butcher tableau of the form

$$\begin{array}{c|c} \mathbf{c}^{\mathrm{I}} & \mathbf{A}^{\mathrm{I}} \\ \hline & \mathbf{b}^{\mathrm{I}} \end{array} \quad , \qquad \begin{array}{c|c} \mathbf{c}^{\mathrm{E}} & \mathbf{A}^{\mathrm{E}} \\ \hline & \mathbf{b}^{\mathrm{E}} \end{array}, \tag{6.5}$$

where the superscript "I" denotes the quantities specific to the implicit tableau, and the matrix and vector quantities represent the same stage-specific weights and offsets as their ERK counterparts. For these IMEX methods, the implicit methods will be limited to diagonally implicit methods, i.e. the matrix $\mathbf{A}^{\mathrm{I}}$ is lower-triangular with $a^{\mathrm{I}}_{ij} = 0$ for $j > i$. The MoL

form of the evolution equations is separated into non-stiff explicitly integrated terms and stiff implicitly integrated terms

$$\partial_t \mathbf{U} = \mathcal{F}^{\mathrm{E}}(t, \mathbf{U}) + \mathcal{F}^{\mathrm{I}}(t, \mathbf{U}), \tag{6.6}$$

where $\mathcal{F}^{\mathrm{I}}$ represented the combined evaluation of stiff source terms. The linear combinations for the intermediate stage and final updates are

$$\begin{aligned} \bar{\mathbf{U}}^i &= \mathbf{U}^n + \Delta t \sum_{j=1}^{i-1} a_{ij}^{\mathrm{E}} \mathcal{F}^{\mathrm{E}}\left(t_n + c_j^{\mathrm{E}} \Delta t, \bar{\mathbf{U}}^j\right) + \Delta t \sum_{j=1}^{i} a_{ij}^{\mathrm{I}} \mathcal{F}^{\mathrm{I}}\left(t_n + c_j^{\mathrm{I}} \Delta t, \bar{\mathbf{U}}^j\right), \\ \mathbf{U}^{n+1} &= \bar{\mathbf{U}}^n + \Delta t \left[\sum_{j=1}^{s} b_j^{\mathrm{E}} \mathcal{F}^{\mathrm{E}}\left(t_n + c_j^{\mathrm{E}} \Delta t, \bar{\mathbf{U}}^j\right) + b_j^{\mathrm{I}} \mathcal{F}^{\mathrm{I}}\left(t_n + c_j^{\mathrm{I}} \Delta t, \bar{\mathbf{U}}^j\right) \right]. \end{aligned} \tag{6.7}$$

Each stage will require an implicit update when $a_{ii}^{\mathrm{I}} \neq 0$. Currently, the only stiff terms in the combined evolution equations are the neutrino-matter interaction terms, so responsibility of the implicit update is delegated to the neutrino radiation transport solver to exploit the structure of the update equations presented in chapter 3.

The IMEX time-integrator works with all methods that take this form. Similarly to the ERK methods, new IMEX methods can be added simply by providing a Butcher tableau. See appendix G.2 for more information on the currently available methods in `Flash-X`.

6.1.3 Multi-Rate Methods

Some problem may have evolution equations that contain terms with faster timescales than the non-stiff explicitly integrated terms, but slower timescales than the stiff terms. In these situations, these semi-stiff terms may not be ideal for implicit methods. Additionally, a subset

of the evolution equations may require a higher-order time-integration method or smaller time-step size to maintain numerical stability, but evolving the remaining equations with the same method or time-step might be too computationally expensive to be practical. For these situations, a multi-rate (MR) method that operates on a slower and faster timescales separately with explicit or IMEX methods can be an ideal solution.

The MR time-integrator is based on the implicit-explicit multi-rate methods presented in Chinomona & Reynolds (2021). In this method, the evolution equations take the form

$$\partial_t \mathbf{U} = \mathcal{F}^{\mathrm{E}}(t, \mathbf{U}) + \mathcal{F}^{\mathrm{I}}(t, \mathbf{U}) + \mathcal{F}^{\mathrm{F}}(t, \mathbf{U}), \tag{6.8}$$

where the superscript "F" denotes terms that will be integrated at the faster timescale; the implicit (I) and explicit (E) terms will be integrated at the slower timescale. The faster timescale method will evolve a modified evolution equation at each of the slower timescale method's stages

$$\mathbf{V}'(\theta) = \mathcal{F}^{\mathrm{F}}(\theta, \mathbf{V}(\theta)) + \mathcal{H}(\theta), \tag{6.9}$$

where θ is a new time-coordinate specific to the fast integration method, $\mathbf{V}$ represents the variables evolved by the fast method as a function of θ (this is not the same as the similarly named vector in the operator-split method earlier in this chapter) with the prime indicating the derivative with respect to θ, and $\mathcal{H}(\theta)$ represents the contribution of terms from the slow integration method. This term intentionally does not include a dependence on $\mathbf{U}$ to emphasize that only θ changes in its evaluation, as it typically interpolates the results from

the slow method in time. The slow IMEX methods will use a Butcher tableau of the form

$$\begin{array}{c|cc|cc} \mathbf{c}^{\mathrm{S}} & \mathbf{A}^{\mathrm{I},\{k\}} & \mathbf{0}^{\{k\}} & \mathbf{A}^{\mathrm{E},\{k\}} & \mathbf{0}^{\{k\}} \\ 1 & \mathbf{b}^{\mathrm{I},k} & 0^{\{k\}} & \mathbf{b}^{\mathrm{E},\{k\}} & 0^{\{k\}} \\ \hline & \mathbf{b}^{\mathrm{I},\{k\}} & 0^{\{k\}} & \mathbf{b}^{\mathrm{E},\{k\}} & 0^{\{k\}} \end{array}, \tag{6.10}$$

where the superscript "S" denotes quantities specific to the slow integration method, and the additional row with the $\mathbf{b}$ vectors provide a stiffly-accurate form, and the superscripts $\{k\}$ denote a set of $0 \leq k \leq k_{\max}$ tableau that are used in time-interpolation of the slow-method's intermediate states. For an s-stage IMEX method used for the slower timescale integration, its tableau will be extended to $2s \times 2s$ to utilize the solve-decoupled method proposed in Chinomona & Reynolds (2021), which alternates between "slow" and "fast" stages. Slow-stage updates are performed in the same manner as the IMEX method's in Eq. (6.7), and fast-stage updates, which are currently limited to explicit methods, are performed the same as in the ERK method's update in Eq. (6.4), but applied to Eq. (6.9) written in the form for the i-th slow integration stage

$$\begin{aligned} \mathbf{V}'(\theta) &= \Delta c_i^{\mathrm{S}} \mathcal{F}^{\mathrm{F}}\Big(t_n + c_{i-1}^{\mathrm{S}}\Delta t + \Delta c_i^{\mathrm{S}}\theta, \mathbf{V}(\theta)\Big) + \mathcal{H}(\theta) \\ \mathcal{H}(\theta) &= \sum_{j=1}^{i-1}\sum_{k=0}^{k_{\max}} a_{ij}^{\mathrm{E},\{k\}}\left(\frac{\theta}{\Delta t}\right)^k \mathcal{F}^{\mathrm{E}}\Big(t_n + c_j^{\mathrm{S}}\Delta t, \bar{\mathbf{U}}^j\Big) \\ &\quad + \sum_{j=1}^{i}\sum_{k=0}^{k_{\max}} a_{ij}^{\mathrm{I},\{k\}}\left(\frac{\theta}{\Delta t}\right)^k \mathcal{F}^{\mathrm{I}}\Big(t_n + c_j^{\mathrm{S}}\Delta t, \bar{\mathbf{U}}^j\Big) \\ \mathbf{V}(0) &= \bar{\mathbf{U}}^{i-1}, \quad \bar{\mathbf{U}}^i = \mathbf{V}(\theta), \quad 0 \leq \theta \leq \Delta t. \end{aligned} \tag{6.11}$$

The IMEX tableau used for the slow stages takes the form

$$\mathbf{A}^{\mathrm{E}} = \sum_{k=0}^{k_{\max}} \frac{1}{k+1} \mathbf{A}^{\mathrm{E},\{k\}}, \quad \mathbf{A}^{\mathrm{I}} = \sum_{k=0}^{k_{\max}} \frac{1}{k+1} \mathbf{A}^{\mathrm{I},\{k\}}. \tag{6.12}$$

The MR time-integrator works with all methods that take this form. Currently, only a third-order method is available for the slow timescale integration (see appendix G.3), but all ERK methods in appendix G.1 are available for the fast timescale integration.

6.2 GRM1

The Flash-X radiation transport physics module GRM1 implements the neutrino moment evolution equations and numerical methods presented in chapter 3. This module implements the public interface, i.e. accessible to other parts of Flash-X, of the radiation transport module, including initialization, calculation of implicit and explicit RHS terms, implicit updates of the interaction terms, and all pre- and post-time-step synchronization procedures. Following the design principles stated earlier in the chapter, all terms in the evolution equation, Eq. (3.101), are implemented as compact kernels operating on a single cell of data, which are then used in procedures operating on a single block or row of data in the grid. The implementation of the public interface serves as the entry point and runtime controls that interact with Flash-X for tasks such as accessing the grid and its iterators and performing I/O-related operations.

The remainder of this section will provide a comprehensive set of test problems used to validate the implementation of the GRM1 solver. For simplicity, these test problems will all use the uniform grid provided by the Flash-X UG grid module. All test problems will make

use of the third-order IMEX-ARK(3,4,3) time-integrator (see appendix G.2). As is standard practice for numerically evolving hyperbolic equations, the time-steps will be limited by the Courant-Friedrichs-Lewy (CFL) number to provide numerical stability (Toro, 2009)

$$\Delta t = C_{\text{cfl}} \min_i \left(\frac{\Delta x^i}{|\lambda|} \right). \tag{6.13}$$

Frequently, $|\lambda| = c$ when there are regions were neutrinos are free streaming. One- and two-dimensional simulations use $C_{\text{cfl}} = 0.5$ while three-dimension simulations will use $C_{\text{cfl}} = 0.3$. Unless necessary for a particular test, only a single neutrino species and frequency will be used. All test problems will also make use of the finite-volume discretization with a layer of two guard-cells unless stated otherwise.

6.2.1 Radiation Beam

These tests utilize fixed-source radiation beams in vacuum to isolate and test the treatment of the spatial advection terms. Three-dimensional simulations are performed for both flat and curved spacetimes. These tests all verify the solver's free-streaming advection capabilities, and produce results that compare favorably with similar tests performed in Foucart et al. (2015); Weih et al. (2020); Radice et al. (2022).

6.2.1.1 Flat Spacetime

This set of tests uses a static Minkowski background metric in a Cartesian coordinate basis. Separate simulations for on- and off-axis beam configurations test the free-streaming capabilities of the solver. Each configuration will also be used to compare the less- and more-dissipative forms of the generalized minmod limiter. All simulations use a grid dis-

cretized into cells of size $\Delta x = \Delta y = \Delta z = 0.05$, with outflow boundary conditions set at the extents of the domain.

The on-axis beam configuration sets a fixed-source beam directed along the x-axis in the region $x \leq 0.1$ and $0.05 \leq y, z \leq 0.15$. The energy and momentum densities in the beam region are set to $E = 1$ and $F^i = (E, 0, 0)$, respectively. The simulations are initialized to $E = F^i = 0$ outside of this region, and then evolved until a time $t = 0.4$. This configuration is ran separately for the $\theta = 1, 2$ limits of the generalized minmod limiter. The comparison of the results for this test are displayed in fig. 6.1. In both sets of results, the beams maintain their shape and direction, and propagate at the expected speed (in units of $c = 1$). There

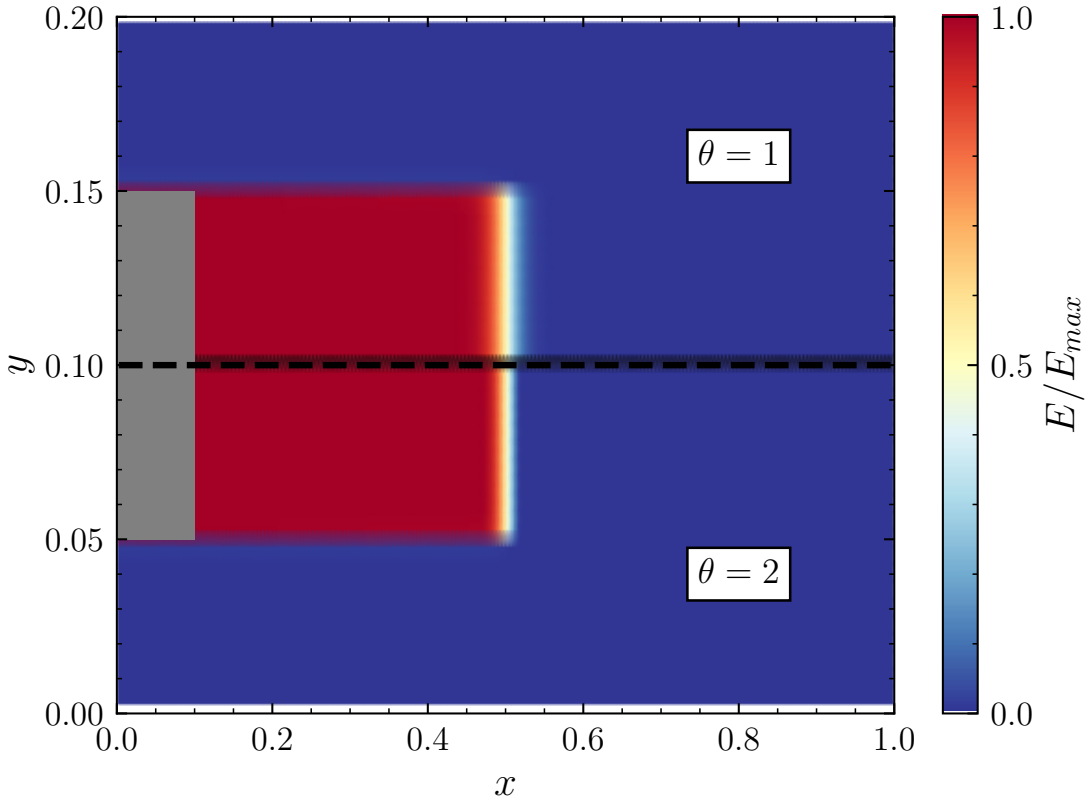

Figure 6.1: Fixed-source radiation beam (gray-shaded region) directed along the x-axis. Results for the normalized energy density are shown in the xy-plane at a time $t = 0.4$. The results for the more-dissipative $\theta = 1$ and less-dissipative $\theta = 2$ limiters are displayed in the upper and lower halves, respectively, with the black-dashed line separating the regions.

is a slight broadening along the leading edge of the beam, with the more-dissipative limiter exhibiting the most broadening.

The off-axis beam configuration translate the on-axis configuration's fixed source to $x = 0.1$ and rotates it by 45° in the xy-plane to lie along the line $x = y$. Again, the simulations are initialized to $E = F^i = 0$ outside of this region, and then evolved until a time $t = 0.3$. The comparison of the results for this test ran for the limiter's $\theta = 1, 2$ cases are displayed in fig. 6.2. In both sets of results, the beams maintain their expected direction and speed. However, the broadening previously limited to the leading edge of the beam in the on-axis

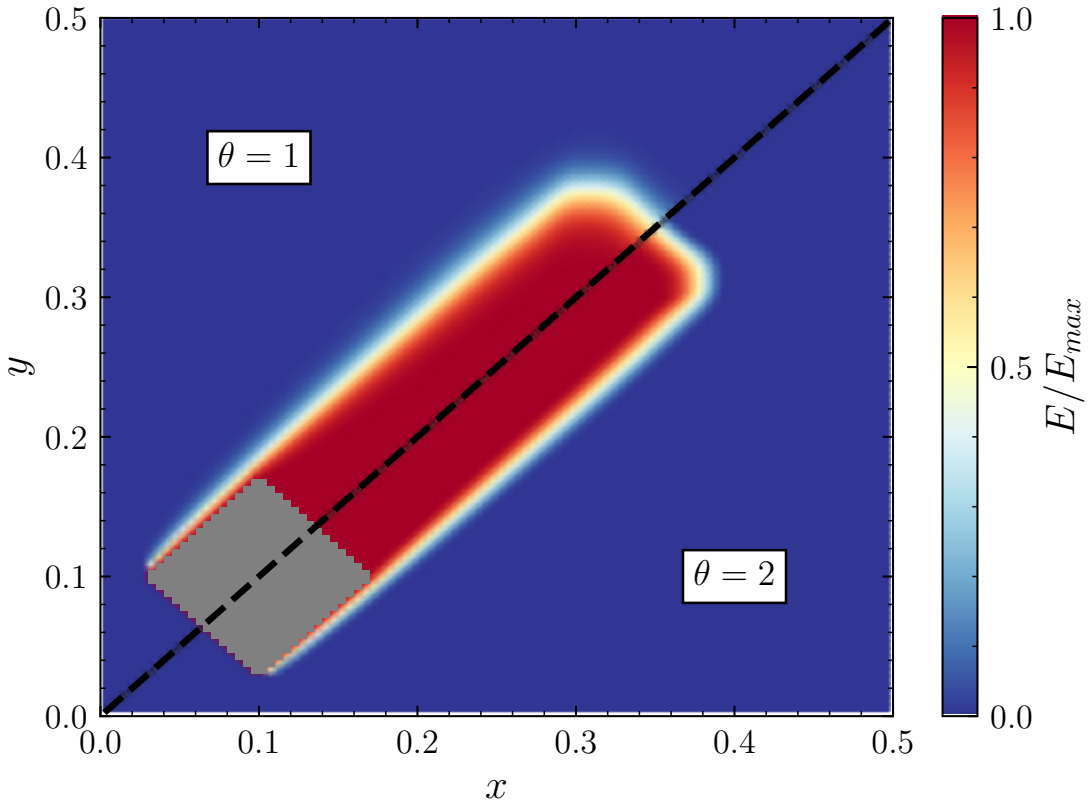

Figure 6.2: Fixed-source radiation beam (gray-shaded region) directed along the line $x = y$. Results for the normalized energy density are shown in the xy-plane at a time $t = 0.3$. The results for the more-dissipative $\theta = 1$ and the less-dissipative $\theta = 2$ limiters are displayed in the upper left and lower-right, respectively, with the black-dashed line separating the regions.

configuration now extends to all sides. This effect is again more pronounced with the more-dissipative $\theta = 1$ limiter.

6.2.1.2 Schwarzschild Spacetime

The second set of radiation beam tests examine the `GRM1` solver's treatment of advection in a curved spacetime. For this purpose, a Schwarzschild black hole with a mass $M = 1$ is placed at the origin, and described by a Kerr-Schild Cartesian coordinate basis. This choice of basis produces a non-unit lapse and metric components, and non-zero shift and extrinsic curvature components; this proves useful in verifying the correct implementation of terms containing these quantities in the evolution equations. In Kerr-Schild Cartesian coordinates, the Schwarzschild lapse, shift, spatial metric, and extrinsic curvature are (Baumgarte & Shapiro, 2010)

$$\begin{aligned} \alpha = \left(1 + \frac{2M}{r}\right)^{-\frac{1}{2}}, \quad \gamma_{ij} &= \eta_{ij} + \frac{2M}{r}\ell_i\ell_j \\ \beta^i = \frac{2M\alpha^2}{r}\ell^i, \qquad\qquad K_{ij} &= \frac{2M\alpha}{r}\left[\eta_{ij} - \left(2 + \frac{M}{r}\right)\ell_i\ell_j\right] \\ r^2 = x^2 + y^2 + z^2, \quad \ell^i = \ell_i &= \frac{x^i}{r}, \end{aligned} \tag{6.14}$$

where $\eta_{ij} = \text{diag}(1, 1, 1)$ is Minkowski metric in Cartesian coordinates.

While three-dimensional simulations are used for these tests, they are confined to the region of $x, y \geq 0$ and a single-layer of cells straddling $z = 0$. The domain is discretized into cells of size $\Delta x = \Delta y = \Delta z = 0.1$. Fixed source beams are added along the y-axis and oriented such that $\alpha F^j - \beta^j$ is parallel to the x-axis. Two separate beam locations are used, one located at $7 \leq y \leq 8$ and one located closer to the black hole at $3 \leq y \leq 4$. Both simulations set the beam's fixed-source to an energy density of $E = 1$ and the magnitude

of the momentum density to $\gamma_{ij}F^iF^j = E^2$. Outflow boundaries are used for the radiation quantities, while the metric quantities are set to their analytic values at the boundaries. Each simulation is then evolved until a time $t = 15$, allowing the beam to propagate through the computational domain.

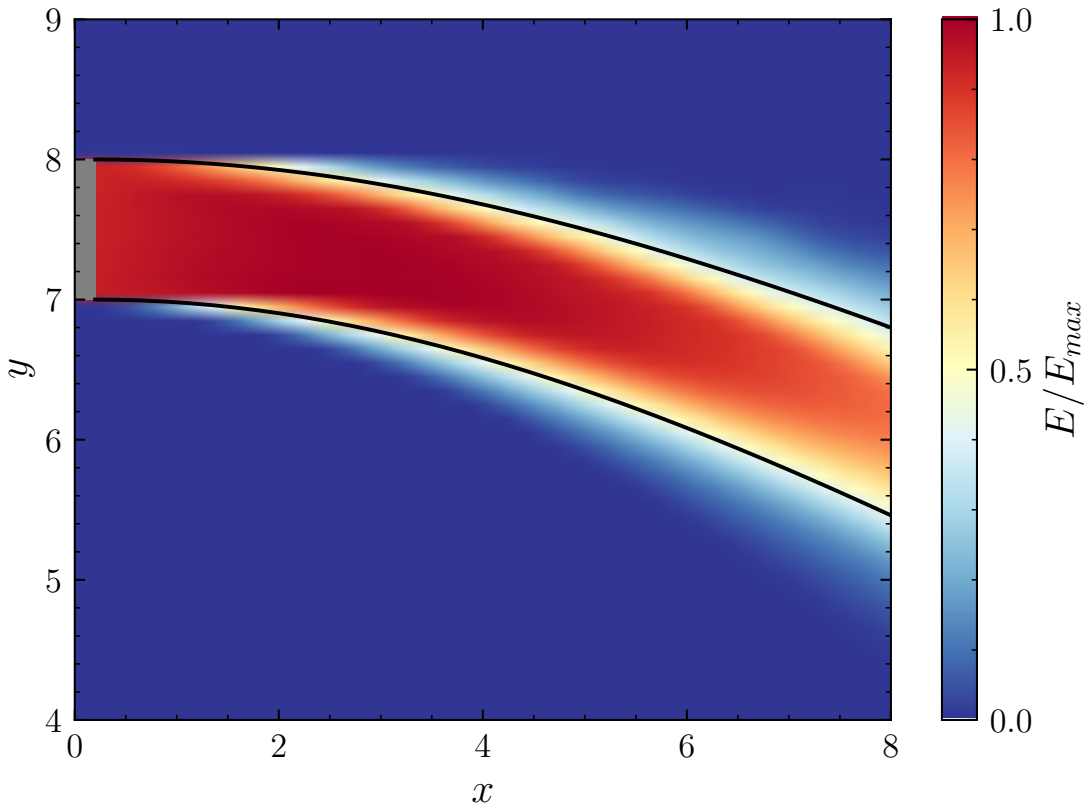

Figure 6.3: Fixed-source radiation beam (gray-shaded region) located between a radius of $7 \leq r \leq 8$ along the y-axis outside of a unit mass Schwarzschild black hole. The beam is oriented in the positive x-direction, and evolved until a time $t = 15$. The results show the normalized energy density compared to the bounding null geodesics (solid black lines) indicating the expected path of the beam.

The results for the far and near beam simulations are displayed in figs. 6.3–6.4. In each figure, the beams are compared to the null geodesics emanating from boundaries of the fixed-source of each beam. In both cases, the beams closely follow these expected trajectories curving around the black hole. The beams diffuse slightly outside of these bounds due to

numerical treatment of the fluxes, e.g. cells that straddle the bounding geodesics allow for transport of energy outside of this region. These type of numerical artifacts are present in similar tests performed in Foucart et al. (2015); Radice et al. (2022).

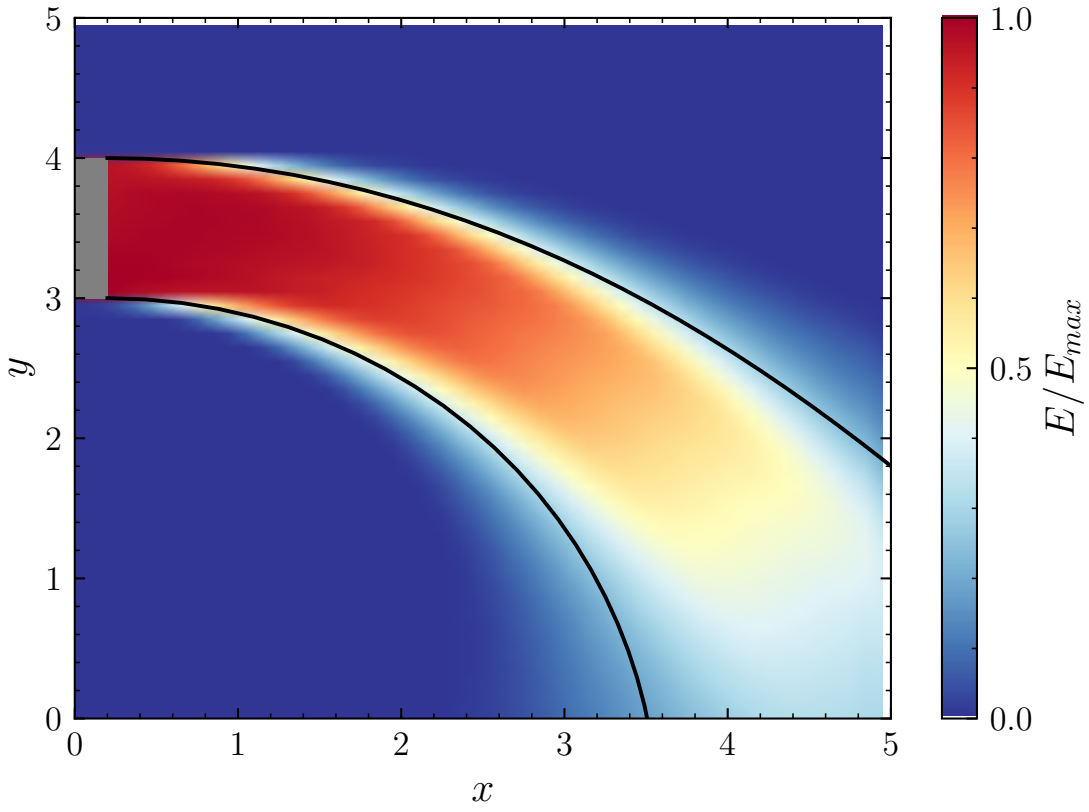

Figure 6.4: Fixed-source radiation beam (gray-shaded region) located between a radius of $3 \leq r \leq 4$ along the y-axis outside of a unit mass Schwarzschild black hole. The beam is oriented in the positive x-direction, and evolved until a time $t = 15$. The results show the normalized energy density compared to the bounding null geodesics (solid black lines) indicating the expected path of the beam.

6.2.2 Radiating Sphere

A radiating homogeneous sphere is used to examine the `GRM1` solver's ability to transition between the optically thin and thick regimes. A uniform density sphere with a radius R is placed at the origin and assigned a constant equilibrium energy density, $J_{\rm eq}$, and absorption

opacity, $\kappa_{\text{abs.}}$, and evolved until reaching a steady-state. This test problem proves useful in examining the validity of the M1 closure by comparison to the analytic steady-state solution presented in Smit et al. (1997)

$$
\begin{aligned}
f(r,\mu) &= J_{\text{eq.}}\Big[1 - e^{-\kappa_{\text{abs.}}\, s(r,\mu)}\Big] \\
s(r,\mu) &= \begin{cases} Rg(r,\mu) + r\mu, & r < R \quad \text{and} \quad -1 < \mu < 1 \\ 2Rg(r,\mu), & r \geq R \quad \text{and} \quad \sqrt{1 - \left(\frac{R}{r}\right)^2} < \mu < 1 \end{cases} \\
g(r,\mu) &= \sqrt{1 - \left(\frac{r}{R}\right)^2 \left(1 - \mu^2\right)}
\end{aligned}
\tag{6.15}
$$

where $\mu = \cos\theta$. The first few projected co-moving frame moments can then be obtained directly by using the distribution function in Eq. (6.15) in Eqns. (3.17)–(3.19).

Two separate simulations are performed for optically thin and thick spheres. For the optically thin sphere, the equilibrium energy density and absorption opacity are set to the values used in Smit et al. (1997), $J_{\text{eq.}} = 0.8$ and $\kappa_{\text{abs.}} = 4$. For the optically thick sphere, these values are set to the ones used in Abdikamalov et al. (2012), $J_{\text{eq.}} = 10$ and $\kappa_{\text{abs.}} = 250$. Both simulations place a sphere with a radius $R = 1$ at the origin on a one-dimensional spherically symmetric domain that extends out to a radius of $r_{\max} = 3$. The radial axis is discretized into 500 cells (for a cell-size of $\Delta r = 0.006$), which ensures that the corrective factor in Eq. (3.128) is used in the optically thick case only. Each simulation initializes the energy density to a constant value inside the sphere with a subsequent $1/r^2$ drop-off outside

$$
E = J_{\text{eq.}} \times \begin{cases} 1, & r \leq R \\ \left(\frac{R}{r}\right)^2, & r > R \end{cases}, \quad F_r = E \times \begin{cases} 10^{-10}, & r \leq R \\ \frac{1}{2}, & r > R \end{cases}. \tag{6.16}
$$

A reflective boundary condition is placed at $r = 0$, and an outflow boundary condition at $r = 3$. Each simulation is evolved past reaching a steady-state until a time $t = 5$.

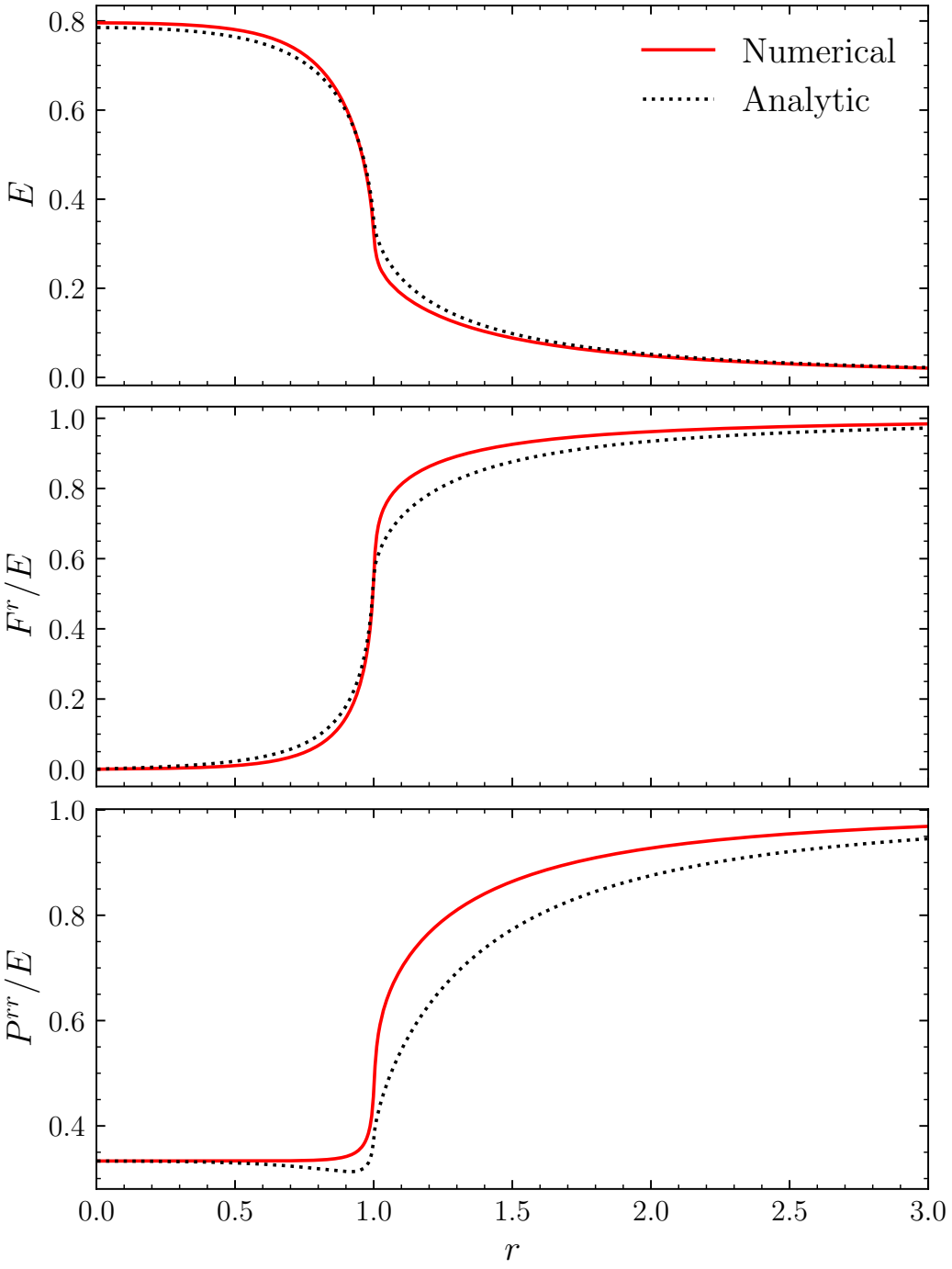

Figure 6.5: Results for the optically thin radiating sphere ($J_{\text{eq.}} = 0.8$ and $\kappa_{\text{abs.}} = 4$) at a time $t = 5$. The numerical (red solid lines) and analytic (black dotted lines) solutions for the energy density, momentum density, and pressure are compared.

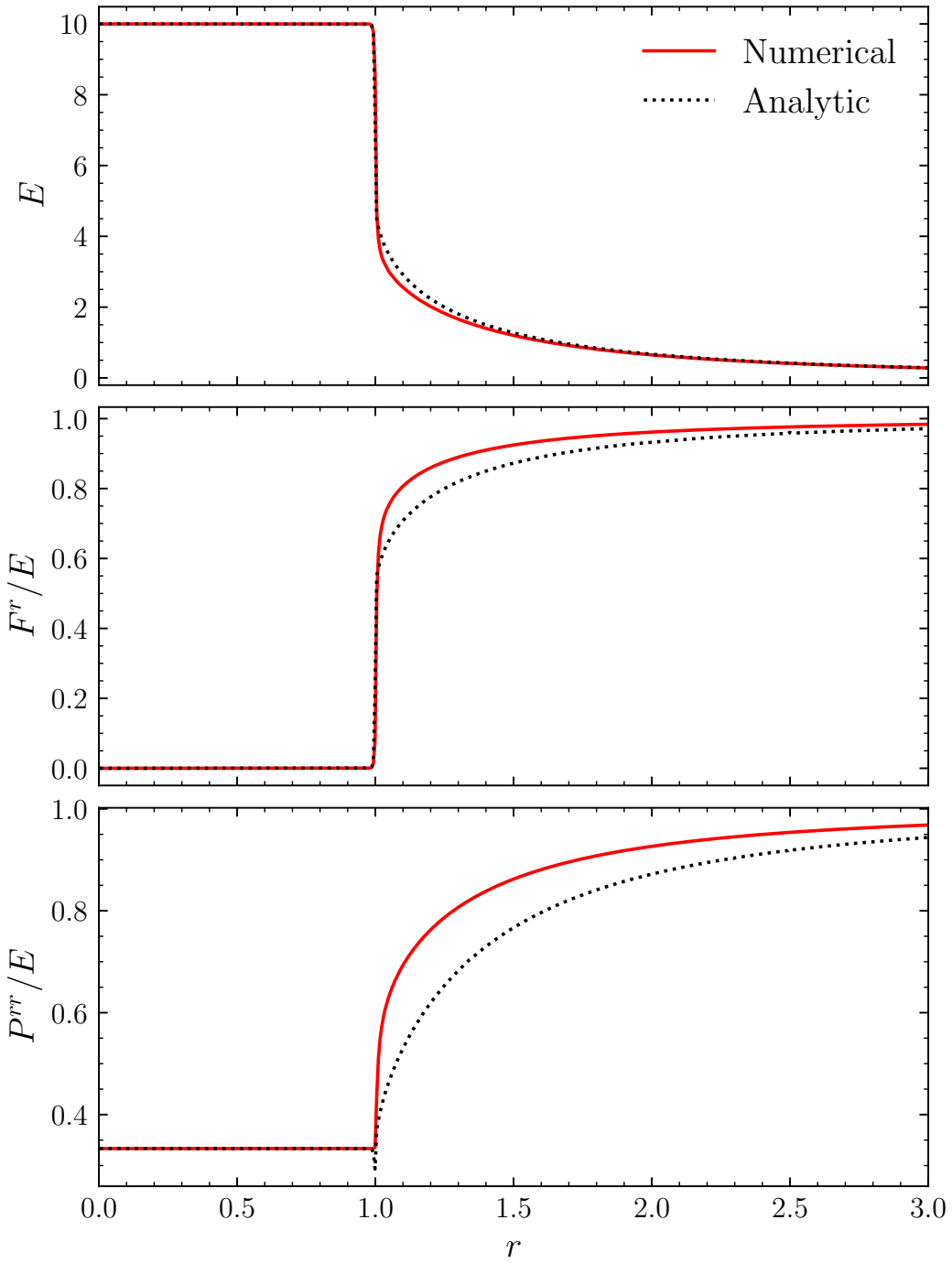

Figure 6.6: Results for the optically thick radiating sphere ($J_{\text{eq.}} = 10$ and $\kappa_{\text{abs.}} = 250$) at a time $t = 5$. The numerical (red solid lines) and analytic (black dotted lines) solutions for the energy density, momentum density, and pressure are compared.

The results for the optically thin and thick simulations are shown in figs. 6.5–6.6. In each case, the solutions closely match their analytic values in the optically thin and thick

limits, but deviate in the intermediate regime just outside of the sphere. These results are on par with similar tests performed with M1 radiation transport solvers Foucart et al. (2015); O'Connor (2015); Weih et al. (2020); Radice et al. (2022); Cheong et al. (2023). The deviation of the results when not fully in the optically thin or thick limit is expected when using the Minerbo Eq. (3.107) or other similar closure relations, as they are unable to produce the exact closure relation of the analytic solution (Murchikova et al., 2017).

6.2.3 Shadow Casting

A series of shadow casting test problems are used as a further test of the `GRM1` solver's ability to transition between the optically thin and thick regimes. These tests will place a completely absorbing medium outside of a radiative source, and examine the occluded downstream region. M1 radiation transport solvers excel at capturing the shadows cast in these regions, while more-approximate methods fail to resolve these features; see Hayes & Norman (2003) for a comparison of the ability of two-moment and flux-limited diffusion solvers to resolve shadows.

The first test extends the previous on-axis radiation beam test to include a completely absorbing sphere placed downstream from the fixed-beam source. The fixed-source is confined to the region $x \leq 0.1$ and $-0.1 \leq y, z \leq 0.1$ and is directed along the x-axis, and its energy density is fixed at $E = 1$. A completely absorbing sphere with a radius 0.05 is placed downstream along the x-axis at $x = 0.3$; the sphere's absorption opacity is set to $\kappa_{\text{abs.}}$, while its emissivity is set to $\kappa_{\text{abs.}} J_{\text{eq.}} = 0$. The domain is discretized into cells of size $\Delta x = \Delta y = \Delta z = 0.005$, and outflow boundary conditions are enforced along at each boundary. The initial energy and momentum density are set to zero everywhere outside of the fixed-source beam. The simulation is evolved until a time $t = 0.5$, allowing the beam to

pass the sphere and exit the domain.

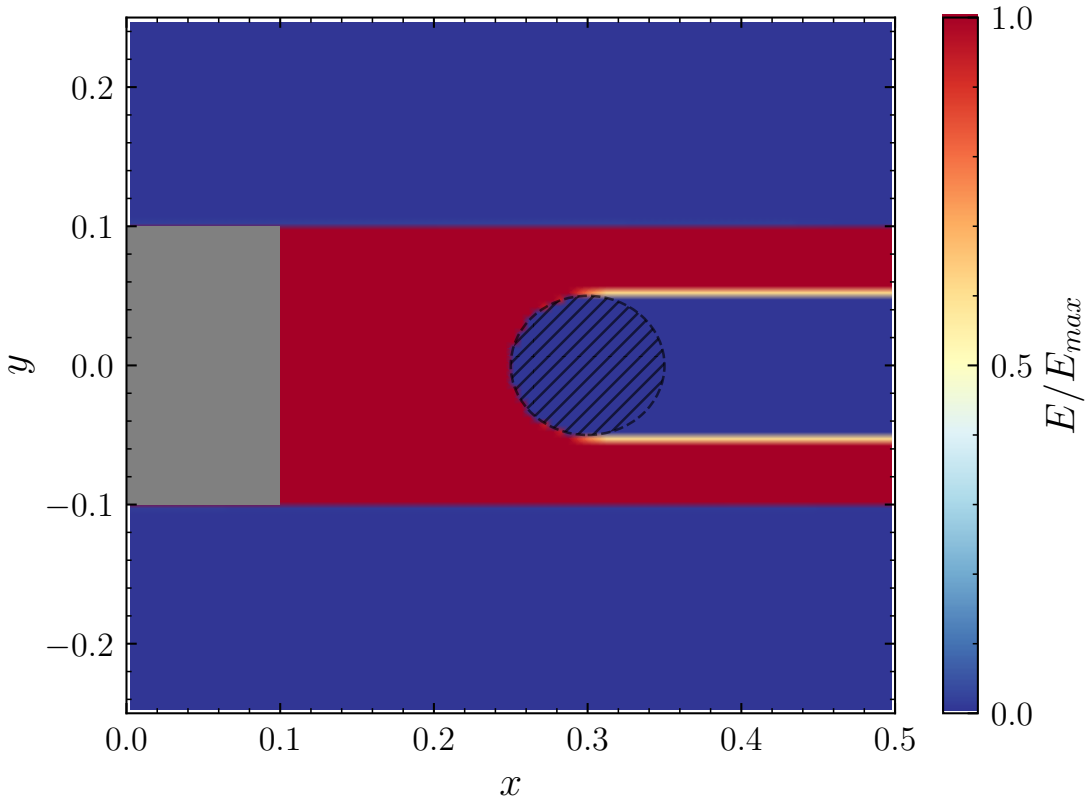

Figure 6.7: Results for the shadow cast by an absorbing sphere (black hatched region) placed in the path of a fixed-source beam (gray shaded region). The beam is evolved until a time $t = 0.5$ allowing it to fully pass by the sphere and exit the domain. A crisp shadow can be seen in the region downstream from the absorbing sphere.

The results for this test are displayed in fig. 6.7 and show a crisp shadow downstream from absorbing sphere. The marginal broadening of the edges of the beam adjacent to the occluded region result from the approximation of the sphere in the rectilinear grid, as the reconstruction stencils used to approximate the flux overlap past the sphere into the upstream beam near the top and bottom edges. However, the beam does not diffuse any further into this region as it continues to propagate through the domain.

A second test to assess the shadow-casting capabilities of the `GRM1` solver utilizes a radiating sphere as the radiative source. This test adopts a similar setup as the tests performed in

O'Connor & Couch (2018a); Cheong et al. (2023). The radiating sphere source with radius $R = 1.5$ is placed at the origin, and assigned an equilibrium energy density of $J_{\text{eq.}} = 1$ and a radially varying absorption opacity $\kappa_{\text{abs.}}(r) = 10 \exp\left[-(4r/R)^2\right]$. A completely absorbing sphere with a radius of $\bar{R} = 2$ is placed outside of the sphere on the x-axis at $x = 8$; the absorbing sphere is again assigned an absorption opacity $\bar{\kappa}_{\text{abs.}} = 10^6$ and the emissivity is forced to zero. The domain is discretized into cells of size $\Delta x = \Delta y = \Delta z = 0.075$, and outflow conditions are enforced at all boundaries. The simulation is evolved until a time $t = 15$, allowing the solution to reach a steady-state.

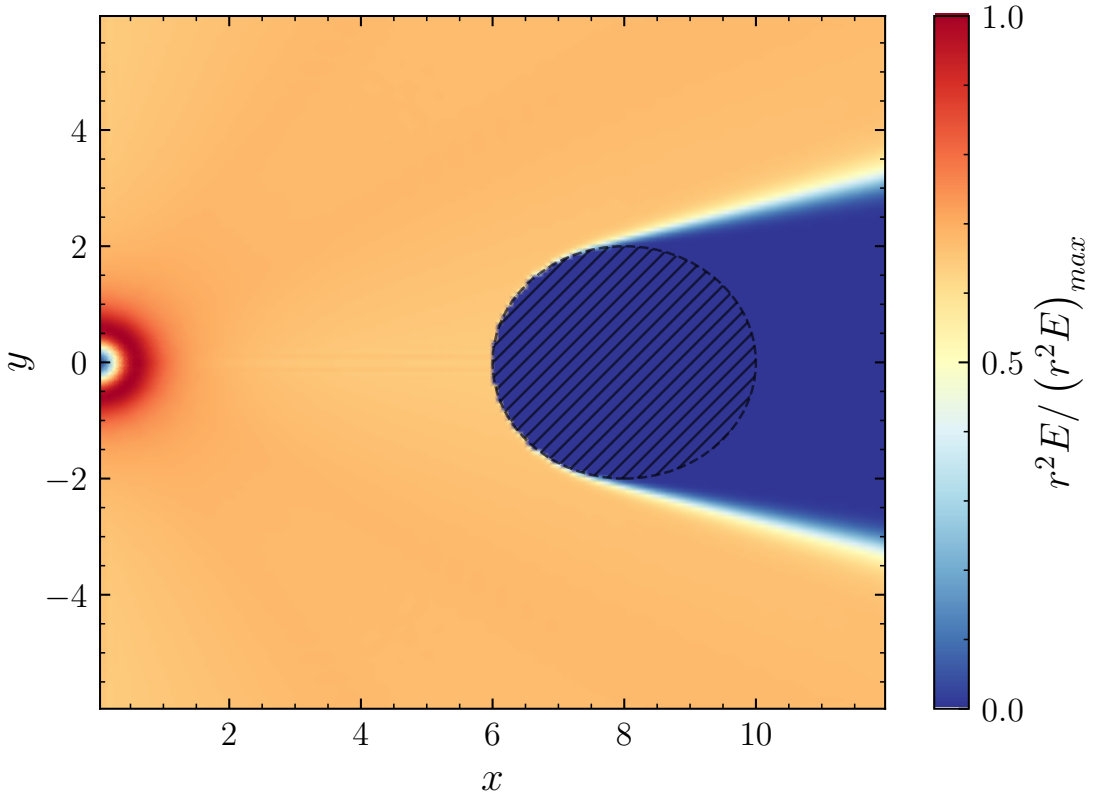

Figure 6.8: Results at the time $t = 15$ for the shadow cast by an absorbing sphere (black hatched region) placed outside of a radiating sphere located at the origin. A crisp shadow forms in the region occluded by the absorbing sphere and persists in the steady-state solution. The normalized densitized energy density displayed to highlight difference between the outward-propagating radiation and the shadowed formed behind the absorbing sphere.

Figure 6.8 displays the results for this test after the steady-state has been reached. As

with the prior test using the fixed-source beam in propagating in a uniform direction, the absorbing sphere correctly produces a shadow in the downstream region. Again, some broadening along the boundary between the free-streaming and occluded regions is present, but the radiation's propagation direction is maintained. These results agree with those of the similar tests performed in O'Connor & Couch (2018a); Cheong et al. (2023), and further validate the `GRM1` solver's treatment of advection and interactions.

6.2.4 Diffusion Limit

In the optically thick limit radiation behaves diffusively in the co-moving frame of the fluid. Since the radiation is fully trapped by the fluid in this limit, any advection observed in the Eulerian frame is the result of the motion of the fluid. A series of diffusion problems are used to assess both the corrections made to the hyperbolic fluxes in this limit and the retention of all non-linear source terms during the implicit update of the neutrino-matter interactions.

All diffusion tests will be performed in strictly-scattering media and a static Minkowski background spacetime. Under these conditions, the evolution of the radiation's energy density can be described by a diffusion equation (Pons et al., 2000; Radice et al., 2022; Cheong et al., 2023), which can be written in the form

$$\partial_t E = \tau_{\rm d} \nabla^2 E, \tag{6.17}$$

where $\tau_{\rm d} = 1/3\kappa$ is the diffusion timescale. Since these problems will exclusively use purely-scattering media, the opacity in the diffusion timescale will always be taken as the scattering opacity, $\kappa \equiv \kappa_{\rm iso}$. For verification purposes, each test will correspond to a problem that permits an analytic solution of Eq. (6.17). Cases for stationary and moving media will be

considered separately in this section; a second stationary medium will later be used for testing spatial convergence rates.

6.2.4.1 Stationary Medium

A stationary medium is used to validate the correction of the hyperbolic fluxes in the asymptotically optically thick limit. This test will use a point-like source as proposed in Pons et al. (2000) to produce steep gradients in the energy density;this would lead to incorrect numerical fluxes without the corrective factor in Eq. (3.128) being applied to the Riemann solver in Eq. (3.126). Assuming that the initial energy density is a point-like source at the origin

$$E(\vec{\mathbf{r}}, t = 0) = \delta^3(\vec{\mathbf{r}}), \tag{6.18}$$

where $\vec{\mathbf{r}}$ is the coordinate position vector. Solving Eq. (6.17) in a spherically symmetric spacetime via a Fourier transform yields the analytic solution

$$E(r, t) = \left(\frac{1}{3\tau_{\mathrm{d}} t}\right)^{\frac{3}{2}} \exp\left(-\frac{r^2}{4\tau_{\mathrm{d}} t}\right). \tag{6.19}$$

Pons et al. (2000) shows that the momentum density in the optically thick limit is then

$$F^r(r, t) = -\tau_{\mathrm{d}} \partial_r E(r, t) = \frac{r}{2t} E(r, t). \tag{6.20}$$

Two tests at different scattering opacities $\kappa = 10^2$ and $\kappa = 10^4$ are conducted. Each test is initialized at a time $t > 0$ from the analytic solution in Eqns. (6.19)–(6.20). One-dimensional spherically symmetric domains are discretized into cells of size $\Delta r = 0.01$; this guarantees that the optical depth for the $\kappa = 10^2$ case remains at $\tau = 1$ and that the fluxes

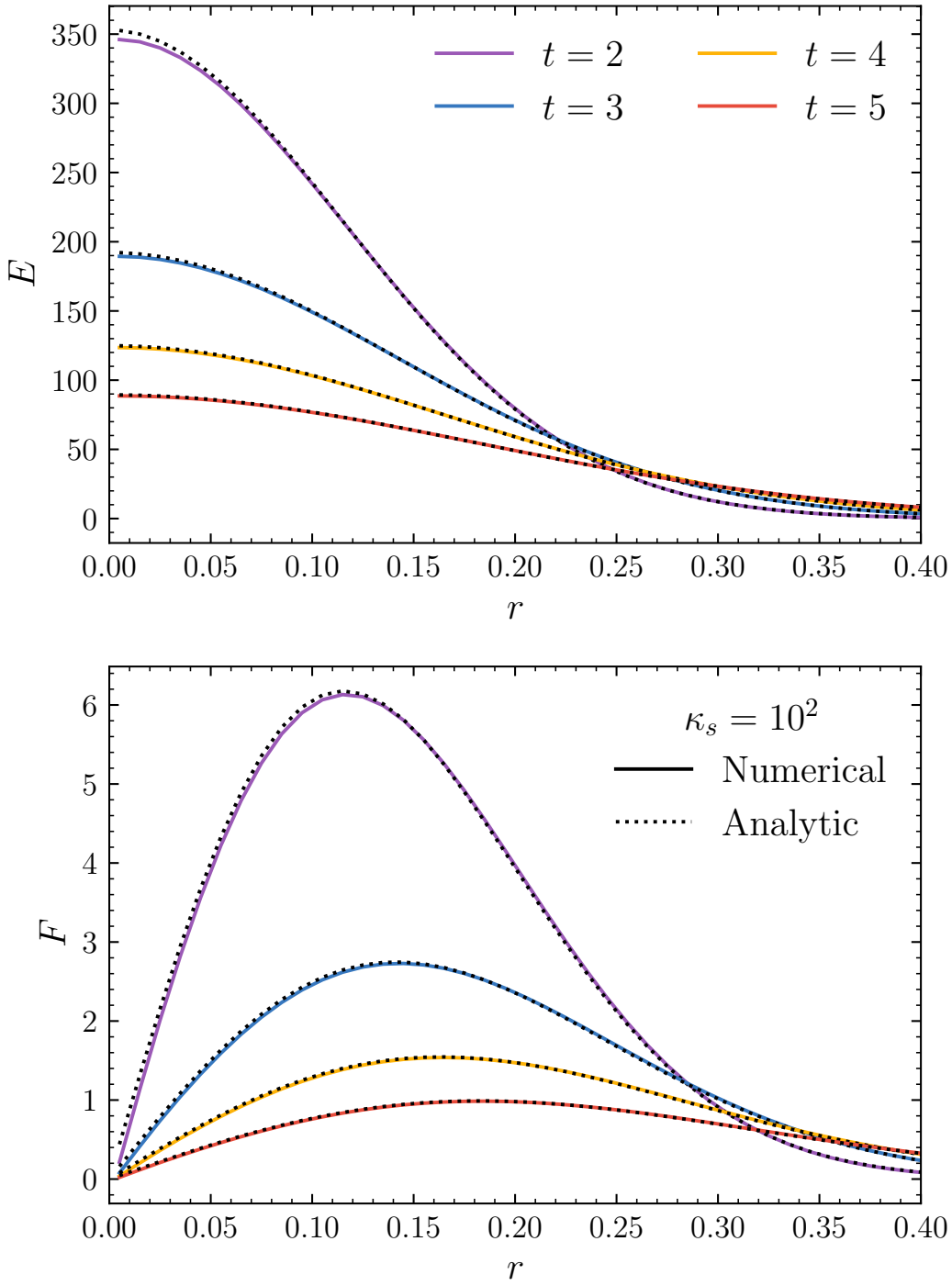

Figure 6.9: Results for the point-like source in a scattering medium with $\kappa = 10^2$. The numerical (solid lines) and analytic (dotted lines) solutions are compared at various times.

are not corrected for the optically thick limit, while the case for $\kappa = 10^4$ has an optical depth of $\tau = 10^2$, and the corrections are applied. Reflective boundary conditions are applied at

the origin, and outflow conditions are enforced at the outer boundary.

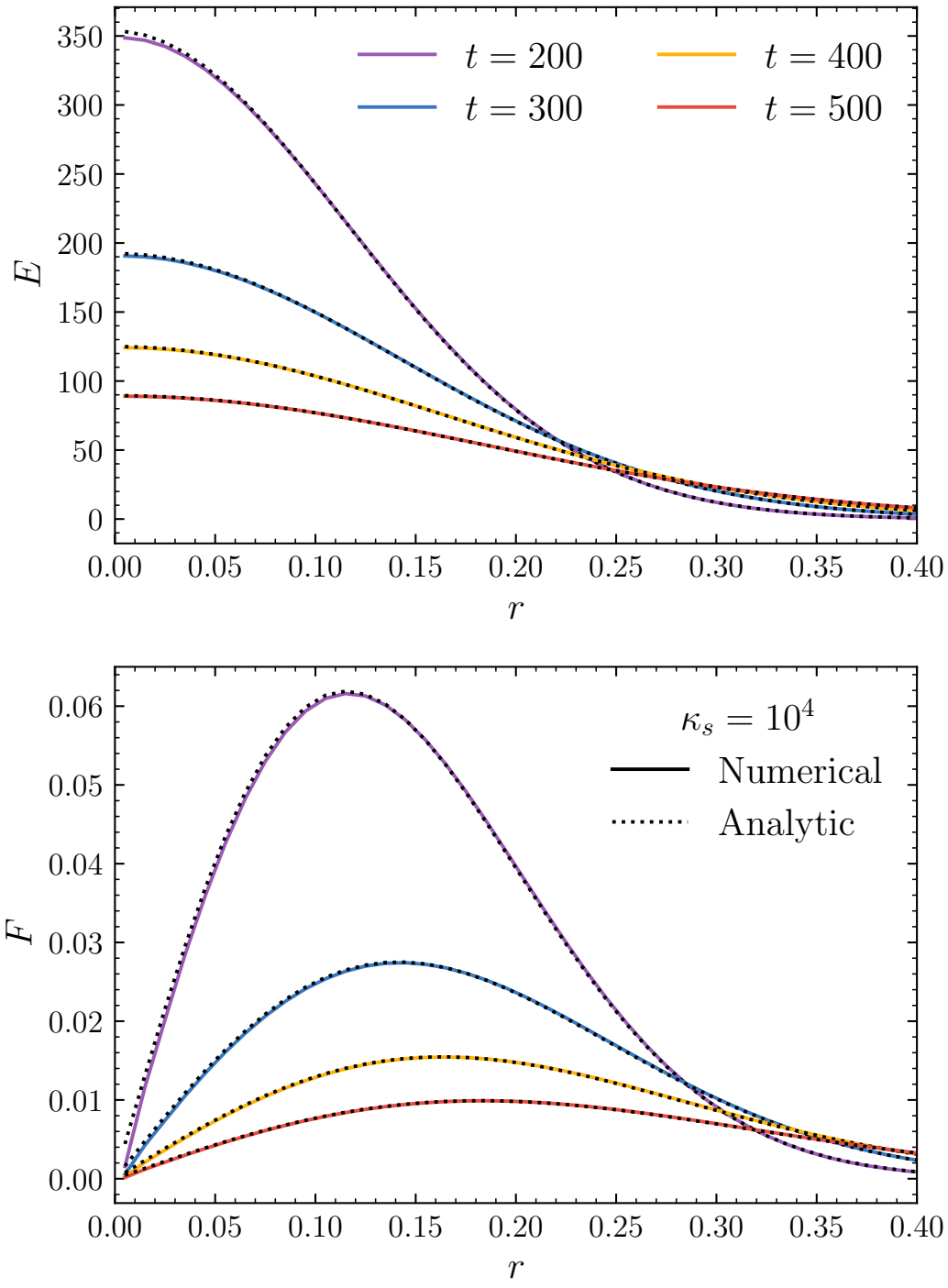

Figure 6.10: Results for the point-like source in a scattering medium with $\kappa = 10^4$. The numerical (solid lines) and analytic (dotted lines) solutions are compared at various times.

The results of each test are displayed in figs. 6.9–6.10. The numerical solution closely

follows the analytic solution in both cases, verifying the optically thick correction to the hyperbolic fluxes works properly. These tests are in agreement with the results of similar tests performed in Pons et al. (2000); O'Connor (2015); Cheong et al. (2023).

6.2.4.2 Moving Medium

A moving medium is used to test the retention of the non-linear interaction source terms in the implicit update. These terms are most impactful in the optically thick limit, and become highly non-linear when the fluid velocity is non-zero. Failure to retain the non-linear terms or use of $\mathcal{O}(v/c)$ approximations can lead to violation of energy conservation and incorrect advective speeds (Radia et al., 2022). This test will follow a similar setup as the one used in Radice et al. (2022); Cheong et al. (2023).

A radiation pulse will be set in a background scattering medium moving with a velocity v in a one-dimensional Minkowski spacetime. The initial pulse is place at the origin with the shape

$$E(x,0) = \exp\left(-9x^2\right). \tag{6.21}$$

Solving Eq. (6.17) with this initial condition and advecting the resulting solution at a constant velocity v yields the expected solution for the time-evolution of the energy density

$$E(x,t) = \frac{1}{\sqrt{1+36\tau_{\mathrm{d}}t}} \exp\left[-\frac{9(x-vt)^2}{1+36\tau_{\mathrm{d}}t}\right]. \tag{6.22}$$

The domain spans $-5 < x < 5$ and the x-axis is discretized into 1025 cells. The background medium is assigned a scattering opacity $\kappa = 10^3$ and a velocity along the x-axis of $v = 0.5$.

The results at a time $t = 4$ are displayed in fig. 6.11. The potential issues associated with

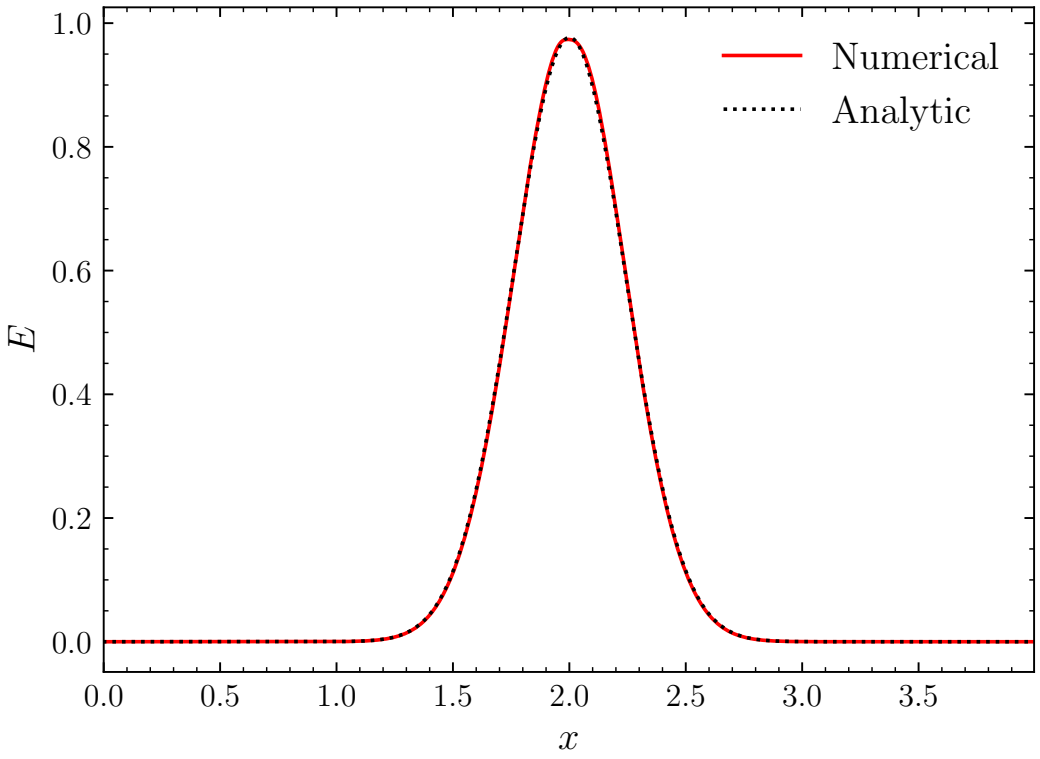

Figure 6.11: Results for a radiation pulse in a moving scattering medium. The numerical result (red solid line) closely matched the expected result (black dotted line). No common issues associated with incorrect treatment of the non-linear interactions terms are present in the solution.

an incorrect treatment of the non-linear source terms are not present in the solution, verifying their implementation in the `GRM1` solver works correctly. These results are in agreement with the tests performed with similar treatments of these terms in Radice et al. (2022); Cheong et al. (2023).

6.2.5 Advection Through Velocity Jump

For highly-relativistic problems, the `GRM1` solver will need to be able to handle large velocities and gradients. A beam of radiation is evolved through a velocity jump to test these limits. This test will be restricted to a completely optically thin background to isolate and verify the velocity-dependence in solving the closure in the co-moving frame.

For this test, a one-dimensional Minkowski spacetime spanning $-1 < x < 1$ will be discretized into 512 cells. Simple outflow conditions will be imposed at the boundaries. The background optically thin medium is assigned a velocity

$$v_x = \begin{cases} 0.9, & x < 0 \\ \\ -0.9, & x > 0 \end{cases}, \tag{6.23}$$

for a relative Lorentz factor $W \approx 10$ across the jump located at $x = 0$. The beam is initialized as $E = F^x = 1$ in the region $x < -0.5$ and zero elsewhere. Separate simulations will use the more-dissipative ($\theta = 1$) and less-dissipative ($\theta = 2$) limiters.

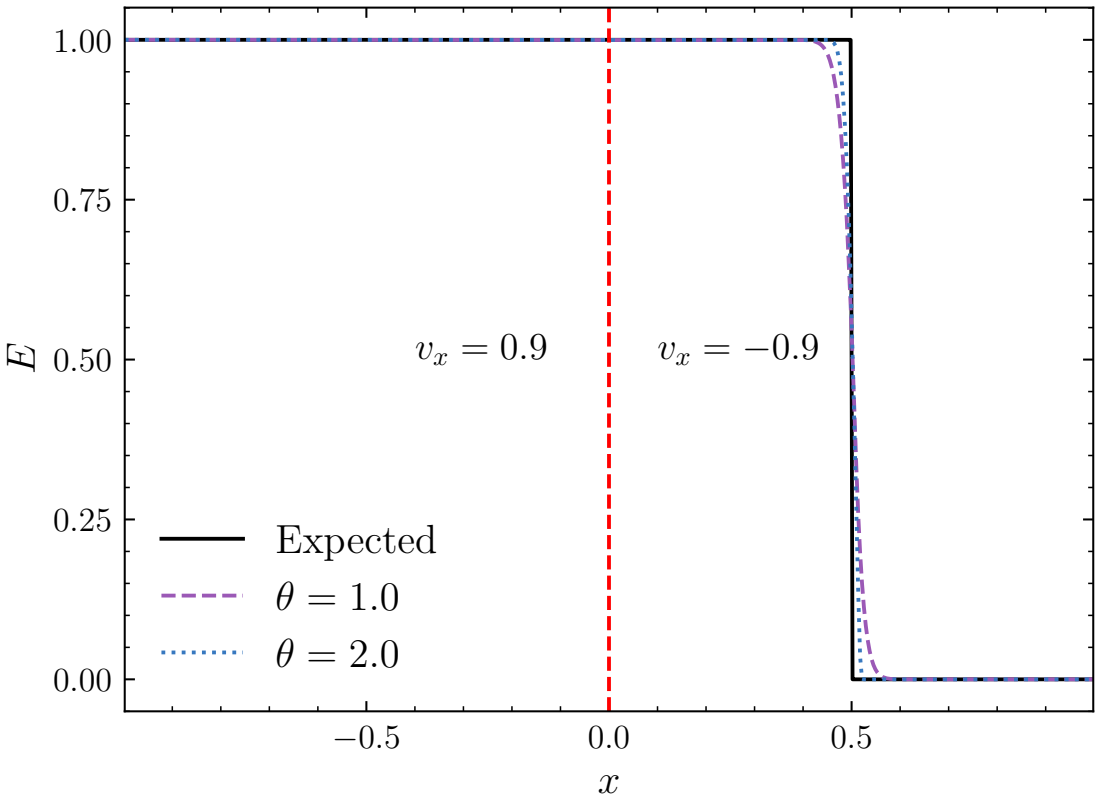

Figure 6.12: Propagation of a radiation beam through a relativistic velocity jump. Velocities are fixed at $v_x = 0.9$ for $x < 0$, and $v_x = -0.9$ for $x > 0$. The numerical solutions for the more-dissipative (purple dashed line) and less-dissipative (blue dotted line) limiters are compared with the expected solution (black solid line).

Both simulations demonstrate the `GRM1` solver's ability to advect the radiation beam through the velocity jump; see fig. 6.12 for the results. There are no spurious oscillations observed at the velocity jump, and the beam maintains its expected speed. As with the previous beam tests, the less-dissipative limiter better resolves the leading edge of the beam. These results compare favorably with the similar tests performed in Radice et al. (2022); Cheong et al. (2023).

6.2.6 Velocity- and Curvature-Dependent Redshifting

The frequency advection terms couple the evolution equations for each frequency when the fluid is accelerating or the spacetime is not flat. To demonstrate the effectiveness of the pseudospectral discretization of the frequency advection operator, the co-moving frame luminosity of an optically thick radiating sphere will be measured. Three cases will be considered: a finite-velocity profile located far outside a zero-mass sphere, a massive sphere and zero-velocity everywhere, and the mixed case of the massive sphere and finite-velocity profile. These tests will utilize setups similar to those described in Müller et al. (2010); O'Connor (2015); Cheong et al. (2023).

In a spherically symmetric spacetime, the luminosity measured by a co-moving observer is simply the total energy flux passing through a spherical surface at a radius r. In the optically thin limit, this takes the form

$$\mathcal{L}(r) = 4\pi \int_0^\infty \mathrm{d}\nu \, H^r_{(\nu)} = \frac{W}{\alpha} \frac{1 - v^r}{1 + v^r} \left[\alpha \int_0^\infty \mathrm{d}\nu \, F^r_{(\nu)} \right]. \tag{6.24}$$

The steady-state limit of the frequency-integrated evolution equation for the energy density,

Eq. (3.44), in the optically thin limit shows that the final bracketed quantity in Eq. (6.24) is

$$\alpha \int_0^\infty \mathrm{d}\nu \, F^r_{(\nu)} = \text{const.} \tag{6.25}$$

everywhere outside of the sphere. The term in the brackets can thus be evaluated at a point outside of the sphere, and scaled by $W\alpha^{-1}(1-v^r)/(1+v^r)$ to determine the expected co-moving frame luminosity.

For all tests, a radiating sphere of radius $R = 8$ is placed at the origin and assigned an absorption opacity $\kappa_{\text{abs.}} = 100$. The frequencies are discretized around a $T = 5$ MeV equilibrium Fermi-Dirac distribution using five frequency collocation points; the equilibrium energy density and emissivity for each frequency are also set from this distribution. A one-dimensional spherically symmetric domain spans $0 \leq r \leq 800$ and is discretized into 4096 radial cells. A reflective boundary is used at the origin, while an outflow condition is imposed at the outer boundary. For cases with a non-zero velocity, the velocity profile is set to

$$v^r = \begin{cases} 0, & r < 90 \\ -0.2\left(\dfrac{r-90}{10}\right), & 90 \leq r \leq 100 \\ -0.2\left(\dfrac{100}{r}\right)^2, & r > 100 \end{cases} . \tag{6.26}$$

For cases with a sphere of mass $M = 1.8$, the constant-density solution of the Tolman-Oppenheimer-Volkoff (TOV) equations is used (see Wald (1984) for the derivation). In a TOV spacetime, a self-gravitating sphere of constant density has an enclosed mass at an interior point $m(r) = M(r/R)^3$. This leads to the spacetime quantities that differ from a

spherical Minkowski spacetime

$$\alpha = \begin{cases} \frac{3}{2}\sqrt{1-\frac{2M}{R}} - \frac{1}{2}\sqrt{1-\frac{2Mr^2}{R^3}}, & r \leq R \\ \sqrt{1-\frac{2M}{r}}, & r > r \end{cases}, \tag{6.27}$$

$$\gamma_{rr} = \begin{cases} \left(1-\frac{2Mr^2}{R^3}\right)^{-1}, & r \leq R \\ \left(1-\frac{2M}{r}\right)^{-1}, & r > R \end{cases}. \tag{6.28}$$

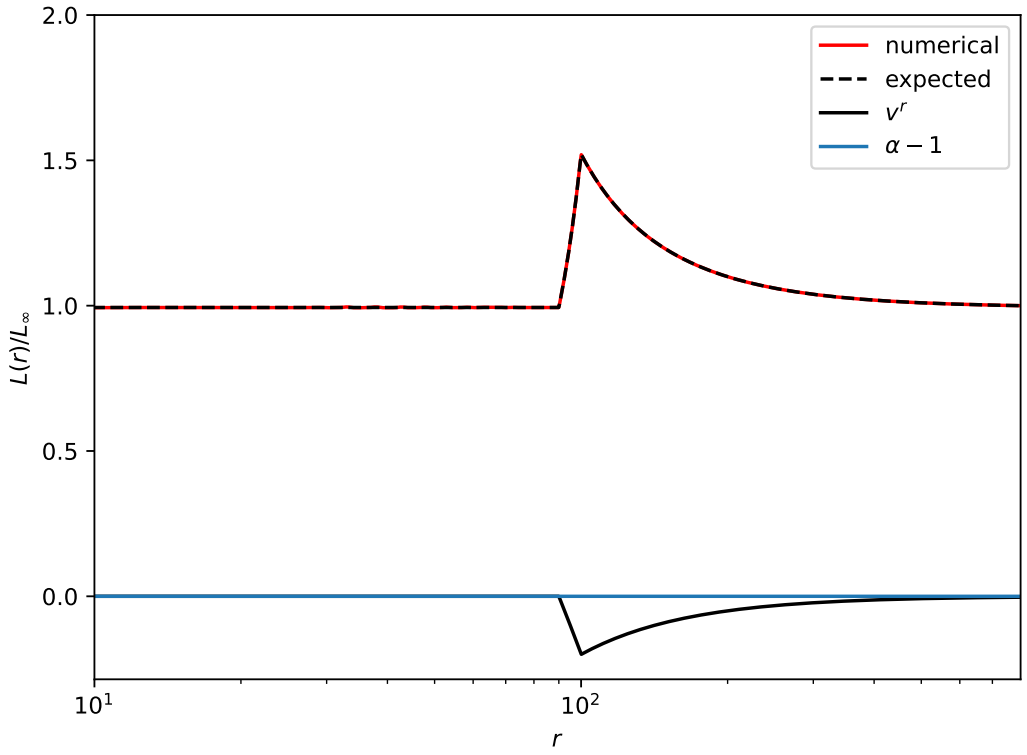

Figure 6.13: Luminosity measured in the co-moving frame for the finite-velocity case. The numerical results (red solid line) are compared to the expected results (black dashed line). The velocity profile (black solid line) and lapse (blue solid line) are shown for reference.

Each simulation was evolved until a time $t = 1000$. An initial set of tests using the finite-volume discretization successfully performed in the finite-velocity and finite-mass only cases, but struggled with the mixed case; oscillations formed near the discontinuities in the

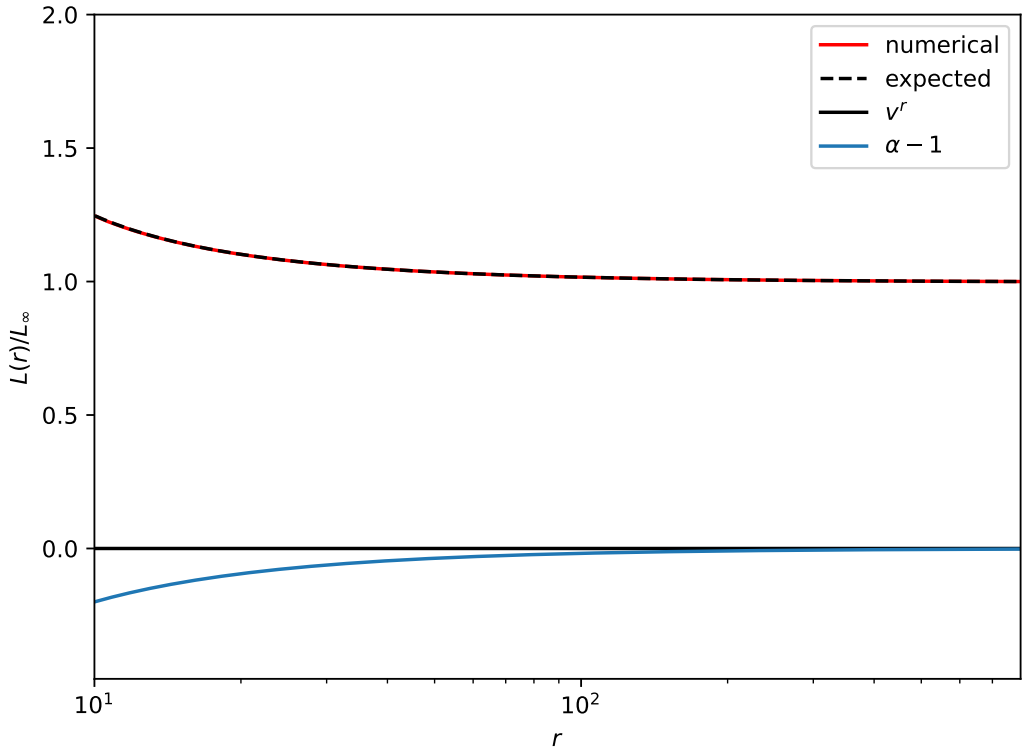

Figure 6.14: Luminosity measured in the co-moving frame for the finite-mass case. The numerical results (red solid line) are compared to the expected results (black dashed line). The velocity profile (black solid line) and lapse (blue solid line) are shown for reference.

velocity's gradient, but remained as a steady-state solution. This prompted the addition of the finite-difference discretization, which was able to successfully evolve all three cases to their expected steady-state solution. These results are displayed in figs. 6.13–6.15. In each case, the numerical solution closely follows the expected results. Some slight noise is present in the region preceding the velocity profile in the finite-velocity case, but this is not unexpected with the fairly coarse discretization used in these tests.

Overall, these results constitute a successful test of the redshifting discretization. The results obtained here with only five neutrino frequencies compare favorably with the similar tests performed in O'Connor (2015); Cheong et al. (2023) using a frequency-bin method utilizing 18 neutrino frequencies. Further testing of this discretization is necessary to determine

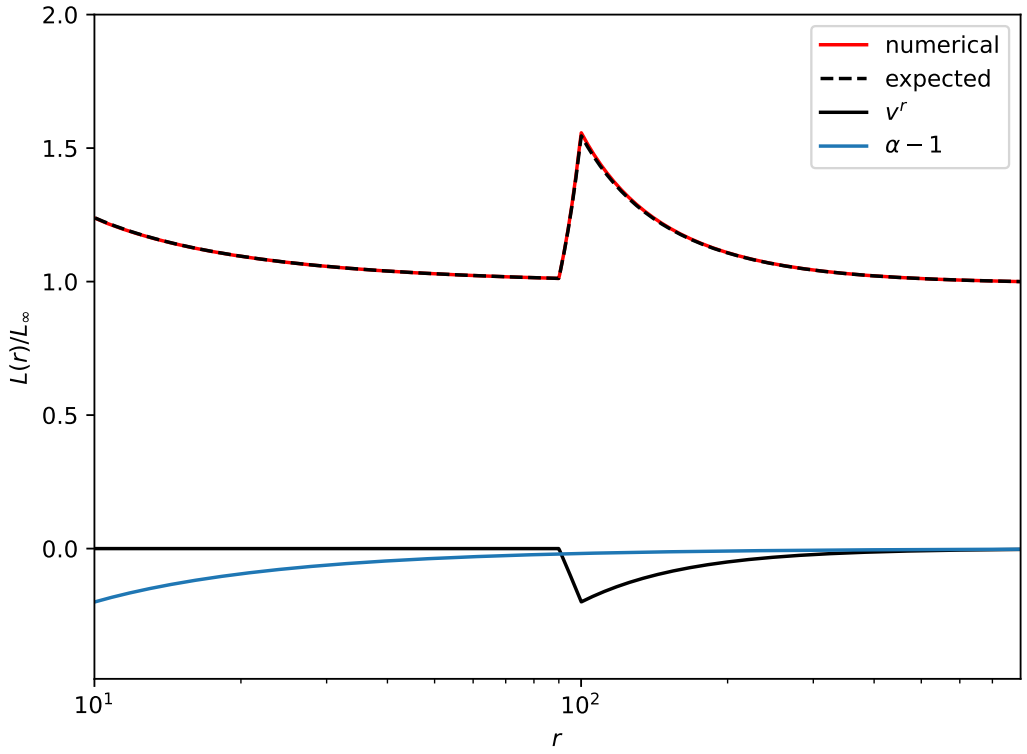

Figure 6.15: Luminosity measured in the co-moving frame for the combined finite-velocity and finite-mass case. The numerical results (red solid line) are compared to the expected results (black dashed line). The velocity profile (black solid line) and lapse (blue solid line) are shown for reference.

how effective and accurate it is when the underlying distribution of the neutrinos deviates far from the equilibrium distribution it has been discretized around.

6.2.7 Spatial Convergence

To verify the expected second-order spatial convergence of the `GRM1` solver, the optically thin and thick cases are tested separately for problems that have analytic solutions $E(x,t)$ for the energy density. These will be used to quantify the numerical error in solution at the N-th time step in the i-th cell by the residual $\delta E_i^N = E_i^N - E(x_i, t_N)$. From these residuals, the

discrete L_2-norm can be computed over the grid (LeVeque, 2002)

$$\left\|\delta E^N\right\|_2 = \sqrt{\Delta x \sum_i \left(E_i^N - E(x_i, t_N)\right)^2}. \tag{6.29}$$

The spatial-convergence rate can easily be obtained from the slope of the best-fit line (in log-log space) of the residual as a function of the grid resolution.

6.2.7.1 Optically Thin Limit

For the optically thin limit, a radiation pulse is evolved in an optically thin Bondi flow outside of a Schwarzschild black hole. The analytic for the outgoing wavelike solution from Shibata et al. (2011) will be used to compare the numerical results against. In Kerr-Schild spherical coordinates, the outgoing solution takes the form

$$E(r,t) = F^r(r,t) = \frac{g_+(r^* - t)}{2\sqrt{\gamma}\alpha^3(1-2M/r)^2}, \tag{6.30}$$

where M is the mass of the black hole, and $g_+(r^* - t)$ is the shape of the radiation pulse at a time t with r^* defined as

$$r^* = \int \mathrm{d}r \, \frac{r+2M}{r-2M} = r + 4M \log\left[\frac{r-2M}{M}\right]. \tag{6.31}$$

A similar setup to the numerical simulations in Shibata et al. (2011) is used, and the shape of the pulse is set to

$$g_+(r^* - t) = \exp\left[-\frac{\left(r^* - t - r_0^*\right)^2}{8M^2}\right]. \tag{6.32}$$

The radial axis of a spherically symmetric domain is discretized into 2^k cells, for $k \in [9, 13]$. The simulations are evolved until a final time $t_N = 35$. The shape of the pulse is extracted from the numerical results as

$$E^+ = 2\sqrt{\gamma}\alpha^3\left(1 - \frac{2M}{r}\right)^2 E. \tag{6.33}$$

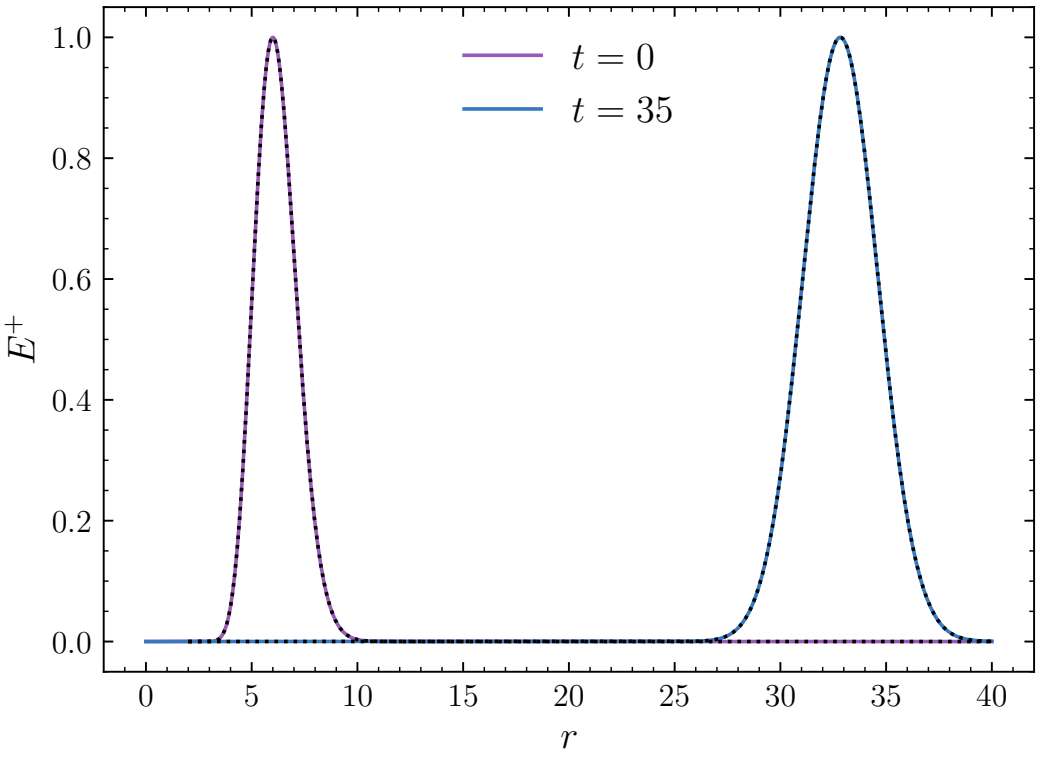

Figure 6.16: Initial and final state for a radiation pulse in an optically thin Bondi flow from the highest resolution ($k = 13$) case. The extracted shape of the wave pulse is extracted and compared to the analytic result.

The results for the shape of the pulse at the initial and final times for the highest resolution case $k = 13$ are shown in fig. 6.16, while fig. 6.17 displays the L_2-norm of the residuals across all resolutions. The spatial convergence rate for these results is 2.08 ± 0.03, demonstrating the expected second-order accuracy of the spatial discretization in the optically thin limit.

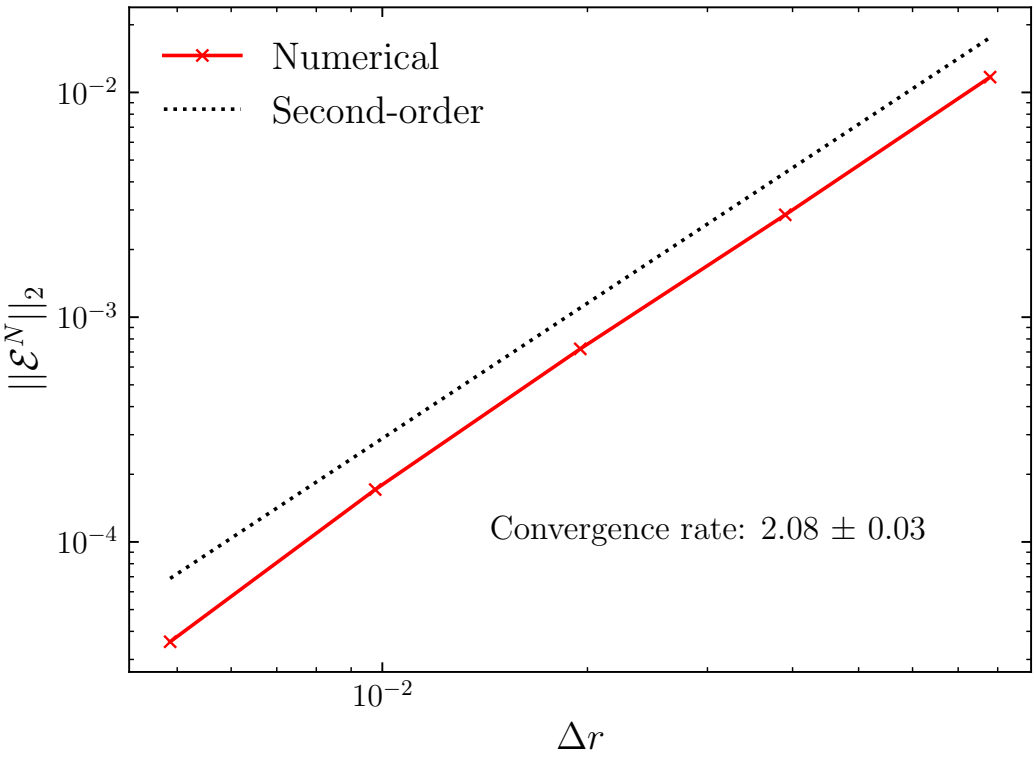

Figure 6.17: Spatial convergence rate for a radiation pulse in an optically thin Bondi flow. The numerical results (red line) meet the expected second-order convergence rate (black dashed line for reference).

6.2.7.2 Optically Thick Limit

The spatial convergence rate in the optically thick limit is tested by evolving a radiation pulse in a purely scattering stationary medium. For this test, the initial pulse is set to a unit box centered at the origin

$$E(x, t = 0) = \begin{cases} 1, & -\frac{1}{2} < x < \frac{1}{2} \\ 0, & \text{otherwise} \end{cases} . \tag{6.34}$$

Solving Eq. (6.17) with this initial condition gives the analytic result

$$E(x,t) = \frac{1}{2}\left[\mathrm{erf}\left(\frac{x+\frac{1}{2}}{\sqrt{4\tau_{\mathrm{d}}t}}\right) - \mathrm{erf}\left(\frac{x-\frac{1}{2}}{\sqrt{4\tau_{\mathrm{d}}t}}\right)\right]. \tag{6.35}$$

The background scattering opacity is set to $\kappa_{\mathrm{iso}} = 10^3$ everywhere in the domain. The domain spans $-2 < x < 2$ and is discretized into 2^k cells, where $k \in [7, 11]$ for separate simulations. Each simulation is evolved until a time $t = 10$.

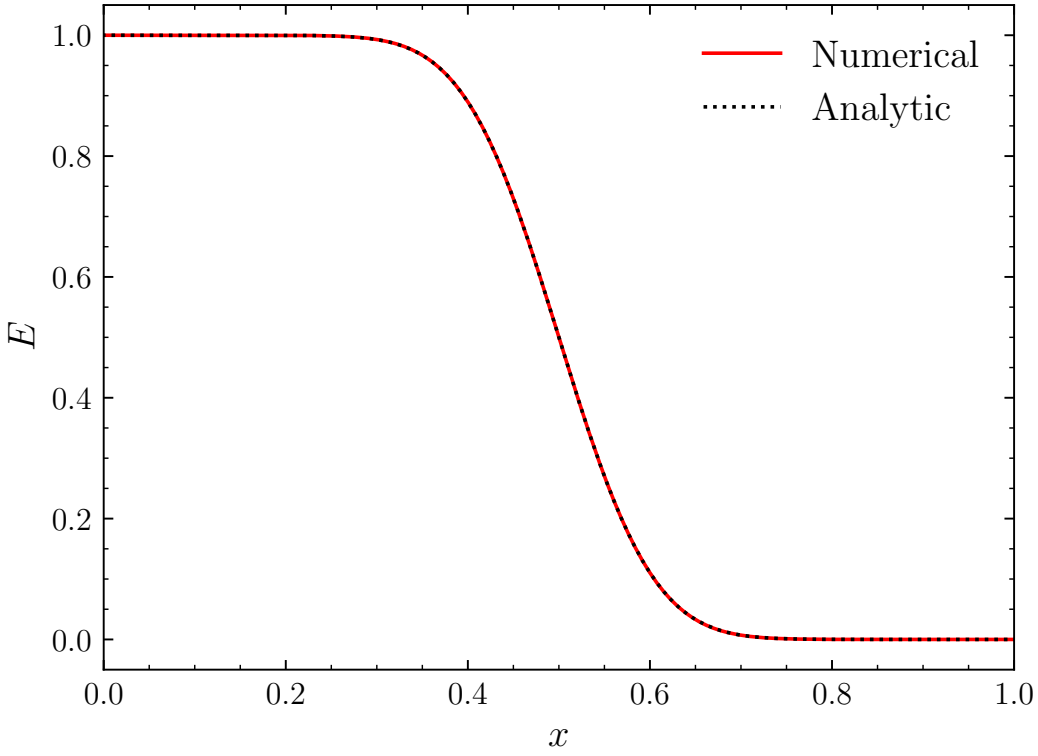

Figure 6.18: Final state for a radiation pulse in an optically thick scattering medium from the $k = 10$ case. The numerical result (red solid line) follows the analytic solution (black dotted line) closely.

The results for the $k = 10$ resolution are displayed in fig. 6.18 and show good agreement with the analytic solution. The convergence rate found across all resolutions was found to be 1.77 ± 0.02, almost reaching the expected second-order rate; see fig. 6.19 for the results. These results are acceptable for the discontinuous initial data; the combination of the HLLE

Riemann solver and the generalized minmod limiter are only second-order accurate in smooth regions of the solution.

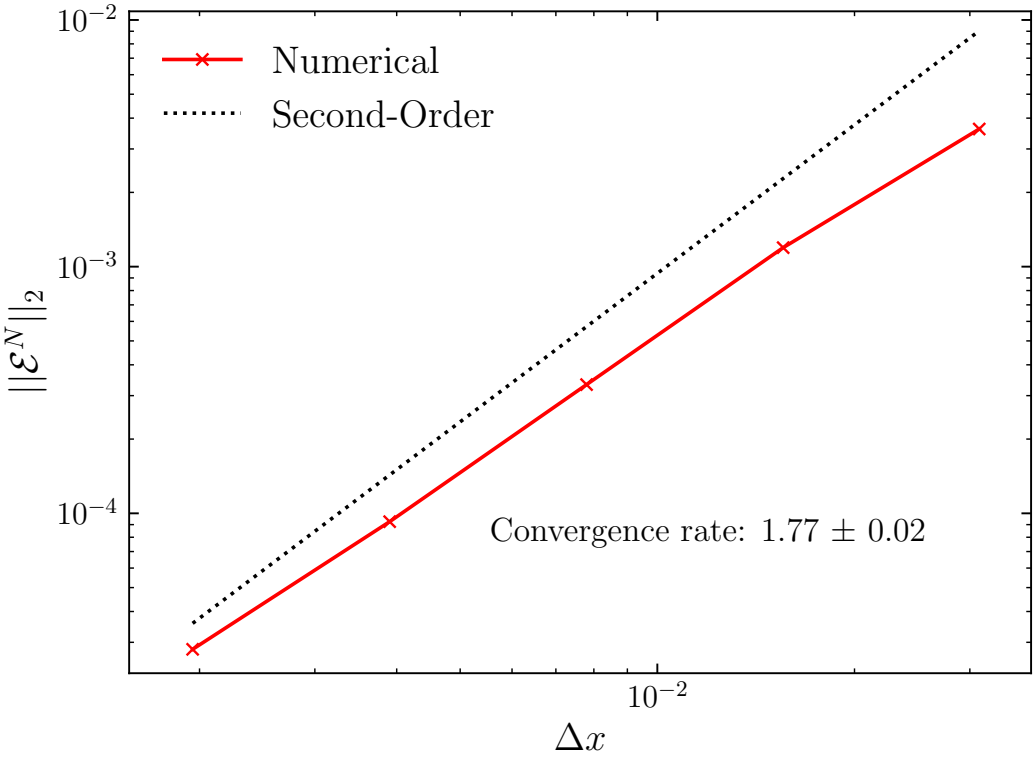

Figure 6.19: Spatial convergence rate for a radiation pulse in an optically thick scattering medium. The numerical results (red line) approach the expected second-order convergence rate (black dashed line for reference).

6.3 GRFD

The `GRFD` solver provides a general relativistic implementation of the `Flash-X` hydrodynamics physics module using the formalism and numerical methods presented in chapter 4. Similarly to the `GRM1` solver, this solver implements the MoL-specific pieces of the module's public interface, including calculation of the explicit RHS terms and pre- and post-update synchronization. This solver also encapsulates the calculations of the terms in the evolution equations in compact kernels, which are then applied to individual cells in grid block-

or row-levels procedures. These procedures are organized and managed directly within the public interface implementations that serve as the entry point and runtime controls for the solver.

The goal of the `GRFD` solver is two-fold: provide an example solver that utilizes and tests the MoL ERK integrator, and provide general relativistic hydrodynamics to be used alongside the `GRM1` solver. The latter goal is presently being worked towards. The remainder of this section will present a series of test problems that serve to verify the solver's implementation and its use of the ERK integrator. All test problems will use the fourth-order ERK-RK4 time-integrator (see appendix G.1). Time-steps will again be limited by a CFL number as in Eq. (6.13). For one-dimensional simulations will use $C_{\text{cfl}} = 0.5$ and two-dimensional simulations will use $C_{\text{cfl}} = 0.2$. All simulations will use the symmetric fifth-order WENO reconstruction scheme described in chapter 4, and utilize a layer of four guard-cells.

6.3.1 Relativistic Shock Tube

One of the standard test problems for hydrodynamics solvers is a shock tube setup. These types of problems set discontinuous left and right states for the fluid's density, pressure, and velocity. Depending on these initial states and their relative velocity, the system will develop into one of three patterns: two shocks, two rarefaction waves, or a shock and rarefaction wave. These are ideal tests to verify a solver's ability to capture the shocks in the fluid.

In relativistic hydrodynamics, the solutions for shock tube problems couple velocities in all directions via the Lorentz factor, and prove more challenging than non-relativistic solutions where only the velocity in the direction of the flow matters (Rezzolla et al., 2003). This test problem will use a relativistic blast-wave setup, and separately consider cases with and without an initial transverse velocity ahead of the shock to verify that the `GRM1` solver

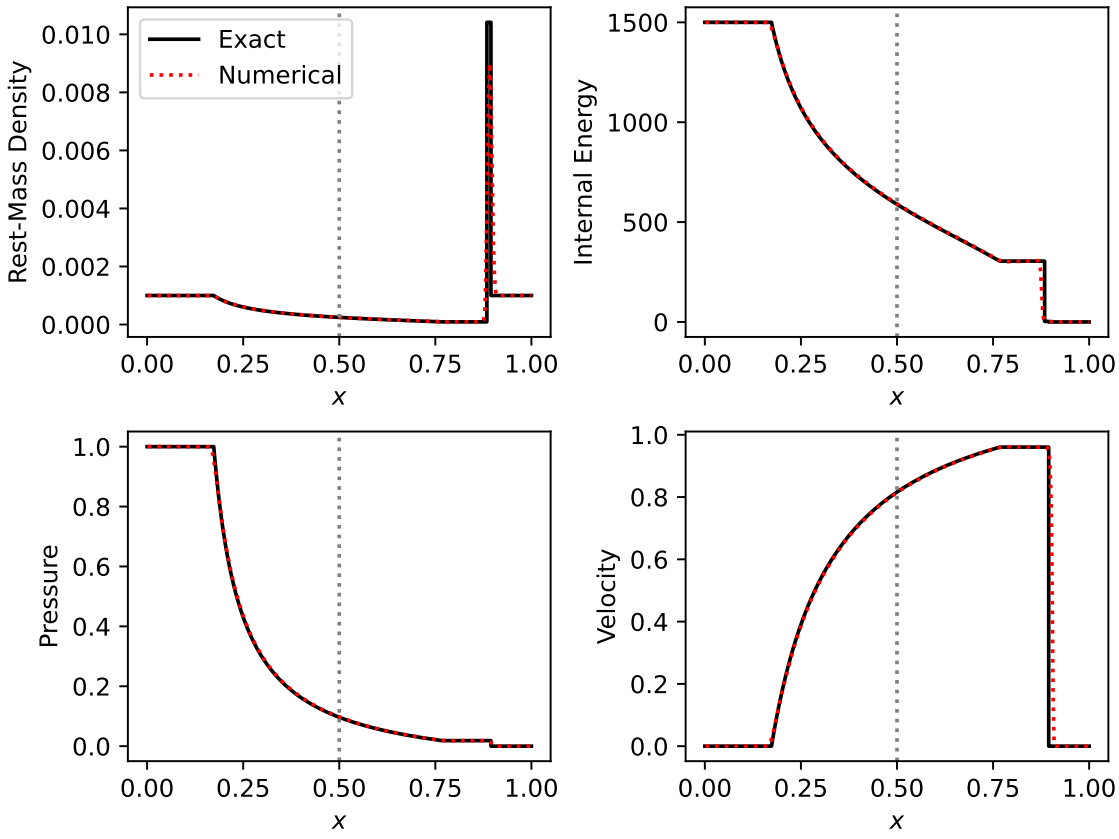

Figure 6.20: Results of the shock tube problem for a relativistic blast-wave and no transverse velocity. The initial discontinuity is located at $x = 0.5$. The numerical results (red dotted lines) capture the shock and closely follow the exact solution (black solid line).

can not only capture the shock, but also correctly include the fully velocity-dependence in the solution. The initial left (L) and right (R) states will be set to

$$\rho^{\mathrm{L}} = 10^{-3}, \qquad p^{\mathrm{L}} = 1, \qquad v^{\mathrm{L}} = 0, \tag{6.36}$$

$$\rho^{\mathrm{R}} = 10^{-3}, \qquad p^{\mathrm{R}} = 10^{-5}, \qquad v^{\mathrm{R}} = 0, \tag{6.37}$$

and an initial transverse velocity (relative to the direction the shock travels) is separately set to $v_{\perp}^{\mathrm{R}} = 0$ and $v_{\perp}^{\mathrm{R}} = 0.99$ ahead of the shock in each simulation, and zero behind the shock for both. The initial discontinuity is placed at $x = 0.5$. An ideal Γ-law equation of state is used with $\Gamma = 5/3$. The domain spans $0 < x < 1$ and is discretized into 400 cells. Both

simulations are then evolved until a time $t = 0.4$, allowing time for the shock and rarefaction waves to clearly develop while still staying within the computational domain.

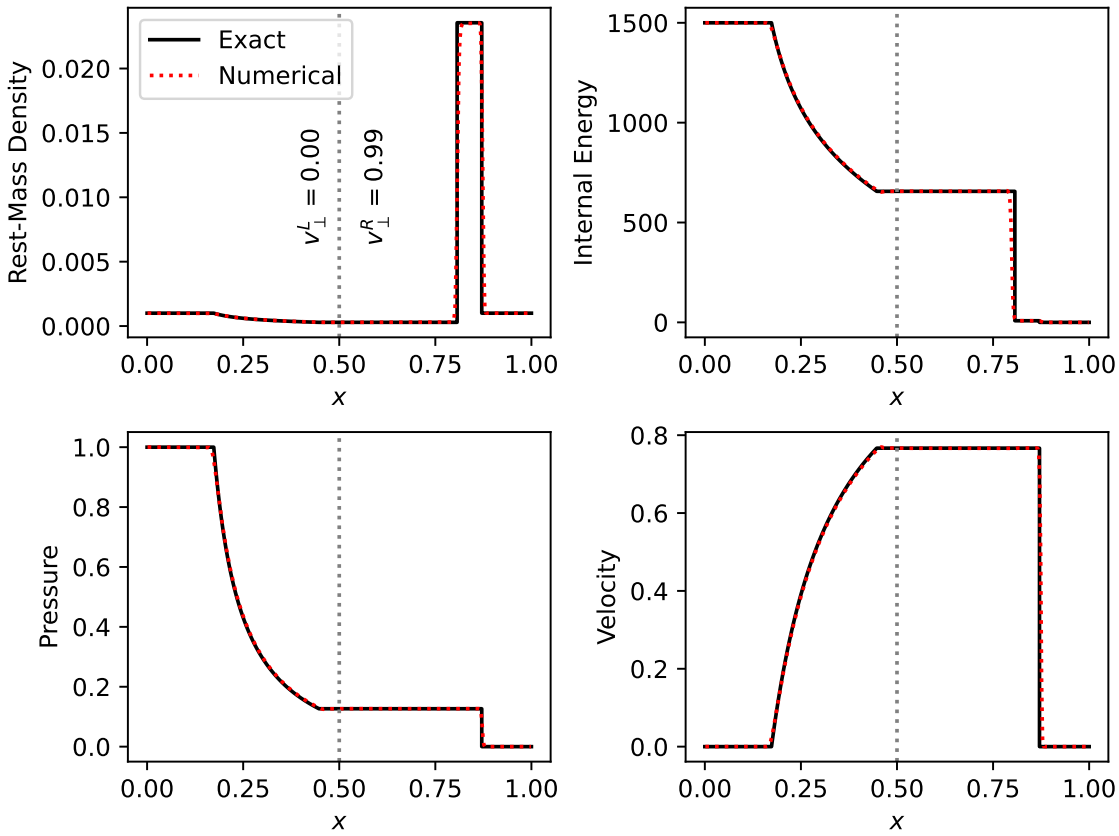

Figure 6.21: Results of the shock tube problem for a relativistic blast-wave with an initial transverse velocity $v_\perp = 0.99$ set in the region ahead of the shock. The initial discontinuity is located at $x = 0.5$. The numerical results (red dotted lines) capture the shock and closely follow the exact solution (black solid line).

The results for these shock tube problem are displayed in figs. 6.20–6.21. The exact solutions are calculated following the procedure described in Rezzolla et al. (2003); Rezzolla & Zanotti (2013). The rest-mass density, pressure, internal energy, and fluid velocity all follow the exact results closely. The solver resolves the leading shock and locations of the contact discontinuity and the tail/head of the rarefaction wave without any spurious oscillations or severe broadening in both cases, although the narrow region between the contact discontinuity and the shock in the case without transverse velocity requires a higher grid resolution to

better capture the magnitude of the density. These results are all in agreement with similar tests presented in Radice & Rezzolla (2012); Rezzolla & Zanotti (2013).

6.3.2 Kelvin-Helmholtz Instability

The next test verifies the `GRFD` solver's multi-dimensional capabilities by examining the development of a Kelvin-Helmholtz instability in the presence of relativistic fluid velocities. These instabilities form vortices along the contact between fluids of different densities moving in different directions. The simulation used here will follow the one presented in Radice & Rezzolla (2012). A high- and low-density regions are placed in contact with each other and assigned opposing velocities, with a small initial perturbation in the transverse directions of the flow to prompt the formation of the instability. The two-dimensional setup for this problem separates these regions along the y-axis, and sets the initial flow directions along the x-axis, and uses the initial values for the velocity and density

$$v^x(y) = \begin{cases} V_0 \tanh\left(\dfrac{y-\frac{1}{2}}{d}\right), & y > 0 \\ -V_0 \tanh\left(\dfrac{y+\frac{1}{2}}{d}\right), & y \leq 0 \end{cases}, \tag{6.38}$$

$$v^y(x,y) = \begin{cases} A_0 V_0 \sin(2\pi x) \exp\left[-\dfrac{\left(y-\frac{1}{2}\right)^2}{\sigma}\right], & y > 0 \\ -A_0 V_0 \sin(2\pi x) \exp\left[-\dfrac{\left(y+\frac{1}{2}\right)^2}{\sigma}\right], & y \leq 0 \end{cases}, \tag{6.39}$$

and

$$\rho(y) = \begin{cases} \rho_0 + \rho_1 \tanh\left(\dfrac{y - \frac{1}{2}}{d}\right), & y > 0 \\ \rho_0 - \rho_1 \tanh\left(\dfrac{y + \frac{1}{2}}{d}\right), & y \leq 0 \end{cases}, \tag{6.40}$$

where V_0 is the magnitude of the initial opposing velocities, d is the distance over which the velocities change, A_0 and σ describe the amplitude and width of the initial perturbation, respectively, and ρ_0 and ρ_1 are used to set the initial densities. The parameters are set to the values

$$\begin{aligned} V_0 &= 0.5, \qquad & d &= 0.01 \\ A_0 &= 0.1, \qquad & \sigma &= 0.1 \\ \rho_0 &= 0.505, \quad & \rho_1 &= 0.495 \end{aligned} \tag{6.41}$$

For this simulation, the domain spans $-0.5 < x < 0.5$ and $-1 < y < 1$, and is discretized into 256×512 grid cells. Periodic boundary conditions are used on all sides of the domain. An ideal Γ-law equation of state is used with $\Gamma = 4/3$. The simulation is ran until a time $t = 3$. The results shown in fig. 6.22 display the development of the primary larger vortices expected for this type of instability. The smaller secondary smaller vortices represent numerical artifacts that develop differently for varying resolutions and different numerical treatments of the fluxes; these are commonly seen in simulations of Kelvin-Helmholtz instabilities (Radice & Rezzolla, 2012). The numerical results mirror each other across the line $y = 0$, maintaining the symmetry expected from the initial fluid state and periodic boundaries conditions. Overall, the `GRFD` solver handles the development of these Kelvin-Helmholtz instabilities very well, verifying its multi-dimensional capabilities in handling relativistic velocities and developing dynamic features such as the vortices seen in Kelvin-Helmholtz instabilities.

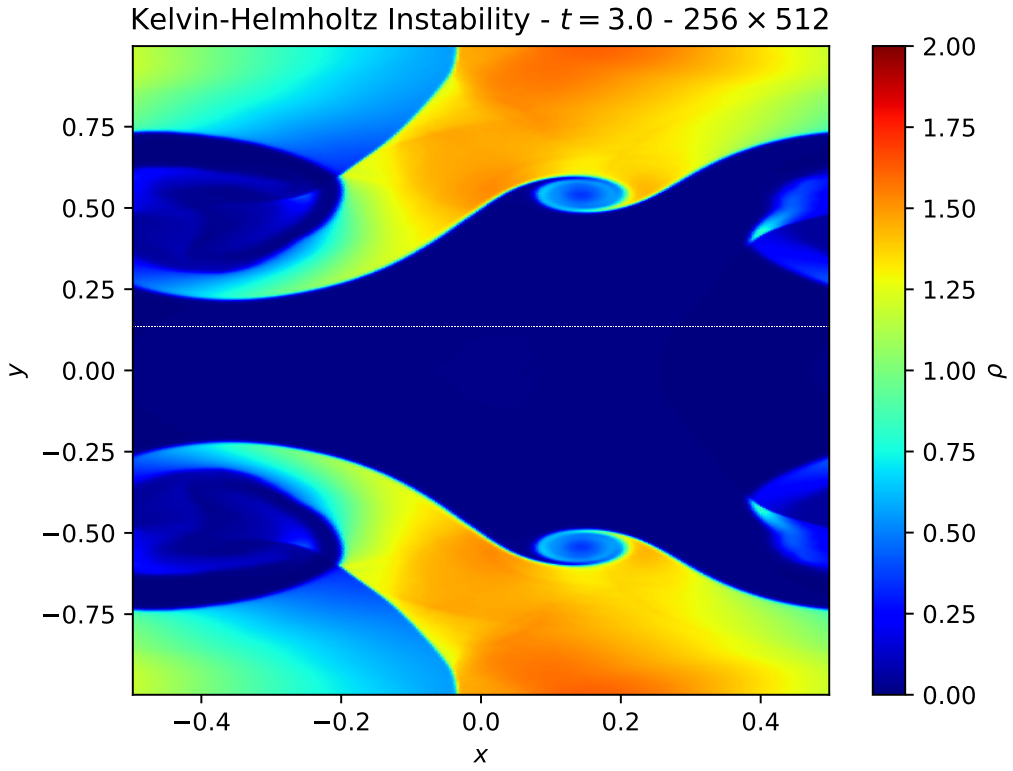

Figure 6.22: Results for the Kelvin-Helmholtz instability problem. The higher-density upper- and lower- regions and middle-regions are given opposing velocities. The numerical results for the rest-mass density show the development of the primary vortices, as well as the smaller secondary vortices that can be attributed to the resolution and numerical methods.

6.3.3 Bondi-Hoyle Accretion

The Bondi-Hoyle problem describes the the accretion of an infinite gas cloud onto a moving star (Edgar, 2004). This problem provides an excellent test for the general relativistic capabilities of the `GRFD` solver when applied to accretion onto a black hole. The alternative case of a constant wind accreting onto a stationary black hole will be considered, such that a constant analytic background spacetime can be used.

This simulation will use an infinite wind with an initial density $\rho_0 = 1$ and velocity $v_0 = 0.3$ in the positive x-direction; the sound speed is chosen to be $c_{s,0}$ such that the wind

will be supersonic. An ideal Γ-law equation of state with $\Gamma = 4/3$ will be used to invert the density and sound-speed to find the initial pressure

$$p_0 = \frac{\rho_0 c_{s,0}^2 (\Gamma - 1)}{\Gamma\Big(\Gamma - 1 - c_{s,0}^2\Big)}, \tag{6.42}$$

from which the remaining fluid variables can then be found. A unit mass Schwarzschild black hole is placed at the origin. The two-dimensional spherical domain in the azimuthal plane will be used, and described by spherical Kerr-Schild coordinates. The metric quantities will set the same as in Eq. (6.14), but with the replacement $\ell^i = (1, 0, 0)$ and $\eta_{ij} = \text{diag}\Big(1, r^2 \sin^\theta, r^2\Big)$. Note that the order of the coordinate bases is (r, ϕ, θ) instead of the typical (r, θ, ϕ) to ensure that these two directions are properly treated as the first- and second-dimensions during and direction-dependent calculations in the two-dimensional simulation. In this coordinate system, the initial velocity components are then set as

$$\begin{aligned} v^r(r, \phi) &= v_0 \sqrt{\gamma^{rr}} \cos\phi, \\ v^\phi(r, \phi) &= -v_0 \sqrt{\gamma^{\phi\phi}} \sin\phi, \end{aligned} \tag{6.43}$$

such that the velocity points in the positive x-direction and $\gamma_{ij} v^i v^j = v_0^2$; these velocity components will always take this form for any spherically symmetric spacetime even when using different coordinate bases so long as the metric is strictly diagonal.

The domain covers the full 2π azimuthal angle and extending out to a radius of $r_{\text{max}} = 100$. The minimum radius is instead chosen as $r_{\text{min}} = 2.2$ such that the inner boundary is located just outside the event horizon; this allows the simulation to disregard the acausally-disconnected region inside the horizon. Outflow conditions will be used at the inner boundary by extrapolating the inner-most cell's values towards the event horizon; this technique has

been found to work well in similar studies of Bondi-Hoyle accretion performed in Font & Ibáñez (1998); Font et al. (1999); Zanotti et al. (2011). The initial wind conditions will be set directly at the outer boundary in the upstream direction $\pi/2 < \phi < 3\pi/2$, while outflow conditions will be used in the downstream region. The domain will use an initial domain discretized into 256 radial cells and 128 azimuthal cells. Unlike the uniform grid used in previous simulations, this simulation will make use of the adaptive mesh refinement (AMR) capabilities in `Flash-X` via the `Paramesh` AMR library (MacNeice et al., 2000). The simulation uses three levels of refinement and the `Flash-X` default second-order monotonic interpolation for refinement operations[1] At each refinement level, the domain is broken up into blocks of size 16×16 cells, with a layer of four guard cells on each side.

The simulation was ran until a time $t = 1500$, and the Cartesian projection of the results are displayed in figure fig. 6.23. This allowed for sufficient time for the expected shock cone downstream from the black hole to form. The three refinement levels were sufficient to resolve the development of the shock cone and maintain numerical stability without a significantly more restrictive time-step that would be necessary if using a uniform grid of similar base resolution. These results compare favorably with the similar simulations performed in Font & Ibáñez (1998); Zanotti et al. (2011); Blakely & Nikiforakis (2015); Donmez (2021), and serve to verify the multi-dimensional general relativistic capabilities of the `GRFD` solver. These results also demonstrate the compatibility with the AMR capabilities in `Flash-X`, even in the presence of a black hole. Further testing of these AMR capabilities is necessary to determine the requirements for interpolation and any special considerations that must be made for dealing with black hole spacetimes.

[1]Second-order interpolation is not ideal for the fourth-order methods used through the `GRFD` solver, but was chosen for testing purposes as the less-often used higher-order interpolators available in `Flash-X` currently require modifications of modifications of the `Grid` module and its interface to `Paramesh` to access.

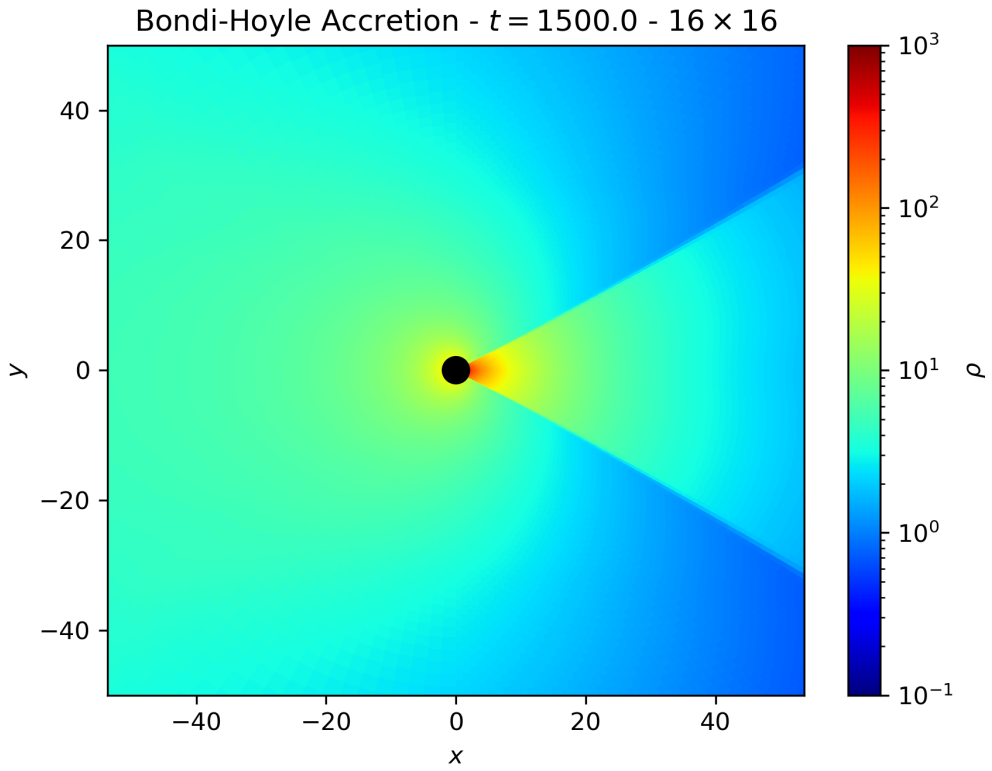

Figure 6.23: Results for Bondi-Hoyle accretion problem. A constant mach-3 wind enters the domain from the left and accretes onto a Schwarzschild black hole (black region at the origin). A strong shock cone forms downstream from the black hole.

6.4 `Z4c` and `CCZ4`

The new `Flash-X Spacetime` physics module was created to manage the metric quantities and support dynamic spacetime evolution. This new module provides the grid variables for the lapse α, shift vector β^i, spatial metric γ_{ij}, the extrinsic curvature K_{ij}, and the three projections of the stress-energy tensor $\mathcal{E}$, $\mathcal{S}_i$, and $\mathcal{S}_{ij}$. The grid variables for the metric quantities are required by the other general relativistic solvers, and are held static by default if no specific implementation for a spacetime solver is requested for a simulation. The grid variables for the projections of the stress-energy provide a common location for all physics

solvers to add their contributions. The Eulerian projections were chosen to represent the stress-energy tensor since these quantities easily calculated by the physics solvers and will be directly used by spacetime solvers. The `Spacetime` module also provides the tensor-indexing scheme used by the code-generator, i.e. the named integers for the x, y, z-components for tensors of rank-one or higher for use as array indices, including the reduction of symmetric indices to a single dimension of an array.

Both the Z4c and CCZ4 formulations described in chapter 5 are provided as implementations of the new `Spacetime` module. For both, the code-generator is used to create the kernel subroutines responsible for calculating the right-hand side terms of the evolution equations, converting the evolved variables to and from the 3+1 split metric quantities, and calculating the violations of the constraint equations as monitors of the stability of the evolving spacetime. These kernels are applied cell-by-cell across blocks of data in the grid. Both implementations make use of the fourth-order finite-difference operators described in chapter 5.

This section will present a series of initial test problems that demonstrate the correctness of the code-generator and the accuracy of the solvers. Beyond these initial problems, further tests of more dynamic spacetimes will require increased compatibility between the spacetime solvers and the adaptive mesh refinement capabilities in `Flash-X`. The currently available refinement criteria and interpolation schemes have proven inadequate for minimizing the build-up of constraint violations, and will require further study and testing to fully make use of alongside the Z4c and CCZ4 spacetime solvers. Due to this, most tests will be restricted to using a uniform grid, but a demonstration of the refinement-related issues will be presented for comparison to an equivalent uniform grid case. In all cases, the following tests will all use the MoL ERK-RK4 time-integrator.

6.4.1 Linear Wave

The first set of tests examines the ability of the spacetime Z4c and CCZ4 solvers to propagate a gravitational wave. This test is based on the linear wave test included in the "Apples with Apples" numerical relativity testbed proposed in Alcubierre et al. (2003); Babiuc et al. (2008). The stable propagation of a linear wave serves as a proxy for evolving gravitational wave, and will verify the accuracy of the solvers and their constraint-damping properties.

The linear wave test is setup following the specifications in Babiuc et al. (2008). A trace-free perturbation of a Minkowski spacetime in a Cartesian coordinate basis will include a linearized plane wave along the x-axis in the transverse components of the spatial metric, such that the line element takes the form

$$\mathrm{d}s^2 = -\,\mathrm{d}t^2 + \mathrm{d}x^2 + (1+B)\,\mathrm{d}y^2 + (1-B)\,\mathrm{d}z^2\,, \tag{6.44}$$

where the perturbation is given by the sinusoidal

$$B = A\sin\left[\frac{2\pi(x-t)}{d}\right], \tag{6.45}$$

with an amplitude A and wavelength d. The 3+1 split spacetime quantities can immediately be read off as

$$\alpha = 1, \quad \beta^i = 0, \quad \gamma_{ij} = \mathrm{diag}(1, 1+B, 1-B). \tag{6.46}$$

With a zero shift vector, the extrinsic curvature can be found from the time derivative of

the spatial metric (Baumgarte & Shapiro, 2010)

$$K_{ij} = -\frac{1}{2\alpha}\partial_t \gamma_{ij} = \mathrm{diag}\left(0,\, -\frac{1}{2}\partial_t B,\, \frac{1}{2}\partial_t B\right). \tag{6.47}$$

The amplitude and the wavelength of the perturbation are set to $A = 10^{-8}$ and $d = 1$.

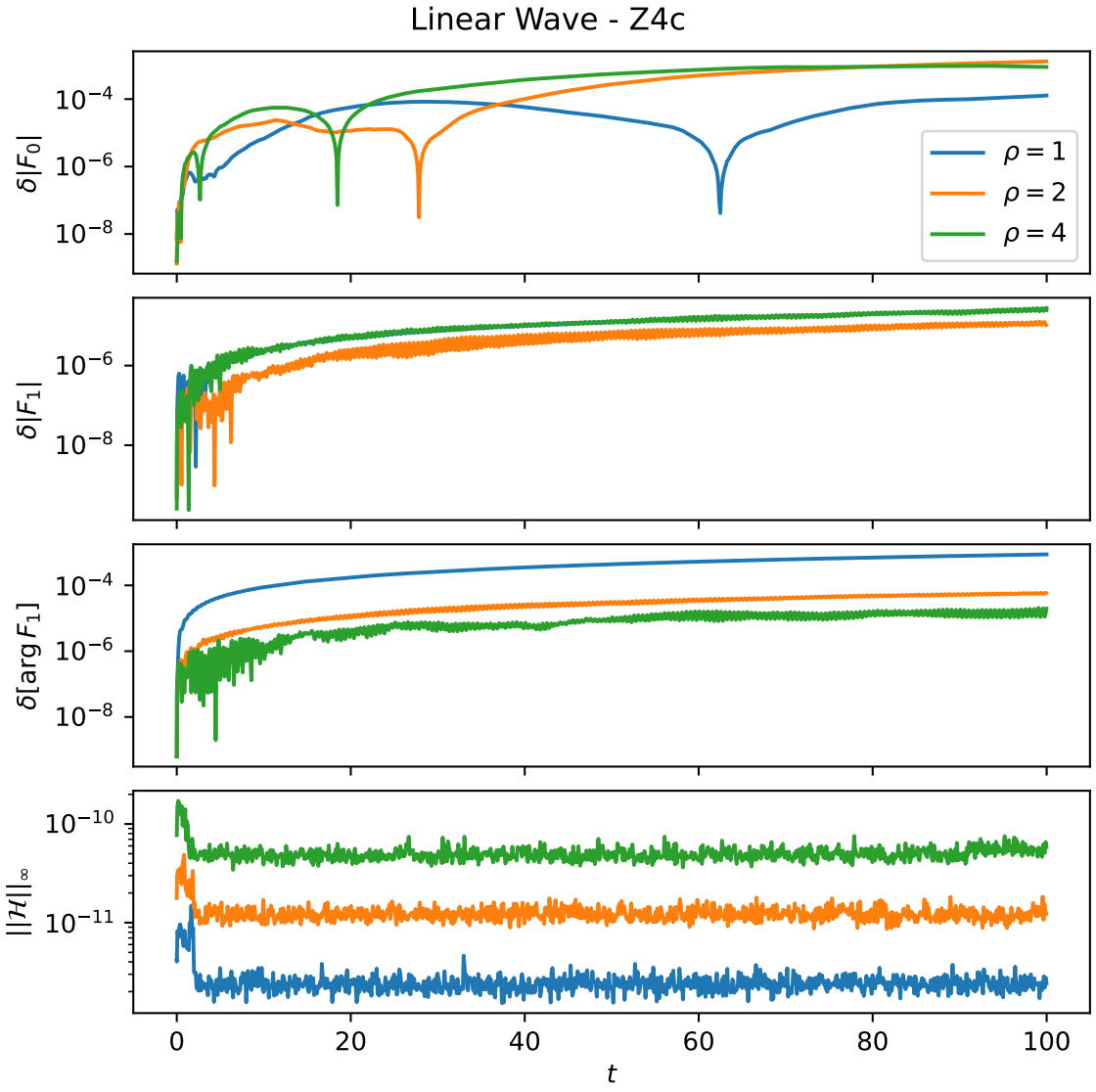

Figure 6.24: Results of the linear wave test performed by the Z4c solver. Relative errors in the offset, amplitude, and phase of γ_{zz} component's waveform to the analytic solution, and the Hamiltonian constraint violation, are presented for the three resolutions described in the text.

The domain spans $0 < x < 1$ to cover one full period of the wave along the x-axis. Three sets of simulations at different resolutions will be performed, with grid cells of size $\Delta x = \Delta y = \Delta z = 1/(50\rho)$ for $\rho = 1, 2, 4$. The limits of y, z-axes will be set such that there are four cells along each of these axes. Periodic boundary conditions will be applied along all spatial dimensions. Simulations will be ran for each grid resolution using both the Z4c

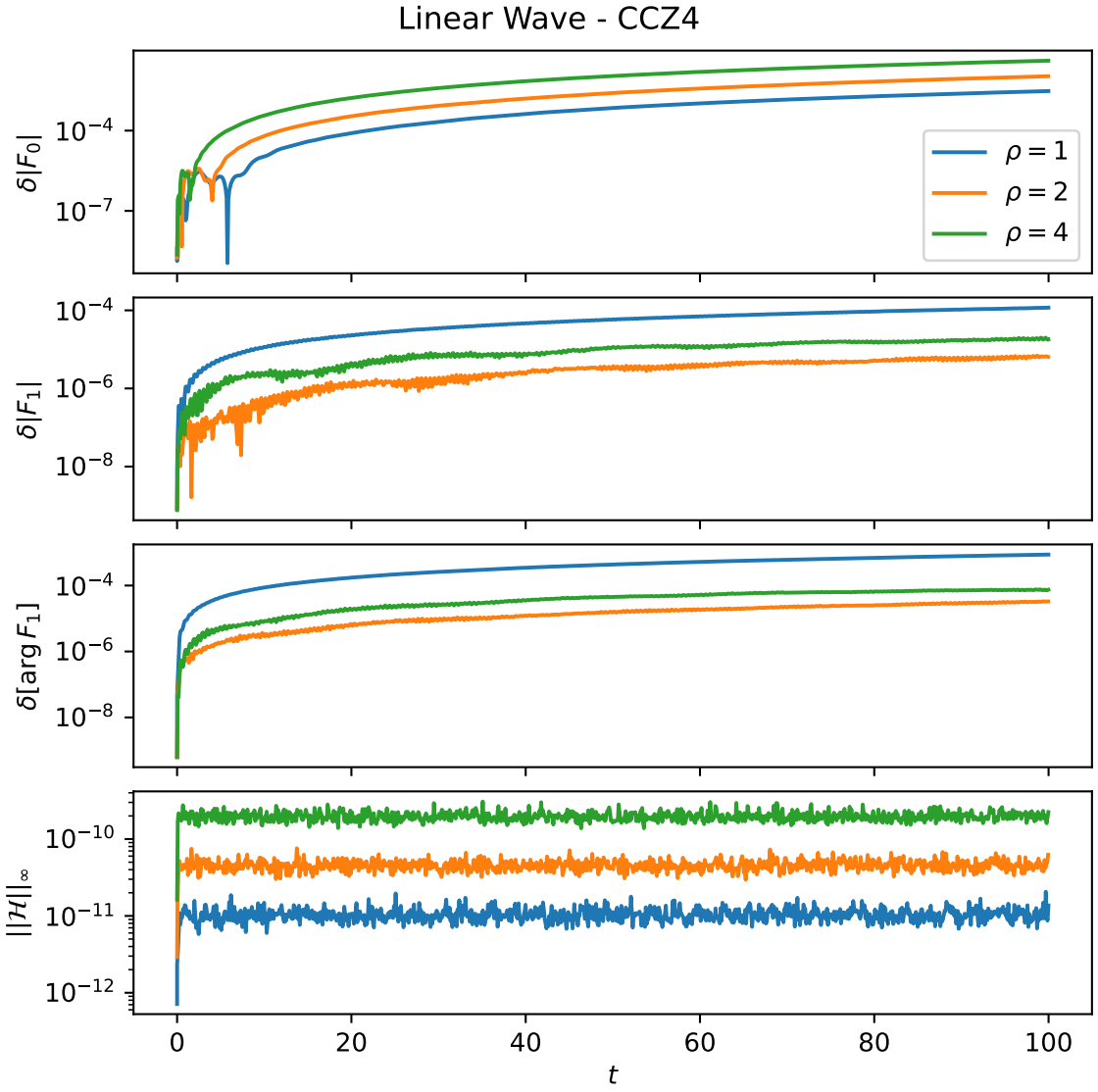

Figure 6.25: Results of the Linear wave test performed by the CCZ4 solver. Relative errors in the offset, amplitude, and phase of γ_{zz} component's waveform to the analytic solution, and the Hamiltonian constraint violation, are presented for the three resolutions described in the text.

and CCZ4 solvers, and evolved for until a time $t = 100$. Time-steps will be determined by the light-crossing time of a cell limited by $C_{\rm cfl} = 0.25$. The Z4c tests will set the damping parameters $\kappa_1 = 0.02$ and $\kappa_2 = 0$ and $\eta = 2$, while CCZ4 sets sets the damping parameters to zero such that the waveform is not damped away. Additionally, the Kreiss-Oliger damping parameters are set to $\sigma = 0.2$ for Z4c and $\sigma = 0.9$ for CCZ4. All of these parameters typically need to be tuned for different simulations and setups, as there is no consensus in the literature on values that work best in all situations.

The results for the Z4c simulations are shown in fig. 6.24, while the results for the CCZ4 simulations are shown in fig. 6.25. Each figure compares the maximum Hamiltonian constraint violation and errors in the offset, amplitude, and phase of the waves, of each resolution. The errors in the waves as compared to the analytic solution for a traveling plane wave are determined by the spatial discrete Fourier transform along the x-axis of the waves (as extracted from the transverse metric components) at each time-step. This procedure was proposed in Daverio et al. (2018), and for the waveform in the γ_{zz} component, the Fourier transform takes the form

$$F_k(t) = \frac{1}{N} \sum_{j=1}^{N} \left[\gamma_{zz}(t, x_j) - 1\right] \exp\left[-2\pi i k \big(x_j - t\big)\right], \tag{6.48}$$

where $N = 50\rho$ is the number of grid cells along the x-axis, and the metric component is explicit written to depend on the time t and its spatial location x_j, the cell-centered coordinate in the j-th cell. The $k = 0$ Fourier component corresponds to the integral of the wave over the domain; since the domain spans one full period, this integral goes to zero if the wave has not drifted away from being centered around zero. The $k = 1$ Fourier component describes the shape of the wave, with the magnitude and argument corresponding to the

amplitude and phase, respectively. The relative errors used in figs. 6.24–6.25 are defined in terms of these Fourier components as

$$\delta|F_0| = \frac{|F_0|}{A}, \quad \delta|F_1| = \frac{|F_1| - A}{A}, \quad \delta[\arg F_1] = \left|\arg F_1 + \frac{\pi}{2}\right|, \tag{6.49}$$

where the initial phase of the wave in the γ_{zz}, $-\pi/2$, is accounted for in the relative phase difference. Overall, the Z4c and CCZ4 solvers produce very similar results that, even though they contain small but acceptable amounts of constraint violation, behave stably and do not produce any significant errors in the evolution of the linear waves. These results compare favorably with similar tests of Z4c and CCZ4 solvers performed in Cao & Hilditch (2012); Daverio et al. (2018); Daszuta et al. (2021).

6.4.2 Black Hole Stability

The next test examines the ability of the spacetime solvers to maintain a stable black hole solution. These tests will further the constraint-damping properties and accuracy of the Z4c and CCZ4 solvers. Additionally, the challenges of using adaptive mesh refinement (AMR) in these problems will be examined.

An initial Schwarzschild black hole in horizon-piercing isotropic Cartesian coordinates will be used in these simulations. The Schwarzschild solution in isotropic coordinates is conformally flat, so it is straightforward to set the initial Z4 conformally-related and gauge variables. In these coordinates, the spacetime time for a Schwarzschild black hole of mass M is described by the lapse and spatial metric (Baumgarte & Shapiro, 2010)

$$\alpha = \frac{1 - M/(2r)}{1 + M/(2r)}, \quad \gamma_{ij} = \left(1 + \frac{M}{2r}\right)^4 \eta_{ij}, \tag{6.50}$$

where η_{ij} is the Minkowski spatial metric; the shift vector and extrinsic curvature are zero. In these coordinates, the event horizon of the black hole is located at $r = M/2$. In all simulations, the black hole mass will be set to $M = 1$.

The first set of tests are ran using a three-dimensional uniform grid with cell-sizes $\Delta x = \Delta y = \Delta z = 0.0125$. For practical purposes, the grid is limited to a long thin slice of size $512 \times 4 \times 4$ spanning $0 < x < 6.4$ and $0 < y, z < 0.05$. A reflective condition is used at the $x, y, z = 0$ boundaries, an outflow condition is used at the outer x-boundary, and the outer y, z boundaries are set analytically as to not produce excessive constraint-violations near the black hole. Ideally, radiative or constraint-preserving boundary conditions should be enforced at the outer boundaries to prevent the inflow of constraint violations (Ruiz et al., 2011; Hilditch et al., 2013), but the required ability to evolve data in the guard cells at these boundaries is not currently available in the MoL time-integrators `Flash-X`.[2] The Z4c simulation sets the damping parameters to $\kappa_1 = 0.07$, $\kappa_2 = 0$, and $\eta = 2$, while the CCZ4 simulation sets $\kappa_1 = 0.1$, $\kappa_2 = 0$, $\kappa_3 = 1$, and $\eta = 2$. Kreiss-Oliger dissipation is set to $\sigma = 0.02$ for Z4c, and $\sigma = 0.2$ for CCZ4. As with the linear wave tests, these parameters require tuning for each simulation.

The results for the Z4c solver are displayed in fig. 6.26, and show the maximum constraint-violations on the grid at each time-step as measured by the Hamiltonian constraint $\mathcal{H}$ from Eq. (5.20), the magnitude of the momentum constraint $\mathcal{M} = \sqrt{\mathcal{M}_i \mathcal{M}^i}$ using Eq. (5.20), the timelike (Θ) and spatial (Z^i) projections of the Z4 vector, and a combined constraint monitor $\mathcal{C} = \sqrt{\mathcal{H}^2 + \mathcal{M}_i \mathcal{M}^i + \Theta^2 + 4 Z_i Z^i}$. The levels of constraint-violations approach and remain steady around time $t = 100$. The CCZ4 results are displayed in fig. 6.27, and show the

[2]The MoL time-integrators store the intermediate states' right-hand side terms in grid-managed scratch memory, which does not contain guard cells in all backend implementations.

maximum Hamiltonian constraint-violation at every time-step; the current implementation of the CCZ4 solver does not output the additional monitors at every time-step yet. In this case, the constraint-violations stabilize much sooner around a time $t = 20$, albeit the oscillatory pattern damps out more slowly than the Z4c case. For both sets of results, the region inside the event horizon is excluded from the calculations of the maximum constraint-violations; these will be much higher in this region, but this region is acausally disconnected from the exterior of the black hole. In all cases, these results compare favorably with similar tests performed in Weyhausen et al. (2012); Clough et al. (2015), and validate the ability of the Z4c and CCZ4 solvers to evolve black hole spacetimes.

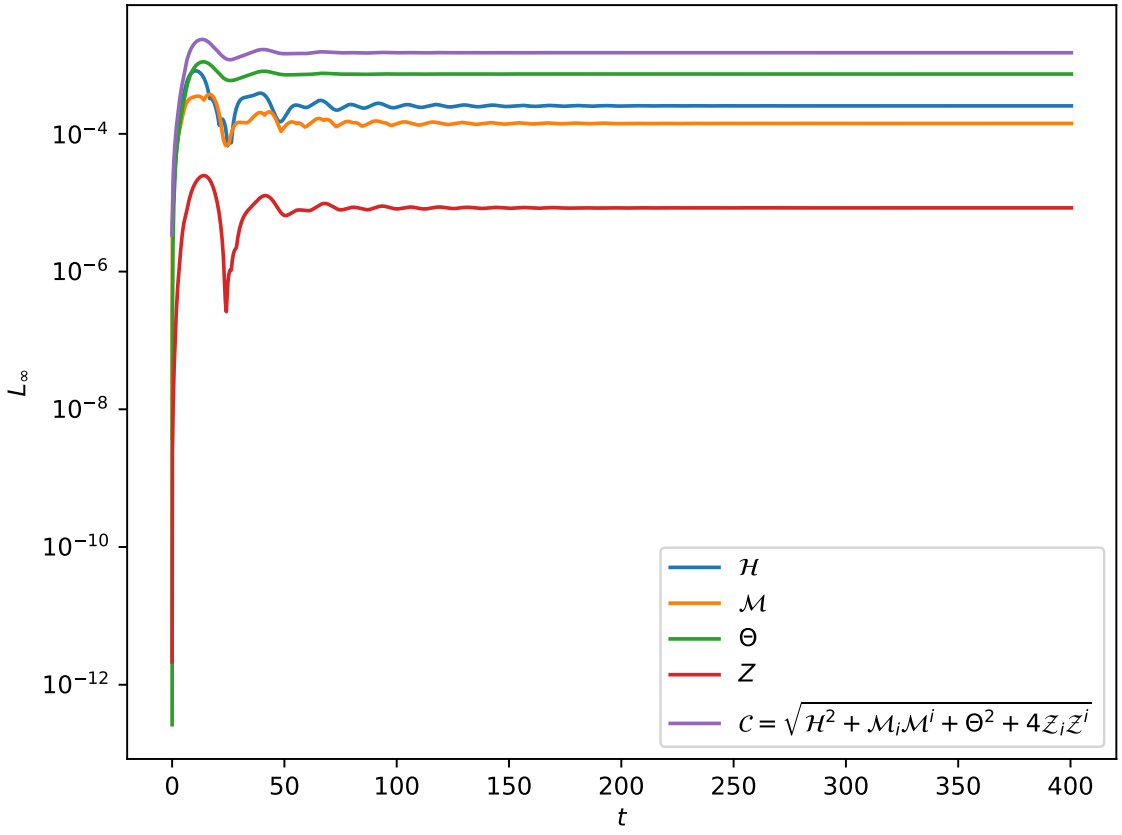

Figure 6.26: Results of the Schwarzschild black hole stability test performed by the Z4c solver using a uniform grid.

Further simulations were performed with the solvers to test compatibility with the AMR

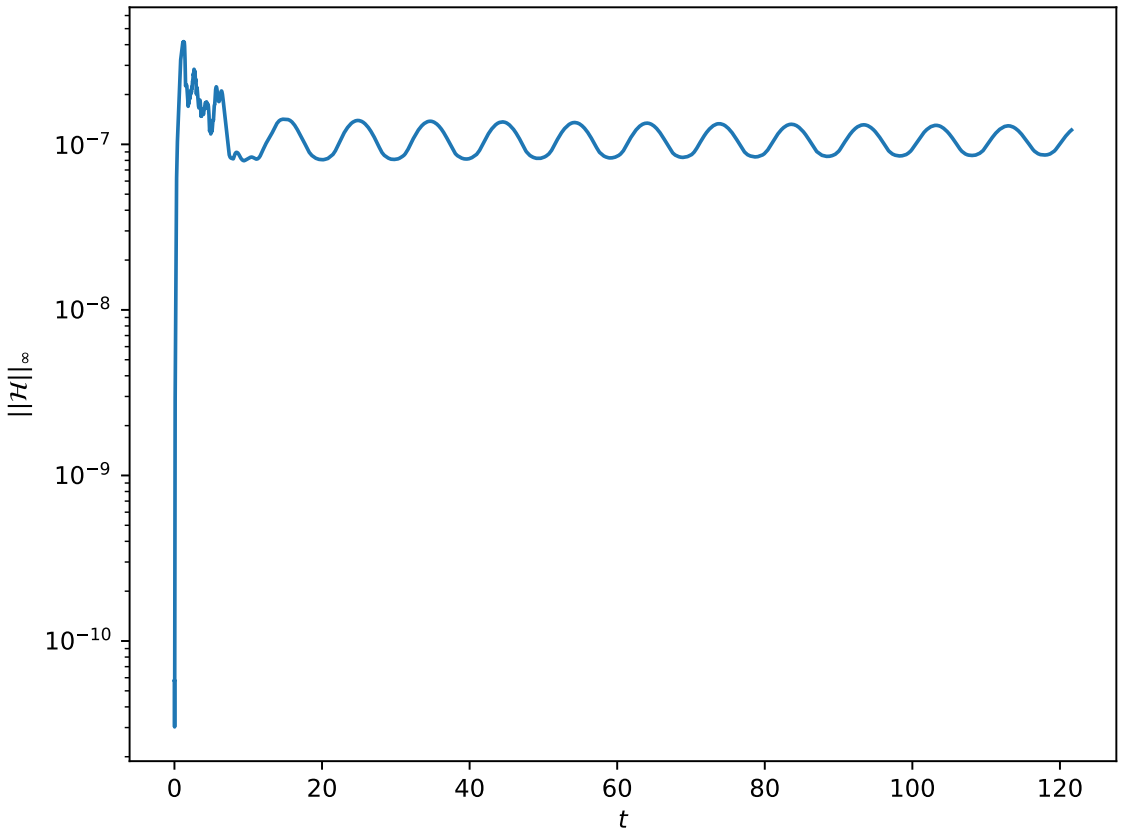

Figure 6.27: Results of the Schwarzschild black hole stability test performed by the CCZ4 solver using a uniform grid.

capabilities in `Flash-X`. The same general setups were used, but extended the long, thin slice near the x-axis to cover a domain $0 < x, y < 3.2$ and $0 < z < 1.6$, with a base resolution of $\Delta x = \Delta y = \Delta z = 0.1$. `Paramesh` was used as the AMR backend, with three levels of refinement, and the ratio of the second- and first-derivatives of the conformal factor was used as the refinement criteria (this is currently the only criteria available in `Flash-X`). This criteria was sufficient to further resolve the grid near in the regions closer to the black hole. Simulation-specific overrides of the AMR interpolators allowed for the use of the fourth-order interpolation scheme in `Paramesh`; without these overrides, higher-order interpolators are currently inaccessible via the `Flash-X` interface to `Paramesh`.

The Z4c solver managed to evolve the spacetime stably for a short time until $t = 10$,

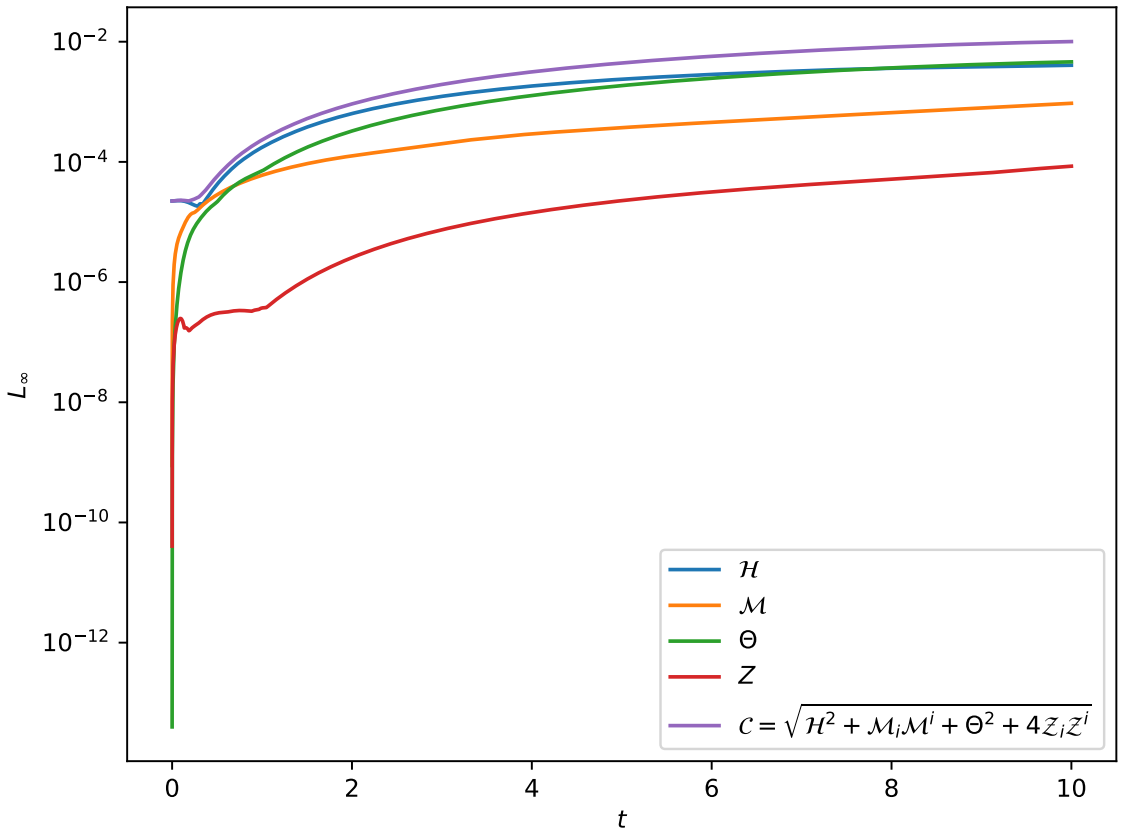

Figure 6.28: Results of the Schwarzschild black hole stability test performed by the Z4c solver using AMR.

but the constraint violations rapidly build up and prevent the simulation from running anywhere near as long as its uniform grid counterpart. The CCZ4 solver's AMR simulations, however, become unstable within a few time-steps due to large constraint-violation build-ups. Figure 6.28 shows the more rapid build-up of the constraint violations, including a much higher initial violation of the Hamiltonian constraint that occurs after the initial refinement of the grid after setting the initial data. Again, these results exclude the region inside the event horizon that where the constraint-violations are much higher. These constraint-violations quickly out pace those in the uniform grid simulations, leading to an unstable evolution.

Further examination of the results for the Z4c AMR simulations shows not only a build-up in constraint violations after the initial grid refinements, but also during guard cell filling

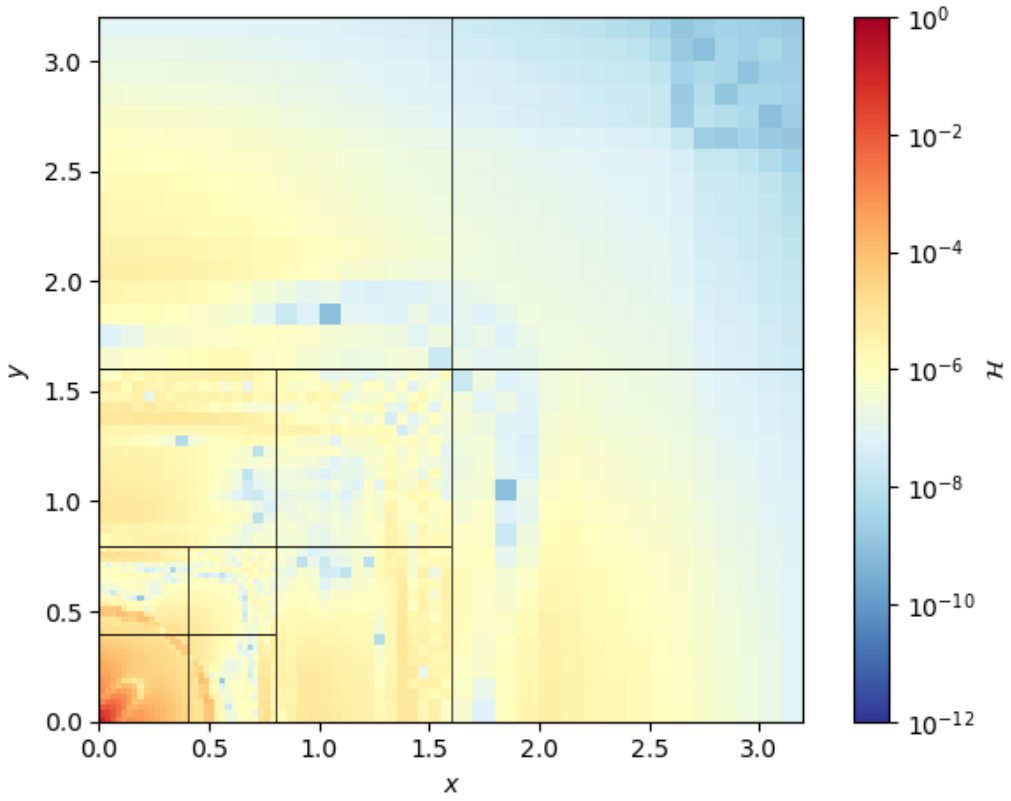

Figure 6.29: Spatial distribution of the Hamiltonian constraint violations at a time $t = 1$ during the evolution of a Schwarzschild black hole performed by the Z4c solver using AMR.

operations across the refinement boundaries. Figure 6.29 shows the spatial dependence of Hamiltonian constraint-violation in the layer of cells adjacent to the xy-plane at a time $t = 1$, and includes an overlay of the block at different refinement levels. While the largest constraint-violations exist inside the event horizon as expected, a clear pattern of higher levels of constraint-violations can be seen propagating outwards from the refinement boundaries. At a later time $t = 10$, these constraint-violations have spread throughout the domain, as can be seen in fig. 6.30. The build-up is significantly worse near the region just outside of the event horizon, leading to the subsequent unstable evolution of the spacetime.

The use of AMR is essential for practical simulations of dynamic spacetimes. Based on these initial tests, the current AMR capabilities in `Flash-X` are not compatible with the

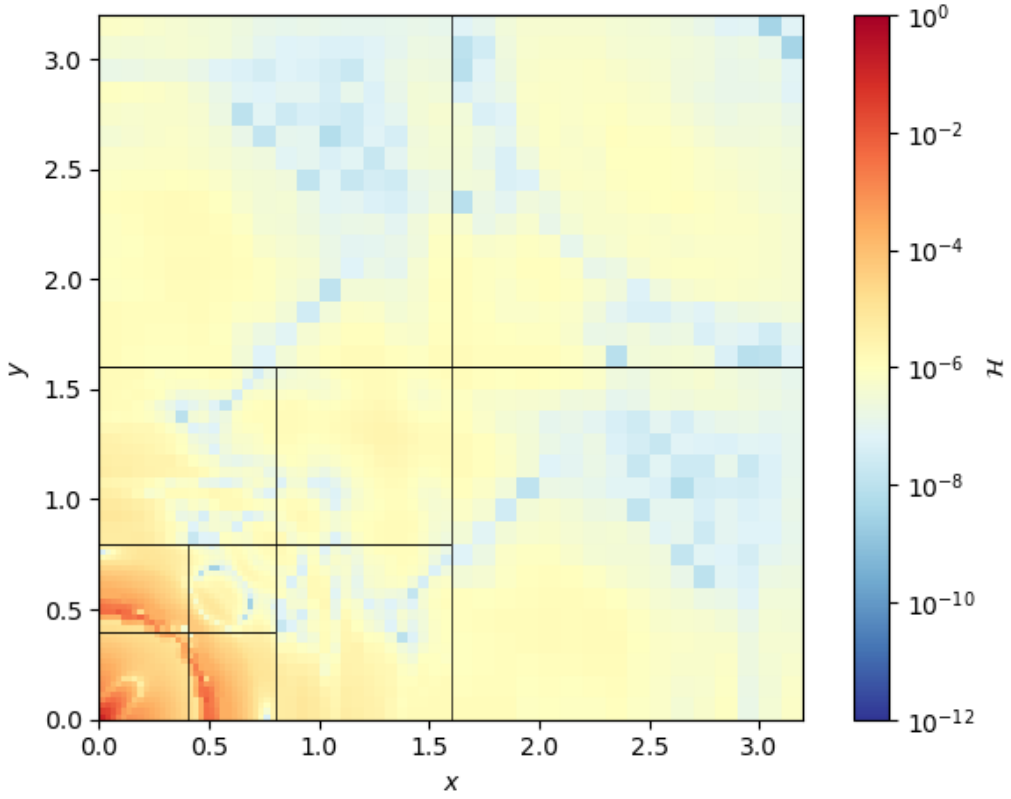

Figure 6.30: Spatial distribution of the Hamiltonian constraint violations at a time $t = 10$ during the evolution of a Schwarzschild black hole performed by the Z4c solver using AMR.

requirements for these simulations, as too high a level of constraint-violations are introduced during initial refinement operations and guard cell filling across refinement operations. Further study is necessary to fully understand the cause of these failures. Continuing work will seek to remedy these issues and provide more robust and accessible capabilities for higher-order interpolation across refinement levels, as well as improving the refinement criteria for use in dynamic spacetime evolution.

Chapter 7

Summary

This work has presented a series of general relativistic radiation transport, hydrodynamics, and spacetime solvers for use in simulations of neutron star mergers (NSMs) and core-collapse supernovae (CCSNe). These simulations are sensitive to the methods and approximations made, particularly for the inclusion of neutrinos. The formalisms presented here have all been implemented and tested in `Flash-X`. Work continues on coupling these solvers and additional capabilities necessary for performing the large-scale multi-physics simulations required for NSMs and CCSNe.

A new general relativistic moment evolution scheme for general relativistic neutrino radiation transport using a novel frequency discretization was presented in chapter 3. This scheme utilizes the tried-and-tested M1 formalism that allows for more detailed calculations than simpler methods, like neutrino leakage schemes, while providing increased computational efficiency over methods that try to directly tackle solving seven-dimensional Boltzmann equations for the neutrino distribution function. A pseudospectral discretization of the neutrino frequency "axis" permits direct use of the monochromatic moment projections and offers computational savings by requiring fewer overall points in frequency space as compared to the more commonly used frequency-bin method. This leads to a smaller memory footprint and increased efficiency and accuracy in calculating the frequency derivatives and integral present in the redshifting and neutrino-matter interaction source terms. The implemented `GRM1` solver's capabilities in flat and curved spacetimes, in the optically thin and thick limits, and its treatment of velocity-dependence at relativistic velocities were verified with a rigorous battery of test problems as presented in chapter 6.

A high-order finite-difference scheme for general relativistic hydrodynamics was presented in chapter 4. Reconstruction of the characteristic fluxes robustly captures shocks and other discontinuities that may form in the fluid. In chapter 6, the implemented `GRFD` solver was verified to resolve this features in the presence of relativistic velocities in both flat and curved spacetimes. Additionally, this implementation of this formalism proves compatible with the adaptive mesh refinement (AMR) capabilities in `Flash-X`, providing a basis for continuing work on improving the AMR capabilities for all of the new and future general relativistic solvers.

A set of dynamic spacetime solvers utilizing the Z4 formalism were presented in chapter 5 alongside a new suite of code-generation utilities for translating the often tensor-algebra-heavy equations into usable Fortran code. The Z4c and CCZ4 solvers provide constraint damping and propagation schemes for stably numerically integrating Einstein's equations of general relativity describing the structure of spacetime. The new code-generation utilities connect the robust symbolic tensor algebra module in the Python symbolic algebra package `SymPy` to improved Fortran code-generation and printing capabilities. The behavior and accuracy of both solvers were verified in chapter 6, but they also demonstrated less-than-ideal compatibility with the AMR capabilities in `Flash-X`. The use of AMR is absolutely crucial for the highly dynamic spacetimes outside of compact objects, and work continues on mitigating the constraint-violation errors produced across refinement levels and developing new refinement criteria suitable for use in NSM and CCSN simulations.

All of these solvers have been implemented in `Flash-X` along with supporting infrastructure and capabilities for tying everything together. Chiefly among these is the new method-of-lines (MoL) time discretization that provides an alternative to the rudimentary operator-split method that delegates time-integration responsibility to each physics solver

on a turn-by-turn basis. The new discretization allows for more consistent coupling of the equations managed by each solver on a per-stage level of a shared common time-integration scheme. A series of explicit-only, implicit-explicit, and implicit-explicit multi-rate methods are provided, and are easily extended with the addition of new Butcher tableau.

For the eventual application of these solvers to simulations of NSMs and CCNSe, there are some remaining challenges to address. Improvements and extensions of the AMR capabilities in `Flash-X` are underway. Higher-order interpolation methods for prolongation and restriction operations across coarse-fine boundaries are necessary for minimizing the build-up of constraint-violations when filling guard cells and creating new refinement levels. Spacetime-aware refinement criteria will also be necessary for such tasks as ensuring coarse-fine boundaries are not near an apparent horizon, which could lead to the acausal propagation of information. Generating constraint-satisfying initial data for NSMs and CCSNe and incorporating it into `Flash-X` will also be necessary. The ideal solution will make use of publicly available utilities, such as the spectral solvers for initial data provided in the `LORENE` (Gourgoulhon et al., 2001; Taniguchi et al., 2001; Taniguchi & Gourgoulhon, 2002a,b, 2003; Bejger et al., 2005; Grandclément, 2006) and `Kadath` (Grandclément, 2010; Papenfort et al., 2021) codes; an interface between solvers such as these and `Flash-X` will allow the rapid development of initial data for running sequences of NSM and CCSN simulations.

BIBLIOGRAPHY

Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2017, Physical Review Letters, 119, 161101, doi: 10.1103/PhysRevLett.119.161101

Abdikamalov, E., Burrows, A., Ott, C. D., et al. 2012, The Astrophysical Journal, 755, 111, doi: 10.1088/0004-637X/755/2/111

Alcubierre, M., Allen, G., Bona, C., et al. 2003, Classical and Quantum Gravity, 21, 589613, doi: 10.1088/0264-9381/21/2/019

Alic, D., Bona-Casas, C., Bona, C., Rezzolla, L., & Palenzuela, C. 2012, Physical Review D, 85, 064040, doi: 10.1103/PhysRevD.85.064040

Anders, E., & Grevesse, N. 1989, Geochimica et Cosmochimica Acta, 53, 197214, doi: 10.1016/0016-7037(89)90286-x

Arlandini, C., Käppeler, F., Wisshak, K., et al. 1999, The Astrophysical Journal, 525, 886, doi: 10.1086/307938

Arnowitt, R., Deser, S., & Misner, C. W. 1962, in in Gravitation: An Introduction to Current Research (Chap. 7), ed. L. Witten (John Wiley & Sons Inc), 227

Asahina, Y., Takahashi, H. R., & Ohsuga, K. 2020, The Astrophysical Journal, 901, 96, doi: 10.3847/1538-4357/abaf51

Ascher, U. M., Ruuth, S. J., & Spiteri, R. J. 1997, Applied Numerical Mathematics, 25, 151167, doi: 10.1016/s0168-9274(97)00056-1

Baade, W., & Zwicky, F. 1934, Physical Review, 46, 76, doi: 10.1103/PhysRev.46.76.2

Babiuc, M. C., Husa, S., Alic, D., et al. 2008, Classical and Quantum Gravity, 25, 125012, doi: 10.1088/0264-9381/25/12/125012

Banik, S., Hempel, M., & Bandyopadhyay, D. 2014, The Astrophysical Journal Supplement Series, 214, 22, doi: 10.1088/0067-0049/214/2/22

Banyuls, F., Font, J. A., Ibáñez, J. M., Martí, J. M., & Miralles, J. A. 1997, The Astrophysical Journal, 476, 221, doi: 10.1086/303604

Baumgarte, T. W., & Shapiro, S. L. 1998, Physical Review D, 59, 024007, doi: 10.1103/PhysRevD.59.024007

Baumgarte, T. W., & Shapiro, S. L. 2010, Numerical Relativity: Solving Einstein's Equations on the Computer (Cambridge University Press), doi: 10.1017/CBO9781139193344

Bejger, M., Gondek-Rosiska, D., Gourgoulhon, E., et al. 2005, Astronomy and Astrophysics, 431, 297306, doi: 10.1051/0004-6361:20041441

Bernuzzi, S., & Hilditch, D. 2010, Physical Review D, 81, 084003, doi: 10.1103/PhysRevD.81.084003

Blakely, P. M., & Nikiforakis, N. 2015, A&A, 583, A90, doi: 10.1051/0004-6361/201525763

Bonazzola, S., Gourgoulhon, E., Grandclément, P., & Novak, J. 2004, Physical Review D, 70, 104007, doi: 10.1103/PhysRevD.70.104007

Boyd, J. 2013, Chebyshev and Fourier Spectral Methods: Second Revised Edition, Dover Books on Mathematics (Dover Publications)

Brent, R. 2002, Algorithms for Minimization Without Derivatives, Dover Books on Mathematics (Dover Publications)

Bruenn, S. W. 1985, The Astrophysical Journal Supplement Series, 58, 771, doi: 10.1086/191056

Burrows, A., Reddy, S., & Thompson, T. A. 2006, Nuclear Physics A, 777, 356, doi: 10.1016/j.nuclphysa.2004.06.012

Butcher, J. C. 1963, Journal of the Australian Mathematical Society, 3, 185201, doi: 10.1017/S1446788700027932

Cao, Z., & Hilditch, D. 2012, Physical Review D, 85, 124032, doi: 10.1103/PhysRevD.85.124032

Cardall, C. Y., Endeve, E., & Mezzacappa, A. 2013, Physical Review D, 87, 103004, doi: 10.1103/PhysRevD.87.103004

Cernohorsky, J., & Bludman, S. A. 1994, The Astrophysical Journal, 433, 250, doi: 10.1086/174640

Cheong, P. C.-K., Ng, H. H.-Y., Lam, A. T.-L., & Li, T. G. F. 2023, The Astrophysical Journal Supplement Series, 267, 38, doi: 10.3847/1538-4365/acd931

Chinomona, R., & Reynolds, D. R. 2021, SIAM Journal on Scientific Computing, 43, A3082, doi: 10.1137/20M1354349

Clough, K., Figueras, P., Finkel, H., et al. 2015, Classical and Quantum Gravity, 32, 245011, doi: 10.1088/0264-9381/32/24/245011

Cordero-Carrión, I., Cerdá-Durán, P., Dimmelmeier, H., et al. 2009, Physical Review D, 79, 024017, doi: 10.1103/PhysRevD.79.024017

Daszuta, B., Zappa, F., Cook, W., et al. 2021, The Astrophysical Journal Supplement Series, 257, 25, doi: 10.3847/1538-4365/ac157b

Daverio, D., Dirian, Y., & Mitsou, E. 2018, arXiv e-prints, arXiv:1810.12346, doi: 10.48550/arXiv.1810.12346

Del Zanna, L., Zanotti, O., Bucciantini, N., & Londrillo, P. 2007, Astronomy and Astrophysics, 473, 11, doi: 10.1051/0004-6361:20077093

Donmez, O. 2021, European Physical Journal C, 81, 113, doi: 10.1140/epjc/s10052-021-08923-1

Dubey, A., Antypas, K., Ganapathy, M. K., et al. 2009, Parallel Computing, 35, 512, doi: https://doi.org/10.1016/j.parco.2009.08.001

Dubey, A., Weide, K., ONeal, J., et al. 2022, SoftwareX, 19, 101168, doi: https://doi.org/10.1016/j.softx.2022.101168

Dumbser, M., Guercilena, F., Köppel, S., Rezzolla, L., & Zanotti, O. 2018, Physical Review D, 97, 084053, doi: 10.1103/PhysRevD.97.084053

Duyvendak, J. J. L. 1942, Publications of the Astronomical Society of the Pacific, 54, 91, doi: 10.1086/125409

Edgar, R. 2004, New Astronomy Reviews, 48, 843, doi: 10.1016/j.newar.2004.06.001

Eichler, D., Livio, M., Piran, T., & Schramm, D. N. 1989, Nature, 340, 126, doi: 10.1038/340126a0

Fan, Y.-Z., Han, M.-Z., Jiang, J.-L., Shao, D.-S., & Tang, S.-P. 2024, Physical Review D, 109, 043052, doi: 10.1103/PhysRevD.109.043052

Farmer, R., Fields, C. E., Petermann, I., et al. 2016, The Astrophysical Journal Supplement Series, 227, 22, doi: 10.3847/1538-4365/227/2/22

Font, J. A., & Ibáñez, J. M. 1998, The Astrophysical Journal, 494, 297, doi: 10.1086/305205

Font, J. A., Ibáñez, J. M., & Papadopoulos, P. 1999, Monthly Notices of the Royal Astronomical Society, 305, 920, doi: 10.1046/j.1365-8711.1999.02459.x

Foucart, F. 2023, Living Reviews in Computational Astrophysics, 9, 1, doi: 10.1007/s41115-023-00016-y

Foucart, F., O'Connor, E., Roberts, L., et al. 2015, Physical Review D, 91, 124021, doi: 10.1103/PhysRevD.91.124021

Freiburghaus, C., Rosswog, S., & Thielemann, F. K. 1999, The Astrophysical Journal, 525, L121, doi: 10.1086/312343

Fryxell, B., Olson, K., Ricker, P., et al. 2000, The Astrophysical Journal Supplement Series, 131, 273, doi: 10.1086/317361

Galeazzi, F., Kastaun, W., Rezzolla, L., & Font, J. A. 2013, Physical Review D, 88, 064009, doi: 10.1103/PhysRevD.88.064009

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. 1995, Design patterns: elements of reusable object-oriented software (USA: Addison-Wesley Longman Publishing Co., Inc.)

Goriely, S., Demetriou, P., Janka, H. T., Pearson, J. M., & Samyn, M. 2005, Nuclear Physics A, 758, 587, doi: 10.1016/j.nuclphysa.2005.05.107

Gourgoulhon, E., Grandclément, P., Taniguchi, K., Marck, J.-A., & Bonazzola, S. 2001, Phys. Rev. D, 63, 064029, doi: 10.1103/PhysRevD.63.064029

Grandclément, P. 2006, Phys. Rev. D, 74, 124002, doi: 10.1103/PhysRevD.74.124002

Grandclément, P. 2010, Journal of Computational Physics, 229, 3334, doi: 10.1016/j.jcp.2010.01.005

Greif, S. K., Hebeler, K., Lattimer, J. M., Pethick, C. J., & Schwenk, A. 2020, The Astrophysical Journal, 901, 155, doi: 10.3847/1538-4357/abaf55

Gundlach, C., Calabrese, G., Hinder, I., & Martín-García, J. M. 2005, Classical and Quantum Gravity, 22, 3767, doi: 10.1088/0264-9381/22/17/025

Hayes, J. C., & Norman, M. L. 2003, The Astrophysical Journal Supplement Series, 147, 197, doi: 10.1086/374658

Hempel, M., Fischer, T., Schaffner-Bielich, J., & Liebendörfer, M. 2012, The Astrophysical Journal, 748, 70, doi: 10.1088/0004-637X/748/1/70

Hernandez, Walter C., J., & Misner, C. W. 1966, The Astrophysical Journal, 143, 452, doi: 10.1086/148525

Hewish, A., Bell, S. J., Pilkington, J. D. H., Scott, P. F., & Collins, R. A. 1968, Nature, 217, 709, doi: 10.1038/217709a0

Hilditch, D., Bernuzzi, S., Thierfelder, M., et al. 2013, Physical Review D, 88, 084057, doi: 10.1103/PhysRevD.88.084057

Hotokezaka, K., Beniamini, P., & Piran, T. 2018, International Journal of Modern Physics D, 27, 1842005, doi: 10.1142/S0218271818420051

Ibanez, J. M., Aloy, M. A., Font, J. A., et al. 1999, arXiv e-prints, astro, doi: 10.48550/arXiv.astro-ph/9911034

Isenberg, J. A. 2008, International Journal of Modern Physics D, 17, 265, doi: 10.1142/S0218271808011997

Jiang, G.-S., & Shu, C.-W. 1996, Journal of Computational Physics, 126, 202, doi: 10.1006/jcph.1996.0130

Kasen, D., Metzger, B., Barnes, J., Quataert, E., & Ramirez-Ruiz, E. 2017, Nature, 551, 80, doi: 10.1038/nature24453

Kurganov, A., & Tadmor, E. 2002, Numerical Methods for Partial Differential Equations, 18, 584, doi: https://doi.org/10.1002/num.10025

Kutta, W. 1901, Beitrag zur nherungsweisen Integration totaler Differentialgleichungen (Teubner)

Lattimer, J. M., & Schramm, D. N. 1974, The Astrophysical Journal, 192, L145, doi: 10.1086/181612

Lattimer, J. M., & Swesty, D. F. 1991, Nuclear Physics A, 535, 331, doi: 10.1016/0375-9474(91)90452-C

LeVeque, R. J. 2002, Finite Volume Methods for Hyperbolic Problems (Cambridge University Press), doi: 10.1017/cbo9780511791253

—. 2007, Finite difference methods for ordinary and partial differential equations: steady-state and time-dependent problems (SIAM)

Liebendörfer, M., Rampp, M., Janka, H. T., & Mezzacappa, A. 2005, The Astrophysical Journal, 620, 840, doi: 10.1086/427203

Lippuner, J., & Roberts, L. F. 2017, The Astrophysical Journal Supplement Series, 233, 18, doi: 10.3847/1538-4365/aa94cb

Liptai, D., & Price, D. J. 2019, Monthly Notices of the Royal Astronomical Society, 485, 819, doi: 10.1093/mnras/stz111

MacNeice, P., Olson, K. M., Mobarry, C., de Fainchtein, R., & Packer, C. 2000, Computer Physics Communications, 126, 330354, doi: 10.1016/s0010-4655(99)00501-9

Martín, M. P., Taylor, E. M., Wu, M., & Weirs, V. G. 2006, Journal of Computational Physics, 220, 270, doi: 10.1016/j.jcp.2006.05.009

May, M. M., & White, R. H. 1966, Phys. Rev., 141, 1232, doi: 10.1103/PhysRev.141.1232

Mayall, N. U., & Oort, J. H. 1942, Publications of the Astronomical Society of the Pacific, 54, 95, doi: 10.1086/125410

Metcalf, M., Reid, J., & Cohen, M. 2018, Modern Fortran Explained: Incorporating Fortran 2018 (Oxford University Press), doi: 10.1093/oso/9780198811893.001.0001

Meurer, A., Smith, C. P., Paprocki, M., et al. 2017, PeerJ Computer Science, 3, e103, doi: 10.7717/peerj-cs.103

Mezzacappa, A., & Bruenn, S. W. 1993, The Astrophysical Journal, 405, 669, doi: 10.1086/172395

Mezzacappa, A., & Messer, O. 1999, Journal of Computational and Applied Mathematics, 109, 281319, doi: 10.1016/s0377-0427(99)00162-4

Minerbo, G. N. 1978, Journal of Quantitative Spectroscopy and Radiative Transfer, 20, 541545, doi: 10.1016/0022-4073(78)90024-9

Misner, C. W., & Sharp, D. H. 1964, Phys. Rev., 136, B571, doi: 10.1103/PhysRev.136.B571

Misner, C. W., Thorne, K. S., & Wheeler, J. A. 2017, Gravitation

Mohr, P. J., Newell, D. B., & Taylor, B. N. 2015, CODATA Recommended Values of the Fundamental Physical Constants: 2014, Zenodo, doi: 10.5281/zenodo.22826

Mösta, P., Mundim, B. C., Faber, J. A., et al. 2014, Classical and Quantum Gravity, 31, 015005, doi: 10.1088/0264-9381/31/1/015005

Müller, B., Janka, H.-T., & Dimmelmeier, H. 2010, The Astrophysical Journal Supplement Series, 189, 104, doi: 10.1088/0067-0049/189/1/104

Murchikova, E. M., Abdikamalov, E., & Urbatsch, T. 2017, Monthly Notices of the Royal Astronomical Society, 469, 1725, doi: 10.1093/mnras/stx986

O'Connor, E. 2015, The Astrophysical Journal Supplement Series, 219, 24, doi: 10.1088/0067-0049/219/2/24

O'Connor, E., & Ott, C. D. 2010, Classical and Quantum Gravity, 27, 114103, doi: 10.1088/0264-9381/27/11/114103

O'Connor, E. P., & Couch, S. M. 2018a, The Astrophysical Journal, 854, 63, doi: 10.3847/1538-4357/aaa893

—. 2018b, The Astrophysical Journal, 865, 81, doi: 10.3847/1538-4357/aadcf7

Oettinger, D., Mendoza, M., & Herrmann, H. J. 2013, Physical Review E, 88, 013302, doi: 10.1103/PhysRevE.88.013302

Oppenheimer, J. R., & Volkoff, G. M. 1939, Phys. Rev., 55, 374, doi: 10.1103/PhysRev.55.374

Pajkos, M. A. 2022, PhD thesis, doi: 10.25335/JW6W-3P83

Papenfort, L. J., Tootle, S. D., Grandclément, P., Most, E. R., & Rezzolla, L. 2021, Physical Review D, 104, 024057, doi: 10.1103/PhysRevD.104.024057

Pareschi, L., & Russo, G. 2005, Journal of Scientific Computing, 25, 129, doi: 10.1007/BF02728986

Patrignani, C., Particle Data Group, Agashe, K., et al. 2016, Chinese Physics C, 40, 100001, doi: 10.1088/1674-1137/40/10/100001

Perego, A., Bernuzzi, S., & Radice, D. 2019, European Physical Journal A, 55, 124, doi: 10.1140/epja/i2019-12810-7

Perego, A., Cabezón, R. M., & Käppeli, R. 2016, The Astrophysical Journal Supplement Series, 223, 22, doi: 10.3847/0067-0049/223/2/22

Peterson, A. J., Willcox, D., & Mösta, P. 2023, Classical and Quantum Gravity, 40, 245013, doi: 10.1088/1361-6382/ad0b37

Pons, J. A., Ibáñez, J. M., & Miralles, J. A. 2000, Monthly Notices of the Royal Astronomical Society, 317, 550, doi: 10.1046/j.1365-8711.2000.03679.x

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. 2007, Numerical Recipes 3rd Edition: The Art of Scientific Computing, 3rd edn. (USA: Cambridge University Press)

Pretorius, F. 2005, Classical and Quantum Gravity, 22, 425, doi: 10.1088/0264-9381/22/2/014

Radia, M., Sperhake, U., Drew, A., et al. 2022, Classical and Quantum Gravity, 39, 135006, doi: 10.1088/1361-6382/ac6fa9

Radice, D., Bernuzzi, S., Perego, A., & Haas, R. 2022, Monthly Notices of the Royal Astronomical Society, 512, 1499, doi: 10.1093/mnras/stac589

Radice, D., Galeazzi, F., Lippuner, J., et al. 2016, Monthly Notices of the Royal Astronomical Society, 460, 3255, doi: 10.1093/mnras/stw1227

Radice, D., Perego, A., Hotokezaka, K., et al. 2018, The Astrophysical Journal, 869, 130, doi: 10.3847/1538-4357/aaf054

Radice, D., & Rezzolla, L. 2012, Astronomy & Astrophysics, 547, A26, doi: 10.1051/0004-6361/201219735

Radice, D., Rezzolla, L., & Galeazzi, F. 2014, Classical and Quantum Gravity, 31, 075012, doi: 10.1088/0264-9381/31/7/075012

Rampp, M., & Janka, H.-T. 2002, Astronomy & Astrophysics, 396, 361392, doi: 10.1051/0004-6361:20021398

Rezzolla, L., Baiotti, L., Giacomazzo, B., Link, D., & Font, J. A. 2010, Classical and Quantum Gravity, 27, 114105, doi: 10.1088/0264-9381/27/11/114105

Rezzolla, L., & Zanotti, O. 2013, Relativistic Hydrodynamics (Oxford University PressOxford), doi: 10.1093/acprof:oso/9780198528906.001.0001

Rezzolla, L., Zanotti, O., & Pons, J. A. 2003, Journal of Fluid Mechanics, 479, 199, doi: 10.1017/S0022112002003506

Roberts, L. F., Ott, C. D., Haas, R., et al. 2016, The Astrophysical Journal, 831, 98, doi: 10.3847/0004-637X/831/1/98

Roman, S. 2007, Advanced Linear Algebra, Graduate Texts in Mathematics (Springer New York)

Rosswog, S. 2015, Living Reviews in Computational Astrophysics, 1, 1, doi: 10.1007/lrca-2015-1

Rosswog, S., & Davies, M. B. 2002, Monthly Notices of the Royal Astronomical Society, 334, 481, doi: 10.1046/j.1365-8711.2002.05409.x

Rosswog, S., & Liebendörfer, M. 2003, Monthly Notices of the Royal Astronomical Society, 342, 673, doi: 10.1046/j.1365-8711.2003.06579.x

Rosswog, S., Sollerman, J., Feindt, U., et al. 2018, Astronomy and Astrophysics, 615, A132, doi: 10.1051/0004-6361/201732117

Ruchlin, I., Etienne, Z. B., & Baumgarte, T. W. 2018, Physical Review D, 97, 064036, doi: 10.1103/PhysRevD.97.064036

Ruffert, M., Janka, H. T., & Schaefer, G. 1996, Astronomy and Astrophysics, 311, 532, doi: 10.48550/arXiv.astro-ph/9509006

Ruiz, M., Hilditch, D., & Bernuzzi, S. 2011, Physical Review D, 83, doi: 10.1103/physrevd.83.024025

Runge, C. 1895, Mathematische Annalen, 46, 167, doi: 10.1007/BF01446807

Shibata, M., Kiuchi, K., Sekiguchi, Y., & Suwa, Y. 2011, Progress of Theoretical Physics, 125, 1255, doi: 10.1143/PTP.125.1255

Shibata, M., & Nakamura, T. 1995, Physical Review D, 52, 5428, doi: 10.1103/PhysRevD.52.5428

Shu, C.-W., & Osher, S. 1988, Journal of Computational Physics, 77, 439, doi: 10.1016/0021-9991(88)90177-5

Skinner, M. A., Dolence, J. C., Burrows, A., Radice, D., & Vartanyan, D. 2019, The Astrophysical Journal Supplement Series, 241, 7, doi: 10.3847/1538-4365/ab007f

Smit, J. M., Cernohorsky, J., & Dullemond, C. P. 1997, Astronomy and Astrophysics, 325, 203. https://ui.adsabs.harvard.edu/abs/1997A&A...325..203S

Sotani, H., Nishimura, N., & Naito, T. 2022, Progress of Theoretical and Experimental Physics, 2022, 041D01, doi: 10.1093/ptep/ptac055

Steiner, A. W., Hempel, M., & Fischer, T. 2013a, The Astrophysical Journal, 774, 17, doi: 10.1088/0004-637X/774/1/17

Steiner, A. W., Lattimer, J. M., & Brown, E. F. 2013b, The Astrophysical Journal, 765, L5, doi: 10.1088/2041-8205/765/1/L5

Symbalisty, E., & Schramm, D. N. 1982, Astrophysical Letters, 22, 143

Taniguchi, K., & Gourgoulhon, E. 2002a, Phys. Rev. D, 65, 044027, doi: 10.1103/PhysRevD.65.044027

—. 2002b, Phys. Rev. D, 66, 104019, doi: 10.1103/PhysRevD.66.104019

—. 2003, Phys. Rev. D, 68, 124025, doi: 10.1103/PhysRevD.68.124025

Taniguchi, K., Gourgoulhon, E., & Bonazzola, S. 2001, Physical Review D, 64, 064012, doi: 10.1103/PhysRevD.64.064012

Taylor, J. H., Fowler, L. A., & McCulloch, P. M. 1979, Nature, 277, 437, doi: 10.1038/277437a0

Thorne, K. S. 1981, Monthly Notices of the Royal Astronomical Society, 194, 439, doi: 10.1093/mnras/194.2.439

Tolman, R. C. 1939, Phys. Rev., 55, 364, doi: 10.1103/PhysRev.55.364

Toro, E. F. 2009, Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction (Berlin, Heidelberg: Springer Berlin Heidelberg), doi: 10.1007/b79761

van Riper, K. A., & Lattimer, J. M. 1981, The Astrophysical Journal, 249, 270, doi: 10.1086/159285

Wald, R. M. 1984, General Relativity (Chicago, USA: Chicago Univ. Pr.), doi: 10.7208/chicago/9780226870373.001.0001

Weih, L. R., Olivares, H., & Rezzolla, L. 2020, Monthly Notices of the Royal Astronomical Society, 495, 2285, doi: 10.1093/mnras/staa1297

Weyhausen, A., Bernuzzi, S., & Hilditch, D. 2012, Physical Review D, 85, 024038, doi: 10.1103/PhysRevD.85.024038

Wilson, J. R. 1972, The Astrophysical Journal, 173, 431, doi: 10.1086/151434

Wilson, J. R., & Mathews, G. J. 1995, Phys. Rev. Lett., 75, 4161, doi: 10.1103/PhysRevLett.75.4161

Zanotti, O., Roedig, C., Rezzolla, L., & Del Zanna, L. 2011, Monthly Notices of the Royal Astronomical Society, 417, 2899, doi: 10.1111/j.1365-2966.2011.19451.x

Zhang, W., & MacFadyen, A. I. 2006, The Astrophysical Journal Supplement Series, 164, 255, doi: 10.1086/500792

APPENDIX A. Conventions

A.1. Notation

Throughout this work, the standard metric $(-,+,+,+)$ used in numerical relativity will be assumed. All tensors will be indexed by the lower-case Latin letters $a, b, c, \ldots$ and assume values of $0, 1, 2, 3$ for full four-dimensional spacetime components, where 0 is the time component, and $1, 2, 3$ are the spatial components. However, the indices i, j, k, l, m, n will refer strictly to spatial-only index values of $1, 2, 3$. The choice of Latin indices in lieu of the more commonly used Greek indices is made to avoid confusion with various quantities, such as the lapse α and shift β, as well as the use of ν and σ as subscripts specifying the frequency and species of a neutrino. The restrictions on the index values do not apply when explicitly appearing in summations where their limiting values are given.

All symbolic tensor equations will assume the Einstein summation convention unless otherwise noted. All covariant-contravariant pairs of "dummy" indices will imply summation over all possible index values, e.g., computing the contraction

$$v_i v^i = v_1 v^1 + v_2 v^2 + v_3 v^3. \tag{A.1}$$

Free indices do not imply summation, even when repeated, and represent a set of values or equations for specific tensor components. For example, lowering the index of the velocity v^j with the spatial metric γ_{ij} using this convention takes the form

$$v_i = \gamma_{ij} v^j \quad \Longrightarrow \quad \begin{aligned} v_1 &= \gamma_{11} v^1 + \gamma_{12} v^2 + \gamma_{13} v^3, \\ v_2 &= \gamma_{21} v^1 + \gamma_{22} v^2 + \gamma_{23} v^3, \\ v_3 &= \gamma_{31} v^1 + \gamma_{32} v^2 + \gamma_{33} v^3. \end{aligned} \tag{A.2}$$

For instances where a repeated index appears on one side of an equation but in two covariant or contravariant slots, this will represent only a subset of the tensor's components, e.g., γ_{ii} refers to only the diagonal components of the spatial metric.

A.2. Geometrized Units

A geometrized systems of units will be used throughout this work to simplify the notation in the numerous equations as well as to provide reasonable numerical values for the scales present in neutron star mergers and core-collapse supernovae simulations. The constants $G = c = M_\odot = 1$ will be used, where G is Newton's gravitational constant, $M_\odot$ is the mass of the sun, and c is the speed of light in a vacuum. The implementations of the solvers presented in this work use the CODATA 2014 (Mohr et al., 2015) and Particle Data Group 206 Summary Tables (Patrignani et al., 2016)[1]

$$G = 6.67408 \times 10^{-8}\ \mathrm{cm}^3\,\mathrm{g}^{-1}\,\mathrm{s}^{-2}, \tag{A.3}$$

$$c = 2.99792458 \times 10^{10}\ \mathrm{cm}\,\mathrm{s}^{-1}, \tag{A.4}$$

$$M_\odot = 1.98848 \times 10^{33}\ \mathrm{g}. \tag{A.5}$$

In this system of units, the basic scales for mass $[M]$, length $[L]$, and time $[T]$ are set to

$$[M] = M_\odot, \quad [L] = \frac{GM_\odot}{c^2}, \quad [T] = \frac{GM_\odot}{c^3}. \tag{A.6}$$

Derived units for areas, volumes, densities, energies, pressures, and opacities can be written in combinations of these units. All quantities measured by these units will be given either

[1] The use of these older sets of values is for compatibility with values used throughout `Flash-X`.

in powers of $M_\odot$ or c, providing a convenient scale for neutron star mergers. Neutron stars have masses of $\mathcal{O}(M_\odot)$ and radii of $\mathcal{O}\big(10^6 \text{ cm}\big)$, which gives masses of $\mathcal{O}(1)$ and radii of $\mathcal{O}(10)$ in geometrized units. Velocities are then simply fractions of the speed of light, such that the magnitude of any velocity v is limited to values $0 \leq v < 1$.

Temperatures are treated differently for compatibility with nuclear equation of state tables. These tables typically represent temperatures as energies measured in MeV. This is accomplished by setting the Boltzmann constant $k_B = 1$, which in units of MeV per Kelvin is (Mohr et al., 2015)

$$k_B = 8.6173303 \times 10^{-11} \text{ MeV K}^{-1}. \tag{A.7}$$

This allows both temperatures and neutrino frequencies (when further assuming the Planck constant $h = 1$) in neutron star mergers to take average values ranges from ones to hundreds of MeV.

APPENDIX B. Tolman-Oppenheimer-Volkoff Equations

The Tolman-Oppenheimer-Volkoff (TOV) equations, named after the authors of Tolman (1939); Oppenheimer & Volkoff (1939), describe the static equilibrium solution for relativistic stars. Solutions of the TOV equation are commonly used to describe cold, non-rotating neutron stars. This appendix provides a brief overview of the TOV equations and an alternative form more suitable for numerical integration. The standard form of the TOV equations is based on the presentation in Baumgarte & Shapiro (2010), and reproduced in part for reference below.

Assuming a spherically symmetric spacetime containing an ideal fluid sphere described by an equation of state $p(\rho, \epsilon)$, the line element of the metric takes the form

$$\mathrm{d}s^2 = -e^{2\Phi(r)}\,\mathrm{d}t^2 + e^{2\lambda(r)}\,\mathrm{d}r^2 + r^2\,\mathrm{d}\Omega^2\,, \tag{B.1}$$

where $\Phi(r)$ and $\lambda(r)$ are two metric potentials that only depend on the radius, and $\mathrm{d}\Omega = \mathrm{d}\theta^2 + \sin^2\theta\,\mathrm{d}\phi^2$ is the differential solid angle on the unit sphere. Since the solution exterior to the star must match the Schwarzschild solution, $\lambda(r)$ can be related to the enclosed mass of the star $m(r)$ as

$$e^{2\lambda(r)} = \left[1 - \frac{2m(r)}{r}\right]^{-1}. \tag{B.2}$$

This guarantees that a star of mass M and radius R satisfies

$$\begin{aligned} m(r \geq R) &= M, \\ \lambda(r \geq R) &= -\frac{1}{2}\ln\left(1 - \frac{2M}{r}\right) \end{aligned} \tag{B.3}$$

everywhere outside of the star. Using the spacetime metric described by Eq. (B.1) in the

Einstein field equations, Eq. (5.1), along with the stress-energy tensor of an ideal perfect fluid in Eq. (4.1), interior solution for the static equilibrium of the star, i.e., for $r \leq R$, is described by the system of differential equations for the mass, pressure, and metric potential $\Phi(r)$

$$\frac{\mathrm{d}m}{\mathrm{d}r} = 4\pi r^2 e, \tag{B.4}$$

$$\frac{\mathrm{d}\Phi}{\mathrm{d}r} = \frac{m + 4\pi r^3 p}{r(r-2m)}, \tag{B.5}$$

$$\frac{\mathrm{d}p}{\mathrm{d}r} = -(e+p)\frac{\mathrm{d}\Phi}{\mathrm{d}r}, \tag{B.6}$$

where $e = \rho(1+\epsilon)$ is the total energy density.

Each possible solution is determined by the central values

$$m(r=0) = 0, \quad p(r=0) = p_c, \tag{B.7}$$

where p_c is the central pressure; other thermodynamic quantities relate to this pressure via the fluid's equation of state. The central value of Φ is not necessary for integration since Eq. (B.5) is linear in Φ; this will typically be set to zero and later matched onto the Schwarzschild solution. With these initial conditions, Eqns. (B.4)–(B.6) can be integrated out to the surface of the sphere. Unfortunately, outside of a constant density solution, there is no way to determine the radius R of the surface from the initial conditions alone. The surface will be located where the pressure goes to zero, i.e., $p(r = R) = 0$. Directly integrating Eqns. (B.4)–(B.6) will require checking this condition at every integration step, and repeating the final step at increasingly smaller radial step sizes to find the exact location without stepping past the surface of the star.

Instead, it is simpler to recast Eqns. (B.4)–(B.6) as derivatives with respect to the pressure, such that the equations can be integrated over the domain $p_c \leq p \leq 0$. The new system takes the form, including a new equation for the radius of the star

$$\frac{\mathrm{d}\Phi}{\mathrm{d}p} = -\frac{1}{e+p}, \tag{B.8}$$

$$\frac{\mathrm{d}r}{\mathrm{d}p} = \frac{r(r-2m)}{m+4\pi r^3 p}\frac{\mathrm{d}\Phi}{\mathrm{d}p}, \tag{B.9}$$

$$\frac{\mathrm{d}m}{\mathrm{d}p} = 4\pi r^2 e \frac{\mathrm{d}r}{\mathrm{d}p}. \tag{B.10}$$

These equations can now be integrated out to $p = 0$ by any explicit integration method suitable for initial value problems, such as the classic fourth-order Runge-Kutta method. Once the surface of the star has been found, the metric potential $\Phi(r)$ can be found by matching it onto the Schwarzschild solution

$$\Phi(R) = \frac{1}{2}\ln\left(1 - \frac{2M}{R}\right), \tag{B.11}$$

which can then be added to $\Phi(r \leq R)$ to determine the interior solution (provided that $\Phi(r = 0) = 0$ was chosen as the initial condition). The full exterior solution for $r > R$ is then just the Schwarzschild solution in terms of the mass M.

The TOV equations do permit an analytic solution for the case of uniform density. This limiting case shows that the maximum compactness of a non-rotating relativistic star is $M/R < 4/9$ (Baumgarte & Shapiro, 2010). Please refer to Wald (1984); Rezzolla & Zanotti (2013) for more on the uniform density solution.

APPENDIX C. Eulerian 3+1 Decomposition of the Spacetime

This appendix summarizes the 3+1 decomposition of the spacetime commonly used in numerical relativity as presented in Baumgarte & Shapiro (2010), and will highlight some of the necessary concepts and identities used throughout this dissertation. In a 3+1 decomposition, the full spacetime is split into spacelike hypersurfaces Σ_t each at a constant time t as measured in the frame of a distant Eulerian observer. In this frame, the four-velocity of the Eulerian observer is timelike and normal to Σ_t, and is defined as

$$n_a = -\alpha \boldsymbol{\nabla}_a t, \tag{C.1}$$

where the lapse function α characterizes the separation between the spacetime hypersurfaces. In this 3+1 decomposition, the invariant line element takes the form

$$\mathrm{d}s^2 = g_{ab}\,\mathrm{d}x^a\,\mathrm{d}x^b = -\alpha^2\,\mathrm{d}t^2 + \gamma_{ij}\Big(\mathrm{d}x^i + \beta^i\,\mathrm{d}t\Big)\Big(\mathrm{d}x^j + \beta^j\,\mathrm{d}t\Big), \tag{C.2}$$

where g_{ab} is the spacetime metric, β^i is the shift vector characterizing the coordinate shift from one hypersurface to the next, and γ_{ij} is the spatial metric on Σ_t. Using the normalization $n_a n^a = -1$, Eqns. (C.1)–(C.2) show that the Eulerian four-velocity can be written as

$$n_a = \Big(-\alpha, 0^i\Big), \quad n^a = \Big(\alpha^{-1}, \beta^i\Big), \tag{C.3}$$

where 0^i is the spatial zero-vector.

The spatial metric γ_{ij} acts as a projection operator onto Σ_t, and from Eqns. (C.2)–(C.3)

it follows that

$$\gamma_{ab} = g_{ab} + n_a n_b. \tag{C.4}$$

Since $\gamma_{ab} n^b = 0$, it is useful to note that $\gamma^{a0} = 0$, and for spacelike vectors in the Eulerian frame, γ_{ab} plays the role of the metric, e.g. for a spatial vector v^i, the spatial-components of the corresponding one-form are $v_i = \gamma_{ij} v^j$.

Terms containing covariant-derivatives of the Eulerian four-velocity are commonly encountered when projecting evolution equations into the spacelike hypersurfaces. Using Eq. (C.1) and Eq. (C.3), some useful identities involving these covariant derivatives are

$$n^a \boldsymbol{\nabla}_b n_a = \boldsymbol{\nabla}_b (n^a n_b) - n_a \boldsymbol{\nabla}_b n^a = -n^a \boldsymbol{\nabla}_b n_a = 0, \tag{C.5}$$

where the final step follows from the compatibility of the covariant derivative with the spacetime metric, i.e. $\boldsymbol{\nabla}_a g_{bc} = 0$, and

$$n^b \boldsymbol{\nabla}_b n_a = \gamma^a{}_b \boldsymbol{\nabla}_b \ln \alpha = \partial_a \ln \alpha. \tag{C.6}$$

The extrinsic curvature of the spacetime hypersurfaces relative to the full spacetime can be characterized by the change in the Eulerian normal four-velocities, and with the relations in Eqns. (C.5)–(C.6) is given as

$$K_{ab} = -\boldsymbol{\nabla}_b n_a - n_b n^c \boldsymbol{\nabla}_c n_a = -\boldsymbol{\nabla}_b n_a - n_b \partial_a \ln \alpha, \tag{C.7}$$

from which it follows that $n^a K_{ab} = 0$.

For spatial vectors, it follows from Eqns. (C.2)–(C.3) that

$$n_a v^a = n^a v_a = 0 \quad \Longrightarrow \quad v_a = \Big(v_j \beta^j, v_i\Big), \tag{C.8}$$

leading to some useful identities involving the Eulerian four-velocity and spatial vectors

$$n^a \boldsymbol{\nabla}_b v_a = \boldsymbol{\nabla}_b (n^a v_a) - v^a \boldsymbol{\nabla}_b n_a = -v^a \boldsymbol{\nabla}_b n_a \tag{C.9}$$

and

$$v_a \frac{\partial n^a}{\partial x^b} = \Big(v_k \beta^k\Big) \frac{\partial}{\partial x^b}\left(\frac{1}{\alpha}\right) - v_k \left[\beta^k \frac{\partial}{\partial x^b}\left(\frac{1}{\alpha}\right) + \frac{1}{\alpha}\frac{\partial \beta^k}{\partial x^b}\right] = -\frac{v_k}{\alpha}\frac{\partial \beta^k}{\partial x^b} \tag{C.10}$$

In a 3+1 decomposition of the spacetime, the covariant divergence of a tensor $Q^{ab}{}_c$ can be written as

$$\boldsymbol{\nabla}_a Q^{ab}{}_c = \frac{1}{\alpha\sqrt{\gamma}}\frac{\partial}{\partial x^a}\Big(\alpha\sqrt{\gamma} Q^{ab}{}_c\Big) + \Gamma^b{}_{da} Q^{ad}{}_c - \Gamma^d{}_{ca} Q^{ab}{}_d, \tag{C.11}$$

where the Christoffel symbols are

$$\Gamma^a{}_{bc} = \frac{1}{2} g^{ad}\left(\frac{\partial g_{cd}}{\partial x^b} + \frac{\partial g_{bd}}{\partial x^c} - \frac{\partial g_{bc}}{\partial x^d}\right) \tag{C.12}$$

APPENDIX D. Deriving the Moment Evolution Equations

This appendix provides the lengthier parts of the derivation of the radiation transport moment evolution equations presented in chapter 3. The covariant derivative of the arbitrary-rank moment expansion, Eulerian-projection of the second-rank moment, and the projection of the redshifting frequency-space advection term will be considered separately, and will make use of the 3+1 split spacetime and various identities covered in appendix C. In all of the following derivations, the explicit frequency-subscript will be dropped for clarity — all radiation quantities will be assumed to be for a specific-frequency ν, e.g. $E \equiv E_{(\nu)}$.

D.1. Covariant Derivative of the Moment Expansion

This section derives the evolution equations for the moment-expansion of radiation distribution function. With a more compact notation, the arbitrary-rank moment-expansion repeated here for convenience

$$M^{A_k} \equiv M^{A_k}\Big(x^b\Big) = \int \mathrm{d}V_{p'} \, \frac{\delta\big(\nu + u_c p'^c\big)}{(-u_c p'^c)^{k-2}} \left(\prod_{a_k \in A_k} p'^{a_k} \right) f_{(\nu')}, \tag{D.1}$$

where $f_{(\nu')} \equiv f\Big(x^b, p'^a\Big)$. Taking the covariant-divergence of Eq. (D.1) with respect to the spacetime coordinates x^b results in the three separate terms

$$\Big[\boldsymbol{\nabla}_b M^{A_k b}\Big]_1 = -\int \frac{\mathrm{d}V_{p'}}{(-u_c p'^c)^{k-1}} \left(\prod_{a_k \in A_k} p'^{a_k} \right) p'^b f_{(\nu')} \left[p'^c \boldsymbol{\nabla}_b u_c \frac{\partial \delta\big(\nu + u_c p'^c\big)}{\partial(-u_c p'^c)} \right], \tag{D.2}$$

$$\Big[\boldsymbol{\nabla}_b M^{A_k b}\Big]_2 = \int \mathrm{d}V_{p'} \, \frac{\delta\big(\nu + u_c p'^c\big)}{(-u_c p'^c)^{k}} \left(\prod_{a_k \in A_k} p'^{a_k} \right) p'^b f_{(\nu')} \big[(k-1) p'^c \boldsymbol{\nabla}_b u_c\big], \tag{D.3}$$

and

$$\Big[\boldsymbol{\nabla}_b M^{A_k b}\Big]_3 = \int \mathrm{d}V_{p'} \, \frac{\delta\big(\nu + u_c p'^c\big)}{(-u_c p'^c)^{k-1}} \left(\prod_{a_k \in A_k} p'^{a_k} \right) p'^b \boldsymbol{\nabla}_b f_{(\nu')}. \tag{D.4}$$

The goal is to rewrite each of these in terms of various ranks of the moment-expansion in Eq. (D.1). For the first term, integrating by parts allows gives allows writing Eq. (D.2) as

$$\begin{aligned} &- \boldsymbol{\nabla}_b u_c \int \mathrm{d}V'_p \left(\prod_{a_k \in A_k} p'^{a_k} \right) \frac{p'^b p'^c f_{(\nu')}}{(-u_c p'^c)^k} \left[\big(-u_c p'^c\big) \frac{\partial\big[\delta\big(\nu + u_c p'^c\big)\big]}{\partial(-u_c p'^c)} \right] \\ = \; &- \boldsymbol{\nabla}_b u_c \int \mathrm{d}V'_p \left(\prod_{a_k \in A_k} p'^{a_k} \right) \frac{p'^b p'^c f_{(\nu')}}{(-u_c p'^c)^k} \left[\frac{\partial\big[\big(-u_c p'^c\big)\delta\big(\nu + u_c p'^c\big)\big]}{\partial(-u_c p'^c)} \right], \end{aligned} \tag{D.5}$$

where the additional derivative term of the form $\partial\big(-u_c p'^c\big)/\partial\big(-u_c p'^c\big) \to 0$ under integration is suppressed in the final result. Next, applying the δ-function identity $\partial_y[y\delta(x-y)] = -\partial_x[x\delta(x-y)]$ allows further simplification of Eq. (D.5) into the form

$$\begin{aligned} &\boldsymbol{\nabla}_b u_c \int \mathrm{d}V'_p \left(\prod_{a_k \in A_k} p'^{a_k} \right) \frac{p'^b p'^c f_{(\nu')}}{(-u_c p'^c)^k} \left[\frac{\partial\big[\nu\delta\big(\nu + u_c p'^c\big)\big]}{\partial\nu} \right] \\ &= \frac{\partial}{\partial\nu} \left[\nu \boldsymbol{\nabla}_b u_c \int \mathrm{d}V'_p \left(\prod_{a_k \in A_k} p'^{a_k} \right) p'^b p'^c \frac{\delta\big(\nu + u_c p'^c\big) f_{(\nu')}}{(-u_c p'^c)^k} \right] \\ &= \frac{\partial}{\partial\nu}\Big(\nu M^{A_k bc} \boldsymbol{\nabla}_b u_c\Big). \end{aligned} \tag{D.6}$$

For the next term, pulling momentum-independent quantities outside of the integral allows Eq. (D.3) to be written as

$$(k-1)\boldsymbol{\nabla}_b u_c \int \mathrm{d}V_{p'} \, \frac{\delta\big(\nu + u_c p'^c\big)}{(-u_c p'^c)^k} \left(\prod_{a_k \in A_k} p'^{a_k} \right) p'^b p'^c f_{(\nu')} = (k-1) M^{A_k bc} \boldsymbol{\nabla}_b u_c. \tag{D.7}$$

The final term involves the covariant derivative of the distribution function, which reduces to a simple partial derivative since $f_{(\nu')}$ is a scalar function. As the moment expansion is valid only for specific values of the frequency ν, using the Boltzmann equation in Eq. (3.6) at a fixed momentum-coordinate, Eq. (D.4) can be further simplified as

$$\begin{aligned}
&\int \mathrm{d}V_{p'} \frac{\delta\big(\nu + u_c p'^c\big)}{(-u_c p'^c)^{k-1}} \left(\prod_{a_k \in A_k} p'^{a_k} \right) p'^b \frac{\partial f_{(\nu')}}{\partial x^b} \\
&= \int \mathrm{d}V_{p'} \frac{\delta\big(\nu + u_c p'^c\big)}{(-u_c p'^c)^{k-1}} \left(\prod_{a_k \in A_k} p'^{a_k} \right) p'^b \frac{\big(-u_c p'^c\big)}{p'^b} \left[\frac{\mathrm{d}f}{\mathrm{d}t} \right]_{\text{coll.}} \\
&= \int \mathrm{d}V_{p'} \frac{\delta\big(\nu + u_c p'^c\big)}{(-u_c p'^c)^{k-2}} \left(\prod_{a_k \in A_k} p'^{a_k} \right) \left[\frac{\mathrm{d}f}{\mathrm{d}t} \right]_{\text{coll.}} \\
&= S^{A_k}. \qquad \text{(D.8)}
\end{aligned}$$

Collecting these results together shows that the evolution equation for an arbitrary-rank of the moment expansion is

$$\begin{aligned}
\boldsymbol{\nabla}_b M^{A_k b} &= \Big[\boldsymbol{\nabla}_b M^{A_k b}\Big]_1 + \Big[\boldsymbol{\nabla}_b M^{A_k b}\Big]_2 + \Big[\boldsymbol{\nabla}_b M^{A_k b}\Big]_3 \\
\rightarrow \boldsymbol{\nabla}_b M^{A_k b} &- \frac{\partial}{\partial \nu}\Big(\nu M^{A_k bc} \boldsymbol{\nabla}_b u_c\Big) - (k-1) M^{A_k bc} \boldsymbol{\nabla}_b u_c = S^{A_k}. \qquad \text{(D.9)}
\end{aligned}$$

D.2. Eulerian Projection of the Second-Rank Moment

The radiation energy and momentum density equations, Eqns. (3.39)–(3.40), are obtained from taking the Eulerian projection of the second-rank moment evolution in Eq. (3.21). The radiation energy density equation results from projecting Eq. (3.21) along the Eulerian

observer's four-velocity n^a

$$n_a \boldsymbol{\nabla}_b M^{ab} - \frac{\partial}{\partial \nu}\Big(\nu n_a M^{abc} \boldsymbol{\nabla}_b uc\Big) = n_a S^a. \tag{D.10}$$

The final two terms in Eq. (D.10) are already in the form of those in the evolution equations, so the first step is to expand the covariant divergence in the first term as

$$n_a \boldsymbol{\nabla}_b M^{ab} = \boldsymbol{\nabla}_b\Big(n_a M^{ab}\Big) - M^{ab} \boldsymbol{\nabla}_b n_a. \tag{D.11}$$

Taking the projection of Eq. (3.32) along n^a

$$n_a M^{ab} = -E n^b - F^b, \tag{D.12}$$

and using Eq. (C.11), the first term in Eq. (D.11) becomes

$$\begin{aligned}
\boldsymbol{\nabla}_b\Big(n_a M^{ab}\Big) &= -\boldsymbol{\nabla}_b\Big(E n^b + F^b\Big) \\
&= -\frac{1}{\alpha\sqrt{\gamma}} \frac{\partial}{\partial x^b}\Big[\alpha\sqrt{\gamma}\Big(E n^b + F^b\Big)\Big] \\
&= -\frac{1}{\alpha\sqrt{\gamma}} \frac{\partial}{\partial t}[\sqrt{\gamma} E] - -\frac{1}{\alpha\sqrt{\gamma}} \frac{\partial}{\partial x^j}\Big[\sqrt{\gamma}\Big(\alpha F^j - \beta^j E\Big)\Big].
\end{aligned} \tag{D.13}$$

Making use of Eq. (C.7), the second term in Eq. (D.11) is

$$\begin{aligned}
-M^{ab} \boldsymbol{\nabla}_b n_a &= \Big(E n^a n^b + F^a n^b + F^b n^a + P^{ab}\Big)\left(K_{ab} + \frac{\partial \ln \alpha}{\partial x^a}\right) \\
&= P^{ij} K_{ij} - F^a \frac{\partial \ln \alpha}{\partial x^a}.
\end{aligned} \tag{D.14}$$

Combining Eqns. (D.13)–(D.14) in Eq. (D.11) and using the result for the first term of Eq. (D.10) yields the radiation energy density evolution equation

$$\frac{\partial \tilde{E}}{\partial t} + \frac{\partial}{\partial x^j}\Big(\alpha \tilde{F}^j - \tilde{E}\beta^j\Big) + \frac{\partial}{\partial \nu}\Big(\nu \alpha n_a \tilde{M}^{abc} \boldsymbol{\nabla}_b u_c\Big) = \alpha\bigg(\tilde{P}^{ij} K_{ij} - \tilde{F}^i \frac{\partial \ln \alpha}{\partial x^i} - n_a \tilde{S}^a\bigg), \tag{D.15}$$

where the tilde-quantities are once again the densitized moments, e.g. $\tilde{E} = \sqrt{\gamma} E$.

The radiation momentum density evolution equation results from the spacelike projection of Eq. (3.21)

$$\gamma_{ia} \boldsymbol{\nabla}_b M^{ab} - \frac{\partial}{\partial \nu}\Big(\nu \gamma_{ia} M^{abc} \boldsymbol{\nabla}_b u_c\Big) = \gamma_{ia} S^a. \tag{D.16}$$

In a similar manner as the timelike projections, the first term in Eq. (D.16) can be written as

$$\gamma_{ia} \boldsymbol{\nabla}_b M^{ab} = \boldsymbol{\nabla}_b\Big(\gamma_{ia} M^{ab}\Big) - M^{ab} \boldsymbol{\nabla}_b \gamma_{ia}. \tag{D.17}$$

Taking the spacelike projection of Eq. (3.32)

$$\gamma_{ia} M^{ab} = F_i n^b + P^b_{\ i}\,, \tag{D.18}$$

and using Eq. (C.11), the first term of Eq. (D.17) becomes

$$\begin{aligned} \boldsymbol{\nabla}_b\Big(\gamma_{ia} M^{ab}\Big) &= \boldsymbol{\nabla}_b\Big(F_i n^b + P^b_{\ i}\Big) \\ &= \frac{1}{\alpha\sqrt{\gamma}} \frac{\partial}{\partial x^b}\Big[\alpha\sqrt{\gamma}\Big(F_i n^b + P^b_{\ i}\Big)\Big] - \Gamma^c_{\ ib} F_c n^b - \Gamma^c_{\ ib} P^b_{\ c}\,. \end{aligned} \tag{D.19}$$

The first term of Eq. (D.19) is then

$$\frac{1}{\alpha\sqrt{\gamma}}\frac{\partial}{\partial x^b}\Big[\alpha\sqrt{\gamma}\Big(F_i n^b + P^b{}_i\Big)\Big] = \frac{1}{\alpha\sqrt{\gamma}}\frac{\partial}{\partial t}[\sqrt{\gamma}F_k] + \frac{1}{\alpha\sqrt{\gamma}}\frac{\partial}{\partial x^j}\Big[\sqrt{\gamma}\Big(\alpha P^j{}_i - F_i\beta^j\Big)\Big]. \quad \text{(D.20)}$$

Using Eq. (C.3), Eq. (C.7), and Eq. (C.10), the second term of Eq. (D.19) becomes

$$\begin{aligned} -\Gamma^c{}_{ib}F_c n^b &= -F_c\Gamma^c{}_{bi}n^b \\ &= -F_c\left(g^{cd}\boldsymbol{\nabla}_i n_d - \frac{\partial n^c}{\partial x^i}\right) \\ &= -F^d\boldsymbol{\nabla}_i n_d - \frac{F_j}{\alpha}\frac{\partial\beta^j}{\partial x^i} \\ &= F^j K_{ij} - \frac{F_j}{\alpha}\frac{\partial\beta^j}{\partial x^i}. \end{aligned} \quad \text{(D.21)}$$

Using Eq. (C.12) and noting that $g_{ij} = \gamma_{ij}$, the third term in Eq. (D.19) becomes

$$\begin{aligned} -\Gamma^c{}_{ib}P^b{}_c &= -\frac{1}{2}P^b{}_c\, g^{cd}\left(\frac{\partial g_{bd}}{\partial x^i} + \frac{\partial g_{id}}{\partial x^b} - \frac{\partial g_{ib}}{\partial x^d}\right) \\ &= -\frac{1}{2}P^{jk}\left(\frac{\partial\gamma_{jk}}{\partial x^i} + \frac{\partial\gamma_{ik}}{\partial x^j} - \frac{\partial\gamma_{ij}}{\partial x^k}\right) \\ &= -\frac{1}{2}P^{jk}\frac{\partial\gamma_{jk}}{\partial x^i}, \end{aligned} \quad \text{(D.22)}$$

where the final two terms cancels under contraction due the symmetry of P^{ij}. Finally, using Eqns. (C.6)–(C.7) and Eq. (3.32), the second term in Eq. (D.17) becomes

$$\begin{aligned} -M^{ab}\boldsymbol{\nabla}_b\gamma_{ia} &= -\Big(En^a n^b + F^a n^b + F^b n^a + P^{ab}\Big)(\boldsymbol{\nabla}_b g_{ia} + n_i\boldsymbol{\nabla}_b n_a + n_a\boldsymbol{\nabla}_b n_i) \\ &= En^b\boldsymbol{\nabla}_b n_i + F^b\boldsymbol{\nabla}_b n_i \\ &= E\frac{\partial\ln\alpha}{\partial x^i} - F^j K_{ij}. \end{aligned} \quad \text{(D.23)}$$

In total, using Eqns. (D.17)–(D.23) in Eq. (D.16) gives the final form of the momentum density equation

$$\frac{\partial}{\partial \tilde{F}_i} + \frac{\partial}{\partial x^j}\Big(\alpha \tilde{P}^j{}_i - \tilde{F}_i \beta^j\Big) - \frac{\partial}{\partial \nu}\Big(\nu \alpha \gamma_{ia} \tilde{M}^{abc} \boldsymbol{\nabla}_b u_c\Big) = \frac{\alpha}{2}\tilde{P}^{jk}\frac{\partial \gamma_{jk}}{\partial x^i} + F_j \frac{\partial \beta^j}{\partial x^i} - E\frac{\partial \alpha}{\partial x^i} + \gamma_{ia} S^a. \tag{D.24}$$

D.3. Eulerian Projection of Frequency-Space Advection

The core of the frequency-space advection term involves the contraction of the third-rank moment expansion of the distribution function ($k = 2$ in Eq. (3.7)) and the covariant derivative of the co-moving observer's four-velocity. While both of these quantities are most easily defined in the co-moving frame, it is much easier to evaluate this term using the Eulerian projection. Making use of Eq. (3.68), the resulting contraction takes the form

$$\begin{aligned} M^{abc}\boldsymbol{\nabla}_b u_c = \Big[& Q n^a n^b n^c + R^a n^b n^c + R^b n^a n^c + R^c n^a n^b \\ & + S^{ab} n^c + S^{ac} n^b + S^{bc} n^a + T^{abc}\Big] \\ & \times \left[W \boldsymbol{\nabla}_b n_c + n_c \frac{\partial W}{\partial x^b} + \boldsymbol{\nabla}_b (W v_c)\right]. \end{aligned} \tag{D.25}$$

Using Eqns. (C.5)–(C.6) and Eq. (C.9), the first term of Eq. (D.25) becomes

$$\begin{aligned} Q n^a n^b n^c \left[W \boldsymbol{\nabla}_b n_c + n_c \frac{\partial W}{\partial x^b} + \boldsymbol{\nabla}_b (W v_c)\right] &= Q n^a \left[-n^b \frac{\partial W}{\partial x^b} + n^b n^c \boldsymbol{\nabla}_b W v_c \right] \\ &= Q n^a \left[-n^b \frac{\partial W}{\partial x^b} - W n^b v^c \boldsymbol{\nabla}_b n_c \right] \\ &= -Q n^a \left[n^b \frac{\partial W}{\partial x^b} + W v^k \frac{\partial \ln \alpha}{\partial x^k} \right], \end{aligned} \tag{D.26}$$

from which the second term of Eq. (D.25) can be found in the same exact manner as

$$R^a n^b n^c \left[W \boldsymbol{\nabla}_b n_c + n_c \frac{\partial W}{\partial x^b} + \boldsymbol{\nabla}_b (W v_c) \right] = -R^a \left[n^b \frac{\partial W}{\partial x^b} + W v^k \frac{\partial \ln \alpha}{\partial x^k} \right]. \tag{D.27}$$

Using Eq. (C.5), Eq. (C.7) and Eq. (C.9), the third term of Eq. (D.25) becomes

$$\begin{aligned} R^b n^a n^c \left[W \boldsymbol{\nabla}_b n_c + n_c \frac{\partial W}{\partial x^b} + \boldsymbol{\nabla}_b (W v_c) \right] &= n^a R^i \left[-\frac{\partial W}{\partial x^i} + n^c \boldsymbol{\nabla}_i (W v_c) \right] \\ &= -n^a R^i \left[\frac{\partial W}{\partial x^i} + W v^j \boldsymbol{\nabla}_i (n_j) \right] \\ &= -n^a R^i \left[\frac{\partial W}{\partial x^i} - W v^j K_{ij} \right], \end{aligned} \tag{D.28}$$

from which the fifth term in Eq. (D.25) follows as

$$S^{ab} n^c \left[W \boldsymbol{\nabla}_b n_c + n_c \frac{\partial W}{\partial x^b} + \boldsymbol{\nabla}_b (W v_c) \right] = -S^{ai} \left[\frac{\partial W}{\partial x^i} - W v^j K_{ij} \right]. \tag{D.29}$$

Using Eqns. (C.5)–(C.6) and Eqns. (C.9)–(C.10), the fourth term in Eq. (D.25) is

$$\begin{aligned} & R^c n^a n^b \left[W \boldsymbol{\nabla}_b n_c + n_c \frac{\partial W}{\partial x^b} + \boldsymbol{\nabla}_b (W v_c) \right] \\ &= n^a R^i \left[W \frac{\partial \ln \alpha}{\partial x^i} + g_{ic} n^b \boldsymbol{\nabla}_n (W v^c) \right] \\ &= n^a R^i \left[W \frac{\partial \ln \alpha}{\partial x^i} + \gamma_{ij} n^b \frac{\partial W v^j}{\partial x^b} + g_{ic} W v^d \Gamma^c{}_{db} n^b \right] \\ &= n^a R^i \left[W \frac{\partial \ln \alpha}{\partial x^i} + \gamma_{ij} n^b \frac{\partial W v^j}{\partial x^b} + g_{ic} W v^d \left(\boldsymbol{\nabla}_d n^c - \frac{\partial n^c}{\partial x^d} \right) \right] \\ &= n^a R^i \left[W \frac{\partial \ln \alpha}{\partial x^i} + \gamma_{ij} n^b \frac{\partial W v^j}{\partial x^b} - W v^j K_{ij} + \gamma_{ij} \frac{W v^k}{\alpha} \frac{\partial \beta^j}{\partial x^k} \right], \end{aligned} \tag{D.30}$$

from which the sixth term in Eq. (D.25) can be found as

$$S^{ac}n^b\left[W\boldsymbol{\nabla}_b n_c + n_c\frac{\partial W}{\partial x^b} + \boldsymbol{\nabla}_b(Wv_c)\right] = S^{ai}\left[W\frac{\partial \ln\alpha}{\partial x^i} + \gamma_{ij}n^b\frac{\partial Wv^j}{\partial x^b} - Wv^jK_{ij} + \gamma_{ij}\frac{Wv^k}{\alpha}\frac{\partial\beta^j}{\partial x^k}\right]. \tag{D.31}$$

Using Eq. (C.7) and Eq. (C.12), the seventh term in Eq. (D.25) is

$$\begin{aligned} S^{bc}n^a&\left[W\boldsymbol{\nabla}_b n_c + n_c\frac{\partial W}{\partial x^b} + \boldsymbol{\nabla}_b(Wv_c)\right] \\ &= n^aS^{ij}\left[W\boldsymbol{\nabla}_j n_i + \gamma_{ik}\boldsymbol{\nabla}_j\Big(Wv^k\Big)\right] \\ &= n^aS^{ij}\left[-WK_{ij} + \gamma_{ik}\frac{\partial Wv^k}{\partial x^j} + \frac{Wv^k}{2}\left(\frac{\partial\gamma_{i}j}{\partial x^k} + \frac{\partial\gamma_{ik}}{\partial x^j} - \frac{\partial\gamma_{jk}}{\partial x^i}\right)\right] \\ &= n^aS^{ij}\left[-WK_{ij} + \gamma_{ik}\frac{\partial Wv^k}{\partial x^j} + \frac{Wv^k}{2}\frac{\partial\gamma_{ij}}{\partial x^k}\right], \end{aligned} \tag{D.32}$$

where the last line follows from $S^{ij} = S^{ji}$. Finally, the last term in Eq. (D.25) can be found in the same manner as Eq. (D.32) as

$$T^{abc}\left[W\boldsymbol{\nabla}_b n_c + n_c\frac{\partial W}{\partial x^b} + \boldsymbol{\nabla}_b(Wv_c)\right] = T^{aij}\left[-WK_{ij} + \gamma_{ik}\frac{\partial Wv^k}{\partial x^j} + \frac{Wv^k}{2}\frac{\partial\gamma_{ij}}{\partial x^k}\right]. \tag{D.33}$$

In full, using Eqns. (D.26)–(D.33) in Eq. (D.25) give

$$\begin{aligned} M^{abc}\boldsymbol{\nabla}_b u_c = &-\left[n^b\frac{\partial W}{\partial x^b} + Wv^k\frac{\partial\ln\alpha}{\partial x^k}\right](n^aQ + R^a) \\ &- \left[\frac{\partial W}{\partial x^i} - W\frac{\partial\ln\alpha}{\partial x^i} - \gamma_{ij}n^b\frac{\partial Wv^j}{\partial x^b} - \gamma_{ij}\frac{Wv^k}{\alpha}\frac{\partial\beta^j}{\partial x^k}\right]\Big(n^aR^i + S^{ai}\Big) \\ &- \left[WK_{ij} - \gamma_{ik}\frac{\partial Wv^k}{\partial x^j} - \frac{Wv^k}{2}\frac{\partial\gamma_{ij}}{\partial x^k}\right]\Big(n^aS^{ij} + T^{aij}\Big). \end{aligned} \tag{D.34}$$

APPENDIX E. Practical Expressions for the Fluid Frame Projections

Many calculations associated with solving the projected evolution equations require expressing the co-moving frame projections for the energy density J and momentum density H^a in terms of the evolved Eulerian frame projections E and F^a. While these quantities can be interpolated via the closure relation in similar manner as the pressure tensor, a more efficient approach is to collect the terms common to both limiting forms in addition to those specific to each limit. This appendix will present the expressions for these terms in forms aptly suited for numerical computation, using a notation similar to the ones in Radice et al. (2022); Cheong et al. (2023).

To begin, the second-rank moment expansion in the Eulerian frame is written in terms of the closure-interpolated pressure tensor as

$$\begin{aligned} M^{ab} &= En^a n^b + F^a n^b + F^b n^a + d_{\rm thin} P^{ab}_{\rm thin} + d_{\rm thick} P^{ab}_{\rm thick} \\ &= M^{ab}_0 + d_{\rm thin} M^{ab}_{\rm thin} + d_{\rm thick} M^{ab}_{\rm thick} \end{aligned} \tag{E.1}$$

where each M^{ab} represents the common terms (zero-subscript) and optically thin and thick terms[1], and the $d_{\rm thin}$ and $d_{\rm thick}$ are the interpolation coefficients

$$d_{\rm thin} = \frac{3\chi(\xi) - 1}{2}, \quad d_{\rm thick} = \frac{3[1 - \chi(\xi)]}{2}, \tag{E.2}$$

such that $d_{\rm thin} + d_{\rm thick} = 1$. Projecting Eq. (E.1) via Eqns. (3.27)–(3.28) allows writing the

[1] The optically thick term in Eq. (E.1) is different than the similarly-named limiting form of the second-rank moment used previously in Eq. (3.60), and only represents the terms present in the optically thick limit, i.e. $M^{ab}_{\rm thick} = P^{ab}_{\rm thick}$.

co-moving frame projections in the form

$$J = B_0 + d_{\text{thin}} B_{\text{thin}} + d_{\text{thick}} B_{\text{thick}} \tag{E.3}$$

$$H^a = A^a_0 + d_{\text{thin}} A^a_{\text{thin}} + d_{\text{thick}} A^a_{\text{thick}}. \tag{E.4}$$

For the energy density, the projected coefficients follow from Eq. (3.27) as

$$B_0 = W^2\Big(E - 2v_i F^i\Big) \tag{E.5}$$

$$B_{\text{thin}} = W^2 E\Big(v_i \hat{f}^i\Big)^2 \tag{E.6}$$

$$B_{\text{thick}} = \frac{W^2 - 1}{2W^2 + 1}\Big[\Big(3 - 2W^2\Big)E + 4W^2 v_i F^i\Big]. \tag{E.7}$$

The momentum density is obtained similarly, but it is useful to separate the terms further relative to the Eulerian four-velocity n^a, spatial velocity v^a, and the Eulerian frame momentum density F^a. In this form, projecting each term in Eq. (E.1) via Eq. (3.28) gives

$$A^a_0 = A^n_0 n^a + A^v_0 v^a + A^F_0 F^a \tag{E.8}$$

$$A^a_{\text{thin}} = A^n_{\text{thin}} n^a + A^v_{\text{thin}} v^a + A^F_{\text{thin}} F^a \tag{E.9}$$

$$A^a_{\text{thick}} = A^n_{\text{thick}} n^a + A^v_{\text{thick}} v^a + A^F_{\text{thick}} F^a, \tag{E.10}$$

For the common terms shared by both limits, the coefficients are

$$A^n_0 = -W\Big(B - E + v_k F^k\Big) \tag{E.11}$$

$$A^v_0 = -WB \tag{E.12}$$

$$A^F_0 = W. \tag{E.13}$$

For the optically thin terms, the coefficients are

$$A^n_{\text{thin}} = -WB_{\text{thin}} \tag{E.14}$$

$$A^v_{\text{thin}} = -WB_{\text{thin}} \tag{E.15}$$

$$A^F_{\text{thin}} = -W\frac{E}{F}v_k\hat{f}^k. \tag{E.16}$$

Finally, the optically thick coefficients are

$$A^n_{\text{thick}} = -WB_{\text{thick}} \tag{E.17}$$

$$A^v_{\text{thick}} = -W\left(B_{\text{thick}} + \frac{3-2W^2}{2W^2+1}E + \frac{2W^2-1}{2W^2+1}v_kF^k\right) \tag{E.18}$$

$$A^F_{\text{thick}} = -Wv^2, \tag{E.19}$$

where $v^2 = \gamma_{ij}v^iv^j$.

The higher-rank projected co-moving frame moments follow directly from performing the closure interpolation between the optically thin and thick limiting forms. For the second- and third-rank projections, these interpolations take the form

$$L^{ab} = d_{\text{thin}}L^{ab}_{\text{thin}} + d_{\text{thick}}L^{ab}_{\text{thick}} \tag{E.20}$$

$$N^{abc} = d_{\text{thin}}N^{abc}_{\text{thin}} + d_{\text{thick}}N^{abc}_{\text{thick}}. \tag{E.21}$$

APPENDIX F. Neutrino-Matter Interation Source Term Jacobian

For the neutrino-matter interaction source terms in Eq. (3.106), the included interactions do no couple the neutrino species nor frequencies to each other. This leads to a block-diagonal matrix equation for the implicit update of these stiff terms, allowing each block corresponding to a single neutrino species and frequency to be updated independently. In the following, all radiation quantities are assumed to be frequency-dependent, but the explicit ν-subscript will be suppressed for readability. For this case, the jacobian of the interaction source terms required for the implicit update is

$$\frac{\partial \mathbf{S}}{\partial \mathbf{U}} = \alpha \begin{bmatrix} -n_a \dfrac{\partial S^a}{\partial E} & -n_a \dfrac{\partial S^a}{\partial F_j} \\ \dfrac{\partial S_i}{\partial E} & \dfrac{\partial S_i}{\partial F_j} \end{bmatrix}. \tag{F.1}$$

Using Eqns. (3.99)–(3.100), the elements of the jacobian are

$$-n_a \frac{\partial S^a}{\partial E} = -W\kappa_{\text{abs.}} \frac{\partial J}{\partial E} - (\kappa_{\text{abs.}} + \kappa_{\text{iso.}}) n_a \frac{\partial H^a}{\partial E} \tag{F.2}$$

$$-n_a \frac{\partial S^a}{\partial F_j} = -W\kappa_{\text{abs.}} \frac{\partial J}{\partial F_j} - (\kappa_{\text{abs.}} + \kappa_{\text{iso.}}) n_a \frac{\partial H^a}{\partial F_j} \tag{F.3}$$

$$\frac{\partial S_i}{\partial E} = -W\kappa_{\text{abs.}} \frac{\partial J}{\partial E} v_i - (\kappa_{\text{abs.}} + \kappa_{\text{iso.}}) \frac{\partial \bar{H}_i}{\partial E} \tag{F.4}$$

$$\frac{\partial S_i}{\partial F_j} = -W\kappa_{\text{abs.}} \frac{\partial J}{\partial F_j} v_i - (\kappa_{\text{abs.}} + \kappa_{\text{iso.}}) \frac{\partial \bar{H}_i}{\partial F_j}, \tag{F.5}$$

where $\bar{H}_i = \gamma_{ia} H^a$. For each of the co-moving frame quantities, these derivatives can be written in terms of the closure interpolation as

$$\frac{\partial J}{\partial E} = \frac{\partial B}{\partial E} + d_{\text{thin}} \frac{\partial B_{\text{thin}}}{\partial E} + d_{\text{thick}} \frac{\partial B_{\text{thick}}}{\partial E} \tag{F.6}$$

$$\frac{\partial J}{\partial F_j} = \frac{\partial B}{\partial F_j} + d_{\text{thin}} \frac{\partial B_{\text{thin}}}{\partial F_j} + d_{\text{thick}} \frac{\partial B_{\text{thick}}}{\partial F_j} \tag{F.7}$$

and

$$n_a \frac{\partial H^a}{\partial E} = n_a \frac{\partial A^a}{\partial E} + d_{\text{thin}} \left(n_a \frac{\partial A^a_{\text{thin}}}{\partial E} \right) + d_{\text{thick}} \left(n_a \frac{\partial A^a_{\text{thick}}}{\partial E} \right) \tag{F.8}$$

$$n_a \frac{\partial H^a}{\partial F_j} = n_a \frac{\partial A^a}{\partial F_j} + d_{\text{thin}} \left(n_a \frac{\partial A^a_{\text{thin}}}{\partial F_j} \right) + d_{\text{thick}} \left(n_a \frac{\partial A^a_{\text{thick}}}{\partial F_j} \right) \tag{F.9}$$

$$\frac{\partial \bar{H}_i}{\partial E} = \frac{\partial \bar{A}_i}{\partial E} + d_{\text{thin}} \left(\frac{\partial \bar{A}_{i,\text{thin}}}{\partial E} \right) + d_{\text{thick}} \left(\frac{\partial \bar{A}_{i,\text{thick}}}{\partial E} \right) \tag{F.10}$$

$$\frac{\partial \bar{H}_i}{\partial F_j} = \frac{\partial \bar{A}_i}{\partial F_j} + d_{\text{thin}} \left(\frac{\partial \bar{A}_{i,\text{thin}}}{\partial F_j} \right) + d_{\text{thick}} \left(\frac{\partial \bar{A}_{i,\text{thick}}}{\partial F_j} \right). \tag{F.11}$$

Using Eqns. (E.5)–(E.7), the terms in the derivatives to the co-moving frame energy density are

$$\frac{\partial B}{\partial E} = W^2 \tag{F.12}$$

$$\frac{\partial B_{\text{thin}}}{\partial E} = W^2 \left(v_k \hat{f}^k \right)^2 \tag{F.13}$$

$$\frac{\partial B_{\text{thick}}}{\partial E} = \frac{\left(W^2 - 1 \right) \left(3 - 2W^2 \right)}{2W^2 + 1} \tag{F.14}$$

and

$$\frac{\partial B}{\partial F_j} = -2W^2 v^j \tag{F.15}$$

$$\frac{\partial B_{\text{thin}}}{\partial F_j} = \frac{2W^2 E v_k \hat{f}^k}{F} \left[v^j - \left(v_k \hat{f}^k \right) \hat{f}^j \right] \tag{F.16}$$

$$\frac{\partial B_{\text{thick}}}{\partial F_j} = \frac{4W^2 \left(W^2 - 1 \right)}{2W^2 + 1} v^j. \tag{F.17}$$

For the co-moving frame momentum density, instead of considering the derivatives of H^a with respect to E and F_j directly, it will be simpler to consider the timelike and spacelike projections separately. Using Eqns. (E.8)–(E.10), the derivatives of the timelike projections

are

$$n_a \frac{\partial A^a}{\partial E} = W\left(\frac{\partial B}{\partial E} - 1\right) \tag{F.18}$$

$$n_a \frac{\partial A^a_{\text{thin}}}{\partial E} = W\frac{\partial B_{\text{thin}}}{\partial E} \tag{F.19}$$

$$n_a \frac{\partial A^a_{\text{thick}}}{\partial E} = W\frac{\partial B_{\text{thick}}}{\partial E} \tag{F.20}$$

$$n_a \frac{\partial A^a}{\partial F_j} = W\left(\frac{\partial B}{\partial F_j} + v^j\right) \tag{F.21}$$

$$n_a \frac{\partial A^a_{\text{thin}}}{\partial F_j} = W\frac{\partial B_{\text{thin}}}{\partial F_j} \tag{F.22}$$

$$n_a \frac{\partial A^a_{\text{thick}}}{\partial F_j} = W\frac{\partial B_{\text{thick}}}{\partial F_j}. \tag{F.23}$$

Similarly, the derivatives of the spacelike projections are

$$\frac{\partial \bar{A}_i}{\partial E} = -W\left(\frac{\partial B}{\partial E}\right)v_i \tag{F.24}$$

$$\frac{\partial \bar{A}_{i,\text{thin}}}{\partial E} = -W\left[\left(\frac{\partial B_{\text{thin}}}{\partial E}\right)v_i + \left(v_k\hat{f}^k\right)\hat{f}_i\right] \tag{F.25}$$

$$\frac{\partial \bar{A}_{i,\text{thick}}}{\partial E} = -W\left[\left(\frac{\partial B_{\text{thick}}}{\partial E}\right) + \frac{3-2W^2}{2W^2+1}\right]v_i \tag{F.26}$$

and

$$\frac{\partial \bar{A}_i}{\partial F_j} = -W\left[\left(\frac{\partial B}{\partial F_j}\right)v_i + \gamma^j{}_i\right] \tag{F.27}$$

$$\frac{\partial \bar{A}_{i,\text{thin}}}{\partial F_j} = -W\left[\left(\frac{\partial B_{\text{thin}}}{\partial F_j}\right)v_i + \left(v^j - 2v_k\hat{f}^k\hat{f}^j\right)\hat{f}_i + v_k\hat{f}^k\gamma^j{}_i\right] \tag{F.28}$$

$$\frac{\partial \bar{A}_{i,\text{thick}}}{\partial E} = -W\left[\left(\frac{\partial B_{\text{thick}}}{\partial F_j}\right)v_i + \frac{2W^2-1}{2W^2+1}v_iv^j + \frac{W^2-1}{W^2}\gamma^j{}_i\right] \tag{F.29}$$

APPENDIX G. Time-Integration Tableau

The method-of-lines (MoL) time-integrators in `Flash-X` can easily be extended with new Butcher tableau. Each of the three types of integrators, explicit Runge-Kutta (ERK), implicit-explicit (IMEX), and multi-rate (MR), contain a `tableau` directory. To add a new method, a new subdirectory with the name of the method must be created in this directory, e.g., `tableau/rk4` for the classic fourth-order Runge-Kutta method.

```
# tableau/rk4/Config

# Number of intermediate states to store
NRHS 4

# Pre-processor defines for order and number of stages
PPDEFINE MOL_ORDER 4
PPDEFINE MOL_NSTAGES 4
```

Code Snippet G.1: Example `Config` file for the RK4 integration method.

```
subroutine ta_molERKInitTableau()
   use ta_molERKData, only: ta_molERK_tableau, ta_molERK_A,
                            ta_molERK_b, ta_molERK_c
   implicit none

   ! Name of the method
   ta_molERK_tableau = "rk4"

   ! Set the values of the matrix A
   ta_molERK_A = 0.0
   ta_molERK_A(2, 1) = 1.0/2.0
   ta_molERK_A(3, 2) = 1.0/2.0
   ta_molERK_A(4, 3) = 1.0

   ! Set the values of the b and c vectors
   ta_molERK_b(:) = [1.0/6.0, 1.0/3.0, 1.0/3.0, 1.0/6.0]
   ta_molERK_c(:) = [0.0, 1.0/2.0, 1.0/2.0, 1.0]
end subroutine ta_molERKInitTableau
```

Code Snippet G.2: Example subroutine for the ERK RK4 integration method.

This subdirectory must contain a `Flash-X Config` file that defines the number of stages and the order of the method, as well as the required number of intermediate states to be stored; see code snippet G.1 for an example `Config` file. Each method must also provide an implementation for the integrator-specific subroutine that sets the values in the tableau. An example for the RK4 method in the ERK integrator is provided in code snippet G.2.

The remainder of this appendix list the currently available methods in the `Flash-X` MoL ERK, IMEX, and MR time-integrators.

G.1. Explicit Methods

0	0
	1

Table G.1: First-order single-stage forward Euler method (LeVeque, 2007)

0	0	0
1	1	0
	1/2	1/2

Table G.2: Second-order two-stage Heun's SSP RK2 method (Shu & Osher, 1988)

0	0	0	0
1	1	0	0
1/2	1/4	1/4	0
	1/6	1/6	2/3

Table G.3: Third-order three-stage SSP RK3 method (Shu & Osher, 1988)

0	0	0	0	0
1/2	1/2	0	0	0
1/2	0	1/2	0	0
1	0	0	1	0
	1/6	1/3	1/3	1/6

Table G.4: Fourth-order four-stage classic RK4 method (LeVeque, 2007)

G.2. Implicit-Explicit Methods

All methods will be designated by the triplet (s, σ, p), where s is the number of implicit stages, σ is the number of explicit stages, and p is the order of the method. In each tableau presented below, the implicit scheme will be on the left, and the explicit scheme will be on the right.

0	0	0	0	0
1	0	1	1	0
	0	1	1	0

Table G.5: ARK(1,1,1) from Ascher et al. (1997)

0	0	0	0	0
1	0	1	1	0
	0	1	0	1

Table G.6: ARK(1,2,1) from Ascher et al. (1997)

0	0	0	0	0
1/2	0	1/2	1/2	0
	0	1	0	1

Table G.7: ARK(1,2,2) from Ascher et al. (1997)

0	0	0	0	0	0	0
γ	0	γ	0	γ	0	0
1	0	$1-\gamma$	γ	δ	$1-\delta$	0
	0	$1-\gamma$	γ	δ	$1-\delta$	0

Table G.8: ARK(2,2,2) from Ascher et al. (1997); $\gamma = \left(2-\sqrt{2}\right)/2$ and $\delta = 1 - 1/(2\gamma)$

0	0	0	0	0	0	0
γ	0	γ	0	γ	0	0
1	0	$1-\gamma$	γ	δ	$1-\delta$	0
	0	$1-\gamma$	γ	0	$1-\gamma$	γ

Table G.9: ARK(2,3,2) from Ascher et al. (1997); $\gamma = \left(2-\sqrt{2}\right)/2$ and $\delta = -2\sqrt{2}/3$

0	0	0	0	0	0	0
γ	0	γ	0	γ	0	0
$1-\gamma$	0	$1-2\gamma$	γ	$\gamma-1$	$2(1-\gamma)$	0
	0	1/2	1/2	0	1/2	1/2

Table G.10: ARK(2,3,3) from Ascher et al. (1997); $\gamma = \left(3+\sqrt{3}\right)/6$

0	0	0	0	0	0	0	0	0
η	0	η	0	0	η	0	0	0
$\frac{1+\eta}{2}$	0	$\frac{1-\eta}{2}$	η	0	$a_{3,1}$	$a_{3,2}$	0	0
1	0	b_2	b_3	η	$1-2\alpha$	α	α	0
	0	b_2	b_3	η	0	b_2	b_3	η

Table G.11: ARK(3,4,3) from Ascher et al. (1997); Chinomona & Reynolds (2021); see Eqns. (G.1)–(G.5) for the coefficients.

The coefficients in table G.11 are

$$\eta = 0.435866521508458999416019451193556842529 3, \tag{G.1}$$

$$\alpha = 0.552929148035939819361188729738592476494 9, \tag{G.2}$$

$$a_{3,1} = \alpha\left(\frac{15}{4} - 15\eta + \frac{21}{4}\eta^2\right) - \frac{7}{2} + 13\eta - \frac{9}{2}\eta^2, \tag{G.3}$$

$$a_{3,2} = \alpha\left(-\frac{15}{4} + 15\eta - \frac{21}{4}\eta^2\right) + 4 - \frac{25}{2}\eta + \frac{9}{2}\eta^2, \tag{G.4}$$

$$b_2 = -\frac{3}{2}\eta^2 + 4\eta - \frac{1}{4}, \quad b_3 = \frac{3}{2}\eta^2 - 5\eta + \frac{5}{4}. \tag{G.5}$$

0	0	0	0	0	0	0	0	0	0	0
1/2	0	1/2	0	0	0	1/2	0	0	0	0
2/3	0	1/6	1/2	0	0	11/18	1/18	0	0	0
1/2	0	−1/2	1/2	1/2	0	5/6	−5/6	1/2	0	0
1	0	3/2	−3/2	1/2	1/2	1/4	7/4	3/4	−7/4	0
	0	3/2	−3/2	1/2	1/2	1/4	7/4	3/4	−7/4	0

Table G.12: ARK(4,4,3) from Ascher et al. (1997)

G.3. Multi-Rate Methods

Coming soon!