Mon. Not. R. Astron. Soc. **000**, 000–000 (0000) Printed 12 August 2021 (MN LATEX style file v2.2)

Octo-Tiger: a new, 3D hydrodynamic code for stellar mergers that uses HPX parallelisation

Dominic C. Marcello[1,2,*], Sagiv Shiber[1,*], Orsola De Marco[3,4], Juhan Frank[1], Geoffrey C. Clayton[1], Patrick M. Motl[5], Patrick Diehl[2], and Hartmut Kaiser[2]

[1]*Department of Physics and Astronomy, Louisiana State University, Baton Rouge, LA, 70803 USA*
[2]*Center for Computational Technologies, Louisiana State University, Baton Rouge, LA, 70803 USA*
[3]*Department of Physics and Astronomy, Macquarie University, Sydney, NSW 2109, Australia*
[4]*Astronomy, Astrophysics and Astrophotonics Research Centre, Macquarie University, Sydney, NSW 2109, Australia*
[5]*The School of Sciences, Indiana University Kokomo, Kokomo, Indiana 46904, USA*
[*]*The first two authors made equal contributions to this paper.*

Accepted XXX. Received 25 December 2020; in original form ZZZ

ABSTRACT

OCTO-TIGER is an astrophysics code to simulate the evolution of self-gravitating and rotating systems of arbitrary geometry based on the fast multipole method, using adaptive mesh refinement. OCTO-TIGER is currently optimised to simulate the merger of well-resolved stars that can be approximated by barotropic structures, such as white dwarfs or main sequence stars. The gravity solver conserves angular momentum to machine precision, thanks to a "correction" algorithm. This code uses HPX parallelization, allowing the overlap of work and communication and leading to excellent scaling properties, allowing for the computation of large problems in reasonable wall-clock times. In this paper, we investigate the code performance and precision by running benchmarking tests. These include simple problems, such as the Sod shock tube, as well as sophisticated, full, white-dwarf binary simulations. Results are compared to analytic solutions, when known, and to other grid based codes such as FLASH. We also compute the interaction between two white dwarfs from the early mass transfer through to the merger and compare with past simulations of similar systems. We measure OCTO-TIGER's scaling properties up to a core count of $\sim$80 000, showing excellent performance for large problems. Finally, we outline the current and planned areas of development aimed at tackling a number of physical phenomena connected to observations of transients.

Key words: stars: white dwarfs - stars: evolution - binaries: close - hydrodynamics - methods: numerical, analytical

1 INTRODUCTION

Approximately two thirds of observed stars are members of binary or multiple stellar systems (Duquennoy & Mayor 1991). While the evolution of single stars is predictable given the initial mass and composition, the evolution of the components of a binary or multiple system can be dramatically altered if they are close enough to become interacting through mass transfer and mutual irradiation. The most consequential interactions in binary systems occur when the components are so close that one (or both) of the stars fill their Roche lobes and mass is transferred from one star to the other. In some cases, the transfer proceeds steadily for a long time; in others mass and energy are exchanged in a common envelope; while in the most dynamic and interesting cases the interaction leads to a merger giving birth to a single star of unusual properties that cannot be produced by the evolution of a single star. We know that all of these processes do occur in nature yielding observable outcomes such as contact binaries (Smith 1984; Rucinski 2010), helium white dwarfs (WD) (Kippenhahn et al. 1967, 1968), R Coronae Borealis stars (Clayton 2012), Type Iax supernovae (Jha 2017; Solheim 2010), and dwarf novae (Warner 2003; Frank et al. 2002). At least a fraction of all Type Ia supernovae, which are some of the most energetic explosion in the Universe, are likely caused by the Roche-Lobe-overflow induced merger of two WDs (Hillebrandt & Niemeyer 2000). This particular type of supernova is used as a "standard candle" for measuring distances to other galaxies. Because of this, understanding Type Ia supernovae is important for understanding the origins of the universe and its eventual fate.

Common envelope binary interactions (Ivanova et al. 2013) play a crucial role in the formation of close binaries of all types including X-ray binaries (Tauris & van den Heuvel 2014), cataclysmic variables (Meyer & Meyer-Hofmeister 1979; Webbink

1992), close binaries in planetary nebulae (De Marco et al. 2015) and possibly even massive stellar double black hole binaries (Ricker et al. 2019). Clearly, common envelope interactions are close cousins of mergers, the main differences being the final outcomes, which in turn depend on the types of stars involved, their mass ratios, and the energetics of mass loss through unbinding and ejection of the envelope.

Important observational developments in the past decade or so have lent further credence to the theoretical ideas involving binary mergers and given further impetus to the development of accurate and efficient numerical tools for investigating binary mergers and common envelope evolution. For example, in September 2008, the contact binary, V1309 Sco, merged to form a luminous red nova (Tylenda et al. 2011). When the merger occurred, the system increased in brightness by a factor of approximately 10^4. Mason et al. (2010) observed the outburst spectroscopically, confirming it as a luminous red nova. Because the Optical Gravitational Lensing Experiment (OGLE) observed V1309 Sco prior to its merger for six years, this provided the rare opportunity to observe a stellar merger both before and after the event, prompting Tylenda to refer to V1309 Sco as the "Rosetta Stone" of contact-binary mergers. More recently, the Zwicky Transient Factory discovery of an eclipsing double white dwarf binary with an orbital period of 8.8 minutes which is destined to merge and become a hot subdwarf or an R Coronae Borealis star, makes it clear that such progenitor binaries do exist and provides further motivation for investigating stellar mergers and their outcomes (Burdge et al. 2020). Discoveries of close-binary systems have multiplied in number due to surveys such as the Zwicky Transient Factory, and are due to increase exponentially with the arrival on the Vera Rubin Observatory in 2022 (Ivezić et al. 2007).

Ultimately, one would want a high resolution 3D magneto- and radiation-hydrodynamic simulation with nuclear energy production, full chemistry, and resolution down to the dynamical timescale while also being able to follow the system over thermal timescales, but those capabilities are well beyond the horizon at this time. In lieu of such simulations, studies have used all manner of hybrid approaches. For example, 3D smooth particle hydrodynamics (SPH) simulations of merging WD binaries with 1.8 million particles are then mapped into grid codes to model SN detonations (Pakmor et al. 2012) and the output is then used to study the light and polarisation signals (Bulla et al. 2016). In a different simulation, two core-hydrogen burning massive stars are merged using the moving mesh code, AREPO, with a resolution of 400 000 to 4 million cells and the merged stars were then mapped into a 1D code to study the evolution of the remnant (Schneider et al. 2019). A simulation of the V1309 Sco merger was carried out using 100 000 SPH particles (Nandez et al. 2014), and then used as a starting point for a detailed discussion aided by analytical physics. A number of 3D hydrodynamic simulations were also used to model other binary interactions such as the mass transfer preceding coalescence (e.g., Pejcha et al. 2016; MacLeod & Loeb 2020). Also, Kashyap et al. (2018) carried out 200 000 particle SPH simulations of WD mergers that then were mapped into a grid code to investigate the detonation properties.

Much of the research in 3D hydrodynamic simulations of WD mergers is aimed at understanding the dynamics of detonation with a goal to understand Type Ia supernovae. In their detailed review of 3D, WD merger simulations, Katz et al. (2016) described several efforts, most of which concentrate on the details of the properties at the time of merger but do not necessarily model the early mass-transfer and merger phase. This review cites the early pioneering efforts to model the entire merger by Motl et al. (2002), D'Souza et al. (2006) and Motl et al. (2007). In a recent counterpart to those papers, Motl et al. (2017), carried out simulations of merging WD stars, using a finite difference technique code with up to 4 million cells and an SPH code with up to 1 million particles. Aside from the importance of code comparison, that publication explicitly shows the resolution and wall clock time constraints of this type of simulations.

In this paper, we present OCTO-TIGER, a code that aims at improving 3D simulations of interactions using a number of computational techniques that increase accuracy and scalability. This will allow us to calculate full mergers with reasonably high resolution, and reasonable wall clock times and has the capacity to include a greater amount of physics without moving the computation into the realm of impossibility. OCTO-TIGER also conserves energy and angular momentum to excellent precision. OCTO-TIGER's main application currently is simulating the merger of well resolved stars that can be represented via polytropes, such as main sequence stars or WDs. Marcello et al. (2016) presented OCTO-TIGER, with a description of the governing equations along with some preliminary tests. Staff et al. (2018) and Kadam et al. (2018) used OCTO-TIGER in parallel with a suite of other codes to simulate WD binary mergers leading to R Coronae Borealis stars and contact binaries. These studies provided a test of sorts for OCTO-TIGER, but the scope of those papers was such that a systematic verification and validation of OCTO-TIGER was not carried out, and neither was a scaling test aimed at measuring its speed. In the meantime, several improvements have been implemented in the code. It is therefore appropriate and timely to test and document OCTO-TIGER in a systematic way, by carrying out a suite of standard benchmark simulations, a comparison to other codes, complete with scaling tests of the latest code version.

This paper is structured in the following way. In Section 2 we describe OCTO-TIGER's underlying equations. In Section 3 we present the benchmarks, starting with the shock tube (Section 3.2), the Sedov blastwave (Section 3.3), a uniform static sphere (Section 3.4) and continuing with a static pulsating polytrope (Section 3.5), a translating polytrope (Section 3.6) and a rotating polytrope (Section 3.7). We conclude with a binary simulation with a mass ratio of 0.5 in Section 4, and an assessment of the scaling properties of OCTO-TIGER in Section 5. We conclude in Section 6.

2 THE AMR CODE OCTO-TIGER

OCTO-TIGER is an Eulerian AMR code, optimised for the simulation of inviscid, compressible fluids. Below we fully describe its governing equations and numerical methods.

2.1 Hydrodynamic Evolution Equations

Octo-Tiger evolves Euler's inviscid equations of motion for a self-gravitating fluid on a rotating mesh. The evolution equations are:

$$\frac{\partial}{\partial t}\rho_m + \boldsymbol{\nabla}\cdot\rho_m\mathbf{v} = 0, \tag{1}$$

$$\frac{\partial}{\partial t}\mathbf{s} + \boldsymbol{\nabla}\cdot\mathbf{v}\mathbf{s} + \boldsymbol{\nabla}p = \rho\mathbf{g} - \Omega\times\mathbf{s}, \tag{2}$$

$$\begin{aligned}\frac{\partial}{\partial t}\left(E+\frac{1}{2}\rho\phi\right)+\boldsymbol{\nabla}\cdot\mathbf{v}\left(E+\rho\phi\right)+\boldsymbol{\nabla}\cdot\mathbf{u}p\\ =\frac{1}{2}\left(\phi\frac{\partial}{\partial t}\rho-\rho\frac{\partial}{\partial t}\phi\right)+\rho\Omega\cdot\left(\mathbf{x}\times\mathbf{g}\right),\end{aligned} \tag{3}$$

and

$$\nabla^2\phi=4\pi G\rho, \tag{4}$$

where ρ_m is the mass density of the m^{th} species, $\mathbf{v}$ is the velocity in the rotating frame, $\mathbf{s}$ is the inertial frame momentum density, p is the gas pressure, $\mathbf{u}$ is the inertial frame velocity, $\mathbf{g}$ is the gravitational acceleration, Ω is the rotational frequency of the grid, E is the gas internal plus bulk kinetic energy density in the inertial frame, $\rho=\sum\rho_m$ is the total mass density, $\mathbf{x}$ is the position vector on the grid, ϕ is the gravitational potential and G is the gravitational constant. The rotating frame velocity is related to the inertial frame momentum density by

$$\mathbf{s}=\rho\left(\mathbf{v}+\Omega\times\mathbf{x}\right). \tag{5}$$

The gravitational potential is related to the gravitational acceleration by $\mathbf{g}=-\boldsymbol{\nabla}\phi$. The time derivatives are taken in the rotating frame. These are related to time derivatives in the inertial frame by

$$\frac{\partial}{\partial t}=\left(\frac{\partial}{\partial t}\right)_{\text{inertial}}+\left(\mathbf{x}\times\Omega\right)\cdot\nabla \tag{6}$$

A previous version of OCTO-TIGER used a hydro-solver that conserved linear and angular momentum to machine precision. This was based on the work by Després & Labourasse (2015). We successfully adapted this method to OCTO-TIGER for use when velocities are reconstructed in the inertial frame. However, our methodologies could not be applied to when velocities are reconstructed in the rotating frame. The rotating frame is a better choice when dealing with binary interactions, because equilibrium rotating stars retain their initial profiles much better. We thus abandoned the development of the hydrodynamics angular momentum conservation feature. A legacy of this development work is a separately evolved angular momentum field, $\mathbf{l}$, described by the following equation:

$$\frac{\partial}{\partial t}\mathbf{l}+\boldsymbol{\nabla}\cdot\mathbf{v}\mathbf{l}+\nabla\times\mathbf{x}p=\rho\mathbf{x}\times\mathbf{g}-\Omega\times\mathbf{l}. \tag{7}$$

In the current version, this quantity is evolved passively, meaning the evolution of the other variables does not depend on the value of $\mathbf{l}$. It is initialized to $\mathbf{x}\times\mathbf{s}$ at $t=0$. Although our gravity solver exactly conserves angular momentum to machine precision, because of numerical viscosity in the hydrodynamics solver, angular momentum is not exactly conserved. The difference between $\mathbf{x}\times\mathbf{s}$ and $\mathbf{l}$ after $t=0$ gives a measure of the error in angular momentum conservation.

The source terms on the righthand side (RHS) of Equation 2 include a gravitational and a rotational term. The rotational term accounts for the rotation of the momentum vector relative to the rotating mesh. Because we evolve inertial frame quantities on a rotating mesh, as opposed to rotating frame quantities on a rotating mesh, this term is half the Coriolis force (only half because velocities are taken with respect to the rotating grid, but momenta are calculated in the inertial frame). Note that if $\int_V\mathbf{s}=0$, this term does not violate momentum conservation.

Rather than solving Equation 4 using an iterative approach, OCTO-TIGER uses the fast multipole method (FMM; described below). To calculate the the first term on the RHS of Equation 3, we solve for $\frac{\partial}{\partial t}\phi$ using the FMM with the numerically computed value of $\frac{\partial}{\partial t}\rho$ as the source term. This results in the two parts of this term cancelling when summed over the entire grid.

Equation 3 is derived from the usual form of the energy equation:

$$\frac{\partial}{\partial t}E+\boldsymbol{\nabla}\cdot\mathbf{v}E+\boldsymbol{\nabla}\cdot\mathbf{u}p=-\rho\mathbf{u}\cdot\boldsymbol{\nabla}\phi, \tag{8}$$

along with Equation 1. Equation 3 is written in a form that emphasizes that the conserved quantity is the gas energy, E (internal plus bulk kinetic), plus the potential energy or $E+\frac{1}{2}\rho\phi$. The first term on the RHS of Equation 3 vanishes globally because ϕ is linearly related to ρ, while the second term vanishes globally because the total change in angular momentum due to gravity over all space is zero. As discussed by Marcello & Tohline (2012), evolving the energy equation in this form prevents violation of energy conservation due to matter moving up or down a potential well because of numerical viscosity. Because our gravity solver conserves linear and angular momenta to machine precision, using this form of the energy equation conserves total energy to machine precision. In practice, we actually evolve E instead of $E+\frac{1}{2}\rho\phi$, by taking the discretized evolution equation for Equation 3 and solving it for E.

Following the dual energy formalism of Bryan et al. (1995), OCTO-TIGER evolves a second variable for the energy. While they used the internal energy density as the second energy variable, we use the "entropy tracer" (Motl et al. 2002), defined as

$$\tau=\left(\rho\varepsilon\right)^{\frac{1}{\gamma}}, \tag{9}$$

where ε is the specific internal gas energy and $\gamma=\frac{5}{3}$ is the ratio of specific heats. When there are no shocks, τ is conserved and evolves as

$$\frac{\partial}{\partial t}\tau+\boldsymbol{\nabla}\cdot\tau\mathbf{v}=0. \tag{10}$$

The specific internal energy, ε, is computed according to

$$\rho\varepsilon=\begin{cases}E-\frac{1}{2}\rho u^2, & \text{if } E-\frac{1}{2}\rho u^2\geq\epsilon_1 E\\ \tau^\gamma, & \text{otherwise}\end{cases}, \tag{11}$$

where the default value of $\epsilon_1=0.001$. Once we have the internal energy density we can compute the pressure with the ideal gas equation:

$$p=(\gamma-1)\rho\varepsilon. \tag{12}$$

At the end of every update of the evolution variables, the entropy tracer is then reset using Equation 9 in computational cells which satisfy

$$E-\frac{1}{2}\rho u^2>\epsilon_2 E \tag{13}$$

for at least one of the adjacent cells or the cell itself, where the default value of $\epsilon_2=0.1$. Otherwise it is left alone. We use the values for ϵ_1 and ϵ_2 chosen by Bryan et al. (1995) .

This treatment allows for the proper evolution of shocks while simultaneously retaining numerical accuracy of the internal energy in high mach flows, where the kinetic energy dominates. It is roughly analogous to adding extra digits of precision to the internal energy. This method also guarantees positive values for the internal energy even when $E-\frac{1}{2}\rho u^2<0$.

OCTO-TIGER evolves the gas using an ideal gas equation of state (EoS), but by setting $\epsilon_1=\epsilon_2=1$, shock heating is eliminated and effectively OCTO-TIGER evolves the gas with a poly-

tropic EoS. If

$$p_{\rm poly} = K\rho^{1+\frac{1}{n}}, \tag{14}$$

initially, where K and n are constants, this condition will continue to hold as the system evolves.

2.2 Gravity Update and the Angular Momentum Correction

OCTO-TIGER uses a variant of the Cartesian FMM described by Dehnen (2000). This method conserves linear momentum to machine precision. The FMM adapts naturally to an oct-tree based adaptive mesh refinement (AMR) scheme such as that used by OCTO-TIGER. The multipoles of each cell are composed of the multipoles of its child cells, and the expansion in each child cell is derived from the expansion of its parent cell. Both multipole and expansions are relative to cell centres.

OCTO-TIGER has an extension, detailed by Marcello et al. (2016), which allows its gravity solver to conserve both angular and linear momentum to machine precision. Based on the FMM described by Dehnen (2000), which conserves linear momentum to machine precision, the OCTO-TIGER extension works by adding an additional higher order multipoles which is used to calculate a correction to the calculated force. This angular momentum correction cancels the angular momentum conservation violation while preserving linear momentum conservation of the original method. As a result of angular momentum conservation, the last term in the RHS of Equation 3 also sums to zero over all space, enabling conservation of energy to machine precision in the rotating frame.

Conservation of angular momentum and energy is important for the self-consistent accurate modelling of near equilibrium astrophysical systems such as binary stars in the early phases of mass transfer. Violations in energy conservation can cause stars in equilibrium to "evaporate", while the balance of angular momentum may affect the ultimate merger if one occurs.

The FMM requires the specification of an opening criterion. Given two cells, A and B, at grid locations $\mathbf{x}_A$ and $\mathbf{x}_B$, respectively, the opening angle is

$$\theta' = \frac{\Delta x}{|\mathbf{x}_A - \mathbf{x}_B|}, \tag{15}$$

where Δx is the width of a grid cell. If $\theta' < \theta$, where θ is a critical value less than unity, then two cells are well separated. Two well separated cells interact through the multipole interaction when they are well separated from each other, but their respective parents are not well separated. In the models presented in this paper, $\theta = 0.5$ or 0.34, the latter being used for the interacting binary models.

We also define the quantity, $\theta_{\rm min}$, to refer to the lowest θ allowed by OCTO-TIGER. This is determined at compile time and its value depends on the size of the sub-grids, with larger sub-grids allowing for smaller values of $\theta_{\rm min}$. For $8 \times 8 \times 8$ sub-grids, as used in this paper, $\theta_{\rm min} = 0.34$.

It should be noted the current version of OCTO-TIGER does not conserve angular momentum in the hydrodynamics module. We have experimented with extending an angular momentum conserving hydrodynamics method described by Després & Labourasse (2015) for use in OCTO-TIGER. While we have had success when the grid is not rotating, the method fails in the rotating frame. The reconstruction of face values are computed in the rotating frame, eliminating the degree of freedom in the reconstruction required for Després & Labourasse (2015) to work.

2.3 Hydrodynamic Update

The hydrodynamic update begins by reconstructing cell averaged values on the surface of the cells. OCTO-TIGER reconstructs values at the geometric centres of the the 6 cell faces, 12 cell edges, and 8 cell vertices, for a total of 26 quadrature points. Note this results in each face having 9 quadrature points, 1 at the cell face centre, and 4 each for the cell face's edges and vertices. This allows OCTO-TIGER to compute the flux through a face as the integral of the fluxes at 9 points on the face.

Rather than reconstructing the vector of conserved quantities,

$$U = \begin{bmatrix} \rho_m \\ \mathbf{s} \\ E \\ \tau \\ \rho\phi \end{bmatrix},$$

we reconstruct

$$V = \begin{bmatrix} \rho_m \\ \mathbf{v} \\ E - \frac{1}{2}\rho u^2 \\ \tau \\ \phi \end{bmatrix},$$

and then transform back to U. The reconstruction of velocities is done in the rotating frame. In OCTO-TIGER the rotating frame is defined as rotating around the z-axis, and the frequency can be specified by the user. For the binary simulations in this paper the rotating frame frequency is the same as the initial orbital frequency of the binary. Reconstructing the velocities in the inertial frame is also a valid choice. However, we have found that single stars in initial equilibrium retain their original density profiles better when reconstructing the rotating frame velocities.

The the total energy, defined as the sum of the potential and gas energy, is a conserved quantity, but it is not a purely hydrodynamic quantity. The local potential plays virtually no role in the formation of discontinuities in the gas energy. When applying a central advection scheme we want the averaging of left and right states to occur over hydrodynamic variables. Furthermore, discontinuities in the potential energy are caused solely by the local mass density, the specific potential energy itself being smooth. For these reasons we treat the two quantities as separate when computing the reconstruction and fluxes.

The lefthand side of Equation 3 is evolved as the sum of two parts,

$$\frac{\partial}{\partial t} E + \boldsymbol{\nabla} \cdot E\mathbf{v} + \boldsymbol{\nabla} \cdot E\mathbf{u}p = 0, \tag{16}$$

and

$$\frac{\partial}{\partial t} \frac{1}{2}\rho\phi + \boldsymbol{\nabla} \cdot \mathbf{v}\rho\phi = 0. \tag{17}$$

Below we show how we combine these two parts to form a single energy equation for the gas energy.

There are left and right values for each interface, and here we denote those with R and L superscripts, respectively. For a quantity u and for every i, j, and k there are eight vertex values,

$$\begin{aligned} &u^{RRR}_{i+1/2j+1/2k+1/2}, u^{RRL}_{i+1/2j+1/2k+1/2}, u^{RLR}_{i+1/2j+1/2k+1/2}, \\ &u^{RLL}_{i+1/2j+1/2k+1/2}, u^{LRR}_{i+1/2j+1/2k+1/2}, u^{LRL}_{i+1/2j+1/2k+1/2}, \\ &u^{LLR}_{i+1/2j+1/2k+1/2}, \text{ and } u^{LLL}_{i+1/2j+1/2k+1/2}, \end{aligned} \tag{18}$$

twelve edge values,

$$\begin{array}{l} u^{RR}_{i+1/2j+1/2k}, u^{RL}_{i+1/2j+1/2k}, u^{LR}_{i+1/2j+1/2k}, u^{LL}_{i+1/2j+1/2k}, \\ u^{RR}_{i+1/2jk+1/2}, u^{RL}_{i+1/2jk+1/2}, u^{LR}_{i+1/2jk+1/2}, u^{LL}_{i+1/2jk+1/2}, \\ u^{RR}_{ij+1/2k+1/2}, u^{RL}_{ij+1/2k+1/2}, u^{LR}_{ij+1/2k+1/2}, \text{ and } u^{LL}_{ij+1/2k+1/2}, \end{array} \tag{19}$$

and six face values,

$$\begin{array}{l} u^{R}_{i+1/2jk}, u^{L}_{i+1/2jk}, \\ u^{R}_{ij+1/2k}, u^{L}_{ij+1/2k}, \\ u^{R}_{ijk+1/2}, \text{ and } u^{L}_{ijk+1/2}. \end{array} \tag{20}$$

Note that the u used here is not the inertial frame velocity used elsewhere in this paper. We use the piecewise parabolic method (PPM) of Colella & Woodward (1984, with contact discontinuity detection) to compute these values. The five cell stencil required for PPM is taken along the line from the cell centre to each face, edge, or vertex. For example, the reconstructed values at $u^{L}_{i+1/2jk}$ and $u^{R}_{i-1/2jk}$ are computed by applying PPM to the five cell stencil formed by $u_{i-2,j,k}$,$u_{i-1,j,k}$,$u_{i,j,k}$,$u_{i+1,j,k}$, and $u_{i+2,j,k}$, while the reconstructed values at $u^{LR}_{i+1/2j-1/2k}$ and $u^{RL}_{i-1/2j+1/2k}$ are computed with the stencil formed by $u_{i-2,j+2,k}$,$u_{i-1,j+1,k}$,$u_{i,j,k}$,$u_{i+1,j-1,k}$,and $u_{i+2,j-2,k}$. Reconstructing the primitive variables at 26 points across the cell's surface, using PPM, results in multi-dimensional third order convergence. Without this feature, the atmospheres of equilibrium stars turn into box like structures.

Once a cell's evolved quantities are reconstructed at each of the 26 quadrature points on the cells' surfaces, OCTO-TIGER computes the fluxes at each of the 9 quadrature points for each cell's face. We use the central-upwind scheme described by Kurganov et al. (2000). This scheme was originally chosen because it was straightforward to adapt it for use with OCTO-TIGER's hydrodynamics angular momentum conservation feature, as described in Section 2.1. It can be defined in terms of left and right face values,

$$H\left(U_L, U_R\right) = \frac{a^+ F(U_L) - a^- F(U_R)}{a^+ - a^-} + \frac{a^+ a^-}{a^+ - a^-}\left[U_R - U_L\right], \tag{21}$$

where H is the numerical flux, F is the physical flux, and a^+ and a^- are the positive and negative signal speeds. OCTO-TIGER obtains H for each dimension using the physical flux,

$$F_x = \begin{bmatrix} v_x\rho \\ v_x s_x + p \\ v_x s_y \\ v_x s_z \\ v_x E + u_x p \\ v_x \tau \\ v_x \rho\phi \end{bmatrix} F_y = \begin{bmatrix} v_y\rho \\ v_y s_x \\ v_y s_y + p \\ v_y s_z \\ v_y E + u_y p \\ v_y \tau \\ v_y \rho\phi \end{bmatrix} \quad F_z = \begin{bmatrix} v_z\rho \\ v_z s_x \\ v_z s_y \\ v_z s_z + p \\ v_z E + u_z p \\ v_z \tau \\ v_z \rho\phi \end{bmatrix}, \tag{22}$$

and the signal speeds

$$\begin{array}{l} a^+_x = \max\left(c_{s,R} + v_{x,R}, c_{s,L} + v_{x,L}, 0\right), \\ a^-_x = \min\left(c_{s,R} - v_{x,R}, c_{s,L} - v_{x,L}, 0\right), \\ a^+_y = \max\left(c_{s,R} + v_{y,R}, c_{s,L} + v_{y,L}, 0\right), \\ a^-_y = \min\left(c_{s,R} - v_{y,R}, c_{s,L} - v_{y,L}, 0\right), \\ a^+_z = \max\left(c_{s,R} + v_{z,R}, c_{s,L} + v_{z,L}, 0\right), \\ a^-_z = \min\left(c_{s,R} - v_{z,R}, c_{s,L} - v_{z,L}, 0\right), \end{array} \tag{23}$$

where the c_s are the sound speeds,

$$c_s = \sqrt{\frac{\gamma p}{\rho}}. \tag{24}$$

The total flux through a face is then obtained by summing the fluxes taken at each quadrature point on the face. For the x-face, for instance, we have

$$\begin{array}{l} \mathcal{H}_{x,i+1/2jk} = \\ \frac{16}{36}\left[H_x(U^L_{i+1/2jk}, U^R_{i+1/2jk})\right] + \\ \frac{4}{36}\left[H_x(U^{LL}_{i+1/2j+1/2k}, U^{RL}_{i+1/2j+1/2k}) + \right. \\ H_x(U^{LR}_{i+1/2j-1/2k}, U^{RR}_{i+1/2j-1/2k}) + \\ H_x(U^{LL}_{i+1/2jk+1/2}, U^{RL}_{i-1/2jk+1/2}) + \\ \left. H_x(U^{LR}_{i+1/2jk-1/2}, U^{RR}_{i-1/2jk-1/2})\right] + \\ \frac{1}{36}\left[H_x(U^{LLL}_{i+1/2j+1/2k+1/2}, U^{RLL}_{i+1/2j+1/2k+1/2}) + \right. \\ H_x(U^{LLR}_{i+1/2j+1/2k-1/2}, U^{RLR}_{i+1/2j+1/2k-1/2}) + \\ H_x(U^{LRL}_{i+1/2j-1/2k+1/2}, U^{RRL}_{i+1/2j-1/2k+1/2}) + \\ \left. H_x(U^{LRR}_{i+1/2j-1/2k-1/2}, U^{RRR}_{i+1/2j-1/2k-1/2})\right], \end{array} \tag{25}$$

where $\mathcal{H}_{x,i+1/2jk}$ is the total numerical flux through the x-face.

The semi-discrete form of the evolution equations then becomes

$$\begin{array}{l} \frac{d}{dt}U_{ijk} + \\ \quad \frac{H_{x,i+1/2jk} - H_{x,i-1/2jk}}{\Delta x} + \\ \quad \frac{H_{y,ij+1/2k} - H_{y,ij-1/2k}}{\Delta x} + \\ \quad \frac{H_{z,ijk+1/2} - H_{z,ijk-1/2}}{\Delta x} = S_{ijk}, \end{array} \tag{26}$$

where we have added the source term S_{ijk} (defined below).

We still need to re-combine the gas energy and potential energy parts of the flux to form a single equation for the gas energy. This is accomplished through the transformation

$$\begin{array}{rcl} H_E & \rightarrow & H_E + H_\phi \\ H_\phi & \rightarrow & 0, \end{array} \tag{27}$$

where H_E refers to the gas energy flux and H_ϕ refers to the potential energy flux.

The source term, S, is equal to the RHS of the evolution Equations 1, 2, 3, 4 and 10,

$$S = \begin{bmatrix} 0 \\ \rho\mathbf{g} + \Omega \times \mathbf{s} \\ \frac{1}{2}\left(\phi\frac{\partial}{\partial t}\rho - \rho\frac{\partial}{\partial t}\phi\right) + \Omega \times (\rho\mathbf{x} \times \mathbf{g}) \\ 0 \\ 0 \end{bmatrix} \tag{28}$$

Equations 26 and 27 account for all terms of Equations 1 through 3 and Equation 10 except for the $\frac{\partial}{\partial t}\frac{1}{2}\rho\phi$ term on the LHS of Equation 3. This term is accounted for by updating the total energy every time the FMM solver is called to update the potential,

$$E_{\text{after}} = E_{\text{before}} + \frac{1}{2}\rho\left(\phi_{\text{before}} - \phi_{\text{after}}\right), \tag{29}$$

where the "before" and "after" subscripts refer to before and after the FMM solver is called.

The $\frac{d}{dt}$ operator from Equation 26 is discretized using the third order Total Variation Diminishing Runge Kutta integrator of Shu &

Osher (1989). The time step size, Δt, is chosen by the Courant-Friedrichs-Lewy (CFL) condition

$$\Delta t = \eta_{\rm CFL} \max_{all} \left[\frac{\Delta x}{a^{\pm}_{x|y|z}} \right], \tag{30}$$

where the maximum is taken over all the computational cells in the domain and $\eta_{\rm CFL}$ is a positive dimensionless constant less than $\frac{1}{2}$. For the binary simulations presented below, $\eta_{\rm CFL} = \frac{4}{15}$.

A schematic representation of the execution of the steps is shown in Algorithm 1.

```
Set initial conditions;
Do initial output;
while do
    for i ← 1 to 3 do
        if i == 1 then
            U_0 ← U;
            Compute Δt;
        end
        Compute (∂/∂t)U due to advection only;
        Compute g, φ, and (∂/∂t)φ using the FMM;
        Add the source terms to (∂/∂t)U;
        Update U from (∂/∂t)U and U_0 using the 3rd order
         R-K method;
        Update the entropy tracer using the dual energy
         formalism;
        if using floors then
            Apply floor values for density and entropy;
        end
    end
    if time to check refinement then
        refine and de-refine the AMR mesh as needed;
        redistribute the work-load;
    end
    if time to output then
        output to SILO file;
    end
end
```

Algorithm 1: The algorithm Octo-Tiger uses for evolution

2.4 Adaptive Mesh Refinement

OCTO-TIGER uses an oct-tree based AMR. Each node of the oct-tree has associated with it a single $N \times N \times N$ sub-grid and is either fully refined with eight child nodes (an interior node) or not refined at all (a leaf node). For the simulations in this paper, the sub-grids' interior size is $8 \times 8 \times 8$. PPM requires a 3 cell boundary for a total sub-grid size of $14 \times 14 \times 14$. OCTO-TIGER properly nests the sub-grids, meaning there can be no more than one jump in refinement levels across sub-grid boundaries.

We define an "AMR boundary" as a sub-grid boundary across which the refinement level changes. Because of proper nesting, this involves only two refinement levels, which we refer to here as the "fine" and "coarse" level. The fine sub-grid at an AMR boundary cannot get its boundary cells from a neighboring sub-grid of the same refinement level, so it must interpolate its boundary from the coarse sub-grid sharing the same boundary. To describe our AMR boundary interpolation scheme, we use an indexing system, which is aligned such that $\mathbf{x}^C_{i-1/2j-1/2k-1/2} = \mathbf{x}^F_{2i-1,2j-1,2k-1}$, where the superscripts "C" and "F" refer to the coarse and fine levels, respectively. The coarse sub-grid has cell centres at $\mathbf{x}_{ijk}$ while the fine sub-grid has cell centres at half integer locations, $\mathbf{x}^F_{2i\pm1/2,2j\pm1/2,2k\pm1/2}$. The slopes for our interpolation scheme for a variable u (note, the u variable used here is not the inertial frame velocity used everywhere else in this paper) are

$$\begin{aligned}
u^C_{x,ijk} &= \text{minmod}[u^C_{i+1jk}, u^C_{ijk}, u^C_{i-1jk}] \\
u^C_{y,ijk} &= \text{minmod}[u^C_{ij+1k}, u^C_{ijk}, u^C_{ij-1k}] \\
u^C_{z,ijk} &= \text{minmod}[u^C_{ijk+1}, u^C_{ijk}, u^C_{ijk-1}] \\
u^C_{xy,ijk} &= \text{minmod}[u^C_{i+1j+1k}, u^C_{ijk}, u^C_{i-1j-1k}] \\
u^C_{xz,ijk} &= \text{minmod}[u^C_{i+1jk+1}, u^C_{ijk}, u^C_{i-1jk-1}] \\
u^C_{yz,ijk} &= \text{minmod}[u^C_{ij+1k+1}, u^C_{ijk}, u^C_{ij-1k-1}] \\
u^C_{xyz,ijk} &= \text{minmod}[u^C_{i+1j+1k+1}, u^C_{ijk}, u^C_{i-1j-1k-1}].
\end{aligned} \tag{31}$$

where *minmod* is zero if the signs are opposite, or the minimum absolute value is taken if the signs are the same. The interpolation scheme is then

$$\begin{aligned}
u^F_{2i+q,2j+r,2k+s} = u^C_{ijk} &+ \frac{9}{64}\Big(\text{sgn}[q]u^C_{x,i,j,k} + \text{sgn}[r]u^C_{y,i,j,k} \\
&+ \text{sgn}[s]u^C_{z,i,j,k}\Big) + \frac{3}{64}\Big(\text{sgn}[q]\text{sgn}[r]u^C_{xy,i,j,k} \\
&+ \text{sgn}[q]\text{sgn}[s]u^C_{yz,i,j,k} \\
&+ \text{sgn}[r]\text{sgn}[s]u^C_{xz,i,j,k}\Big) \\
&+ \frac{1}{64}\text{sgn}[q]\text{sgn}[r]\text{sgn}[s]u^C_{xyz,i,j,k},
\end{aligned} \tag{32}$$

where $q = \pm 1/2$, $r = \pm 1/2$, and $s = \pm 1/2$ and we emphasise that the variable s does not mean the inertial frame momentum density of Equation 5 and is used only here. The string "sgn" means the sign of q, r and s.

To ensure conservation across the AMR boundary, coarse fluxes on AMR boundaries are taken to be the sum of the fine fluxes through the cell face:

$$\begin{aligned}
H^C_{x,i+1/2jk} = &\frac{1}{4}H^F_{x,2i+1,2j+1/2,2k+1/2} \\
&+ \frac{1}{4}H^F_{x,2i+1,2j+1/2,2k-1/2} + \frac{1}{4}H^F_{x,2i+1,2j-1/2,2k+1/2} \\
&+ \frac{1}{4}H^F_{x,2i+1,2j-1/2,2k-1/2}.
\end{aligned} \tag{33}$$

The refinement criteria for a cell to be refined is either

$$\frac{\Delta x}{\rho}\frac{d\rho}{dx_i} > 0.1, \tag{34}$$

or

$$\frac{\Delta x}{\tau}\frac{d\tau}{dx_i} > 0.1, \tag{35}$$

for any i direction, where ρ is the density and τ is the entropy tracer of Equation 9.

2.5 Boundary Conditions

OCTO-TIGER has three boundary conditions available. With "inflow" boundary conditions, the ghost cells at the edge of the physical domain are copied from the closest interior cell. "Outflow" boundary conditions are the same as inflow except that the momentum is set to zero in the case where the momentum in the closest interior cell points inwards. This prevents the artificial creation of inflows. For our binary simulations we generally use the outflow

boundary condition. These boundary conditions are meant to simulate isolated systems, and do not need to be accounted for by the FMM gravity solver.

A reflecting boundary condition is available for pure hydrodynamics runs with no gravity. Although it is certainly feasible to incorporate reflecting boundary conditions into the FMM and periodic boundary conditions into both the FMM and the hydrodynamics solver, to incorporate these boundary conditions into the FMM solver is a non-trivial task. Since OCTO-TIGER does not use these types of boundary conditions to simulate binary systems they were not implemented at this time.

2.6 Minimum Values for Density and Entropy

Both the mass density and entropy tracers should be always positive. This condition is difficult to maintain numerically when there is a large range of densities present. For our the polytropic binary simulation detailed in Section 4, we fill the regions around the stars with a density equal to 10^{-10} times the maximum density on the grid. The density can drop below this value, because the star's gravity is constantly pulling matter from the near-vacuum region onto the stars. OCTO-TIGER also uses a third order Runge Kutta scheme, so even without the presence of gravity, it is not possible to know *a priori* the maximum time step size needed to guarantee positivity. For these reasons, OCTO-TIGER includes options to set minimum values for the mass density and entropy tracer. These floor values are denoted ρ_f and τ_f, respectively.

Imposing a density floor, results in imposing values on other variables that depend on density. We thus define the density floor scaling parameter,

$$f_\rho = 1 - \max\left[1 - \frac{\max[\rho, 0]}{\rho_f}, 0\right] \tag{36}$$

allowing us to transform the variables according to

$$\begin{array}{ccc} \mathbf{s} & \rightarrow & \mathbf{s} f_\rho \\ E & \rightarrow & E f_\rho + \tau_f^\gamma (1 - f_\rho) \\ \tau & \rightarrow & \tau f_\rho + \tau_f (1 - f_\rho) \\ \rho_m & \rightarrow & \max[\rho, \rho_f] \frac{\rho_m}{\rho} \end{array} \tag{37}$$

This deteriorates the strict machine precision conservation of the affected variables, however it only occurs in the near-vacuum regions.

2.7 AMR Refinement Criteria

OCTO-TIGER checks for refinement every $(2/\eta_{\rm CFL})$ time steps, where $\eta_{\rm CFL}$ is from Equation 30. A window of two cells is used for refinement. This condition prevents waves from propagating through an AMR boundary before the relevant cells are checked for refinement. If a sub-grid contains one or more cells flagged for refinement, the sub-grid is refined, converting its leaf node to an interior node with 8 children. Conversely, interior nodes whose 8 children are all leaf nodes are de-refined if none of their cells and none of the children's cells are flagged for refinement. Sub-grids are also flagged for refinement as needed to ensure the difference in refinement levels across grid boundaries is no greater than one.

For the binary simulations in this paper, we use a density based refinement criterion. If $l_{\max}$ is the maximum allowed refinement level and ρ_r is the refinement density cut-off, a cell is flagged for refinement if the maximum level l for which

$$\rho > 8^{l_{\max} - l} \rho_r \tag{38}$$

holds true is greater than the cell's level of refinement. For the binary simulations ρ_r is held constant initially. Once material ejected from the binary begins to fill the grid, ρ_r is adjusted dynamically in a manner that attempts to keep the total number of sub-grids at a specified level. After every refinement,

$$\rho_r \rightarrow \left(\frac{N_{\rm grids}}{N_{\rm target}}\right)^2 \rho_r, \tag{39}$$

where $N_{\rm grids}$ is the total number of sub-grids and $N_{\rm target}$ is the desired number.

The CFL factor (Equation 30) typically used by simulations is $\sim$0.4. Technically there is a limit to the maximum CFL number one should use. In three dimensions with PPM, it is $1/7$. This is because the maximum ratio between the density reconstructed at the cell's face and the cell averaged density is $7/3$. As a result, the limit should be $3/7$ in one dimension or $3/(3 \times 7)$ in three dimensions. This limit is stringent, in that it prevents any sharp transition in density from emptying a cell of its content possibly resulting in zero or negative densities. In OCTO-TIGER negative densities are corrected by introducing a "floor" density value in that cell (see Section 2.6), which in turn results in mass non-conservation. Preempting the results of Section 3.6, this can result in mass growth at the level of 0.0001 percent from one time step to the next when highly supersonic motions are present. It is therefore important to critically appraise the value of the CFL factor to be used in each simulations, particularly if strong shocks are present. We discuss these choices and their impacts further in Section 3.6.

2.8 Code Units

There are no physical constants present in the hydrodynamic equations, therefore the simulations without gravity enabled are unitless. With gravity we have a single physical constant, G. In the code the value of this constant is set to unity. We convert between code units and physical units in the cgs system using three conversion factors, $m_{\rm cgs}$, $l_{\rm cgs}$, and $t_{\rm cgs}$ for mass, length, and time, respectively. Because the value of G is fixed at unity, we may specify two of these conversion factors, with the third being determined by the relation

$$\frac{l_{\rm cgs}^3}{m_{\rm cgs} t_{\rm cgs}^2} = G_{\rm cgs}, \tag{40}$$

where $G_{\rm cgs}$ is the value of G in cgs units. In OCTO-TIGER the user specifies $l_{\rm cgs}$ and $m_{\rm cgs}$ and OCTO-TIGER calculates $t_{\rm cgs}$ using Equation 40. When outputting the grid to file, OCTO-TIGER converts all quantities to their cgs equivalents using these unit conversion factors.

2.9 The Temperature in OCTO-TIGER

OCTO-TIGER evolves the density, ρ, total gas energy density (internal and bulk kinetic energy density), E, and velocities, and derives all the other physical quantities based on these values and the EoS. The evolution equations, Equations 1 to 4, do not require knowledge of the temperature. For post processing purposes, the temperature can be computed by assigning atomic mass and atomic numbers to each mass density species and assuming a fully ionized gas according to

$$T = \frac{(\gamma - 1)\rho\varepsilon}{k_B \sum \frac{\rho_m (1 + N_{Z,m})}{m_H N_{A,m}}}, \tag{41}$$

where $N_{A,m}$ and $N_{Z,m}$ are the atomic mass and atomic numbers of the m^{th} species and the specific internal energy ε, comes from Equation 11.

2.10 The C++ Standard Library for Concurrency and Parallelism (HPX)

OCTO-TIGER is parallelized for distributed systems using the C++ Standard Library for Concurrency and Parallelism (HPX, Kaiser et al. 2020; Heller et al. 2019a; Daiß et al. 2019). HPX is an open source C++ Standard Library for Concurrency and Parallelism and is within the class of the so-called asynchronous many-task AMT runtime systems.

Another AMT utilized in astrophysics simulation, e.g., ChaNGa (Jetley et al. 2008) or Enzo-P (Bordner & Norman 2012), is Charm++ (Kale & Krishnan 1993). We focus on the comparison of these two AMTs in this paper, for a more comprehensive review for various AMTs we refer to Thoman et al. (2018). The commonality if HPX and Charm++ is the usage of the same concepts for "parallelism" and "concurrency". The distinctness of HPX is that it fully conforms to the `C++17` ISO standard (The C++ Standards Committee 2017) and implements proposed features of the upcoming `C++20` ISO standard (The C++ Standards Committee 2020). This means that HPX's features that are available in the C++ standard can be replaced without changing the function arguments. From a programmer's perspective, HPX is more an abstraction of the C++ language while Charm++ is more a standalone library. The requirement for OCTO-TIGER (distributed, task-based, asynchronous) are met by only few AMTs and HPX has the highest technology readiness level according to this review (Thoman et al. 2018).

OCTO-TIGER takes advantage of four main features of HPX: (i) fine grained, (ii) task based parallelism through light weight user space threads, (iii) the use of C++ futures to encapsulate both local and remote work, and (iv) an active global address space (AGAS), whereby global objects are remotely and locally accessible (Amini & Kaiser 2019; Kaiser et al. 2014). These global objects reside in the memory on a given node but can be accessed remotely from any node.

Through HPX futures, OCTO-TIGER is able to overlap work with communication in a straightforward and efficient manner. For a given sub-grid, the hydrodynamic and gravity computations are performed by an HPX thread. This thread spawns threads sending boundary data to sibling sub-grids. HPX threads (Kaiser et al. 2009) are lightweight and OCTO-TIGER may spawn hundreds or even thousands of threads per system thread. The sub-grid thread also creates a set of futures encapsulating the boundary data it expects from its siblings. This allows the sub-grid thread to sleep while waiting for its boundary data. When this data becomes available, HPX automatically wakes the thread, allowing computation to begin. The use of HPX futures in this manner allows OCTO-TIGER to overlap work with communication in a natural way. AGAS allows each node of the OCTO-TIGER oct-tree to be distributed across the system in a relatively simple manner, and each oct-tree node can access its children or its siblings using the same constructs regardless of whether a particular child or sibling resides locally or on a remote processor. Another benefit of HPX is that a unified application programmer interface (API) for local and remote functionality is provided. Thus, there is some simplification for the application programmer, since there is no need to deal with two different interfaces, like combining the two hybrid parallel approaches such as MPI and OpenMP. Note that HPX utilises MPI for the communication, but provides an abstraction to the application programmer to hide the direct interaction with the MPI API.

To integrate acceleration cards, like GPUs, HPX provides two approaches: `hpx::compute` (Copik & Kaiser 2017) and `hpx::cl` (Diehl et al. 2018) to overlap GPU kernel execution with CPU work and networking. The GPU kernel execution returns a future allowing asynchronous integration of the GPU work into the overall asynchronous execution flow. `hpx::cl` provides features to integrate existing CUDA kernels and textttthpx::compute automatically generates CUDA kernels from C++ code. OCTO-TIGER extends textttthpx::compute to launch hand-written CUDA kernels. For more implementation details, we refer to (Daiß et al. 2019).

3 BENCHMARKING OCTO-TIGER

Below we carry out a number of simulations aimed at verifying and validating OCTO-TIGER. In doing so, we sometimes compare our benchmark test results with a similar test performed with FLASH (Fryxell et al. 2000) a well used hydrodynamic code often used for astronomical applications including binary interactions.

3.1 Benchmark Design

We start testing the code by running pure hydrodynamic problems, namely, the shock tube problem (Section 3.2), and the blastwave problem (Section 3.3). The purpose is to validate and examine the performance of the hydrodynamic solver of OCTO-TIGER in isolation. This is an important task as this current version of OCTO-TIGER has an extended, more accurate (and more computationally demanding) hydrodynamic solver, which includes reconstruction of the cell averaged values not only on the cell faces and edges but also on the cell vertices (see Section 2.3).

To test our gravity solver in isolation we utilize the grid with the mass distribution of a uniform density sphere and let the code compute the gravitational potential without evolving it dynamically. The sphere is surrounded by a negligible amount of gas (see Section 3.4). OCTO-TIGER's gravity solver uses an opening angle parameter to modulate the accuracy of the gravity solution (see Section 2.2). We examine the accuracy of the computed potential with different values of this parameter as well with different resolutions.

The next test is a simulation of a polytropic structure (which could model certain types of stars) evolved over a number of dynamical timescales. We check the structure stability over several dynamical timescales, for different resolutions, values of the gravity solver opening angle and EoS (see Section 3.5).

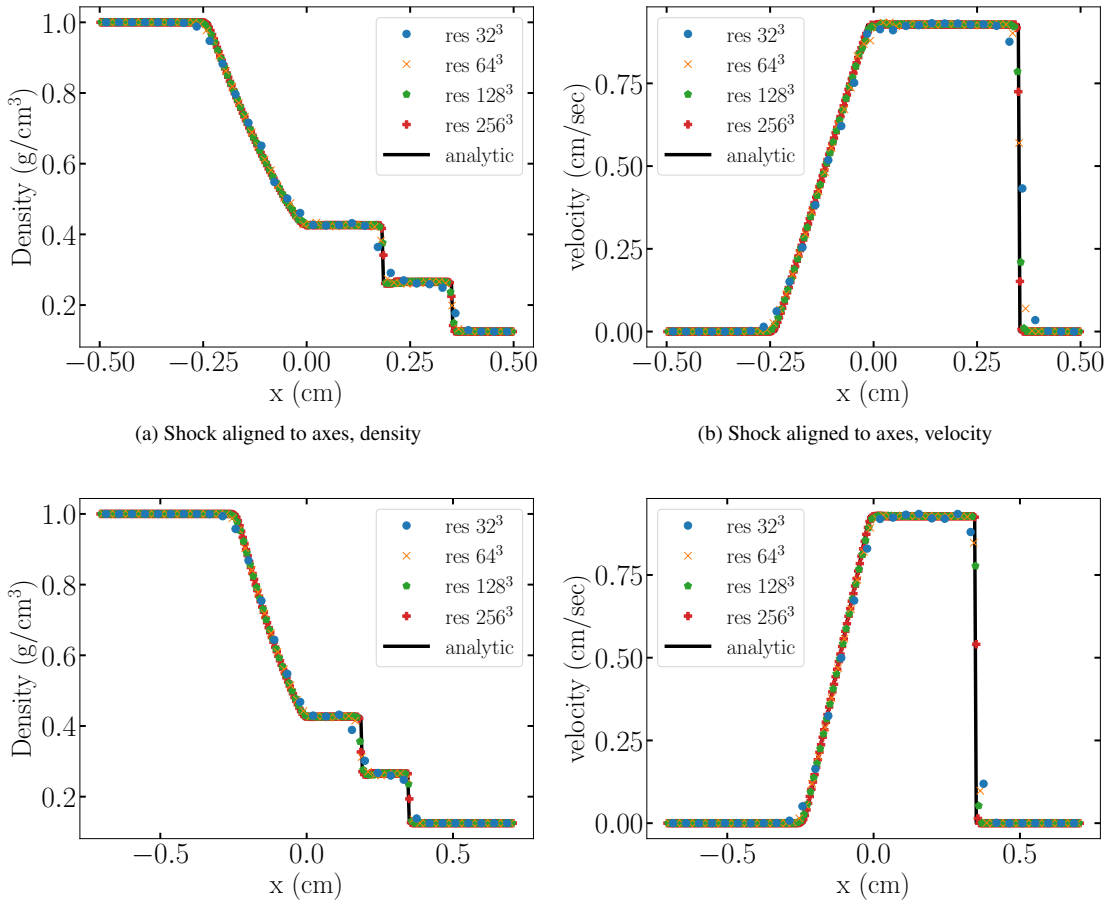
Next, we test the polytrope in a wind tunnel to check how the star behaves as it moves through a hot, low density medium (see Section 3.6). We also test a rotating polytrope and we check for diffusion of the stellar rotation profile over time (see Section 3.7).

Finally, in Section 4, we simulate two polytropic stars orbiting one another in a detached configuration and then merging, with a mass ratio of 0.5. We check stability of the two structures over a number of orbits with a number of resolutions.

We compare some of these tests with equivalent ones using the code FLASH (Fryxell et al. 2000). We also carry out scaling tests in Section 5.

3.2 Shock Tube

To test the hydrodynamic solver we run the Sod shock tube problem (Sod 1978). We use the conventional initial configuration of



(a) Shock aligned to axes, density

(b) Shock aligned to axes, velocity

(c) Shock diagonal to axes, density

(d) Shock diagonal to axes, velocity

Figure 1. Comparison in density and velocity between the numerical results with higher resolution of OCTO-TIGER simulations and the exact solution for the shock tube at $t = 0.2$. Top row: The density and velocity are plotted in the x direction, respectively, at the end of the simulation, along a line perpendicular to the discontinuity and through the centre of the grid. Bottom row: The density and the parallel component of the velocity are plotted along the line that starts from the bottom left corner and ends at the upper right corner (in the xy plane)

this problem: $\rho_l = 1,\ p_l = 1,\ v_l = 0;\ \rho_r = 0.125,\ p_r = 0.1,\ v_r = 0$, where l denotes the left side of the discontinuity, and r the right side. The variables ρ, p, and v are the gas mass density, pressure and velocity, respectively. The gas is taken to be an ideal gas with an adiabatic index of $\gamma = 7/5$ (this value is historical as it pertained to molecular gas typical of air). Although the problem is one-dimensional in nature, we run it in three dimensions for testing. We simulated two configurations: in the first the discontinuity plane is $x = 0$, and in the second the discontinuity plane is $x + y = 0$. From the problem symmetry, planes parallel to the xy plane are identical. We discuss the results of the simulations in the next two subsections.

3.2.1 Shock front aligned along the x-axis

In this simulation we set the initial discontinuity plane to be $x = 0$. As the simulation evolves in time, a shock wave propagates to the right of the box, along the positive x-axis, while a rarefaction wave propagates at the sound speed of the unperturbed denser gas to the left, along the negative x-axis. Between them the density discontinuity moves to the right. To show the convergence of the numeric solution to the analytic one, we run four simulations in which the grid is uniform and have a growing resolution of 64^3, 128^3, 256^3, and 512^3 cells. We stop the simulation at time $t = 0.2$, when the shocks fronts have not yet reached the grid boundary. In Figure 1, top row, we plot the density and velocity in the x direction, respectively, at the end of the simulation, along a line perpendicular to the discontinuity and through the centre of the grid and compare it to the analytic solution.

The simulations, even with low resolution, nicely fit the rarefaction wave. Around the discontinuity, there are cells that underestimate the density behind the shock and overestimate the density in front of the shock. This discrepancy diminishes with higher resolution. We ran, in addition, two simulations in which the shock is aligned to the y-axis and the z-axis. We find the same behaviour along the direction of the shock. The velocities perpendicular to the normal direction of the plane of discontinuity vanish everywhere as they should.

Overall, the Sod problem with a shock aligned to an axis shows agreement between the OCTO-TIGER simulation and the analytical solution. As expected, the numerical solution approaches the analytic one with higher resolution.

3.2.2 Shock front aligned diagonally

To check the effect of a discontinuity that is not aligned along as axis we fixed the discontinuity plane at an angle of 45 degrees from the x-axis (the $x + y = 0$ plane). A strong shock front propagates towards the higher x and y values, to the less dense upper right corner on the xy plane. A discontinuity propagates in this direction as well but with lower speed. A rarefaction wave propagates toward the bottom-left corner at the sound speed. In this configuration the wavefronts do cross the grid boundary. As the simulation evolves, an increasingly larger part of the waves encounters the grid boundary. In principle, the analytic solution is only valid in the regions where the wavefronts did not reach the boundaries, e.g., diagonally from the bottom-left corner towards the upper-right corner.

In Figure 1, bottom row, we plot the density and the parallel component of the velocity along the line that starts from the bottom left corner and ends at the upper right corner (on the xy plane) of four OCTO-TIGER simulations that differ only in their resolutions. Across this line, we notice similar behaviour to when the shock was aligned along the x axis. The simulations reproduce accurately the rarefaction wave. Around the discontinuity, some cells underestimate the density behind the shock and overestimate the density in front of it, a discrepancy that diminishes with higher resolution.

It is, however, interesting to understand the features that appear at the opposite corners as a function of particular boundary conditions. For this test, we run two OCTO-TIGER simulations with 256^3 cell resolution that differ only in their boundary conditions. We also compare the OCTO-TIGER simulations to an identical FLASH (version 4.6) simulation. We run FLASH with the split hydro solver that uses the PPM method. The first boundary condition termed "outflow" in FLASH and "inflow" in OCTO-TIGER (see Section 2.5), means gas can inflow back to the grid. The second boundary condition, called "diode" in FLASH and "outflow" in OCTO-TIGER means no inflow is allowed.

In Figure 2, we present slices along the xy plane ($z = 0$) of the difference in the x components of the velocity between the simulations and the analytic solution, which assumes that there is no boundary. Despite the excellent agreement with the analytical solutions (convergence is first order near the shock front and second order at the rarefaction wave), both FLASH and OCTO-TIGER simulations show similar boundary features at the upper left and bottom right corners. At those corners the density is smaller than the analytical solution due to gas that escapes from the grid. The gas in the upper left corner escapes easily through the $y = 0.5$ boundary and accelerates in the negative x-direction, but it also encounters the gas to its right and decelerates in the positive x-direction. The exact opposite happens at the bottom right corner. For the same reason, the differences in the y-velocities have exactly the same mirror picture. The diode boundary, by virtue of setting to zero the momentum parallel to the boundary component, does not allow velocities in the boundary direction to evolve. The rarefaction wavefront diverts near the boundary and propagates in a perpendicular direction to the boundary. This creates a square pattern at the corners.

3.3 Sedov-Taylor Blast Wave

The Sedov-Taylor blast wave test (Sedov 1946) has an analytic solution in three dimensions. The shock wave it produces is much stronger than that produced by the usual Sod shock tube and its spherical geometry provides a stringent test of the hydrodynamics. The initial conditions are a constant density medium with a value of 1 code unit. The internal energy density is setup so as to be described by a delta function: we approximate this in OCTO-TIGER, in a manner similar to FLASH, namely by setting the internal energy density to $E_0/(160\Delta x^3)$ for the 160 computational cells satisfying $|\mathbf{x}| < 3.5$, and setting its value to 1×10^{-20} for all other cells. We ran this model using AMR with 2, 3, 4, 5, 6, and 7 levels of refinement and the usual 8^3 base grid. The finest grid cell size of the model with 2 levels is 3.1×10^{-2} code units and the finest grid cell size of the model with 7 levels is 9.8×10^{-4} code units.

Density slices at $t = 0.25$ code units are shown for the models with 6 and 7 levels in Figure 3. We also show the AMR grid structure for the run with 7 levels. In Figure 3(d) we show the difference between the computed density and the analytic solution as a function of fine grid cell size. The blast wave is a difficult problem to obtain convergence as it requires high resolution to resolve the shock front. OCTO-TIGER obtains slightly better than 1st order convergence from the run with 6 level of refinement to that with 7 levels.

3.4 Uniform Static Sphere

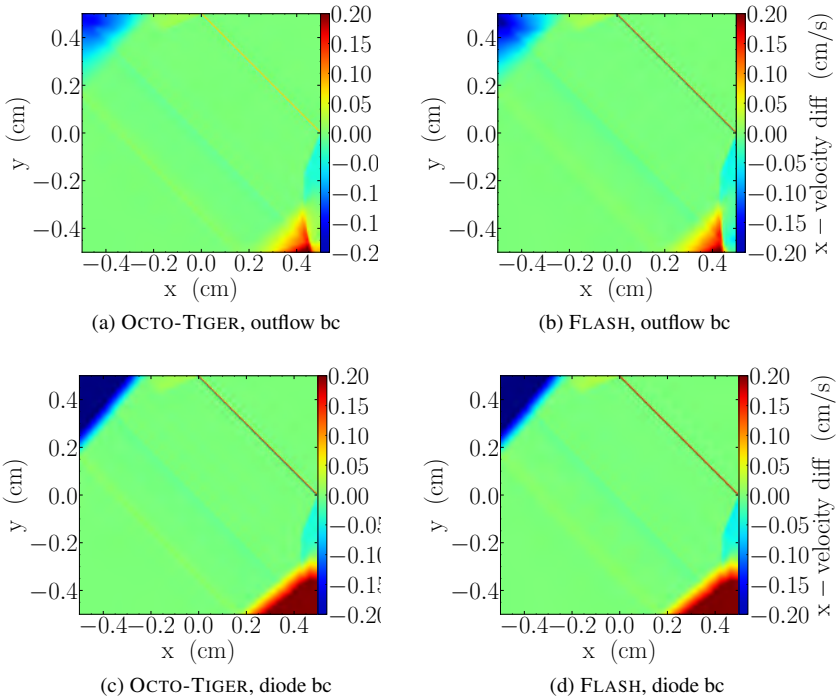
To test the performance of the gravity solver that solves the Poisson equation we initiate a problem with a static uniform density sphere. This problem has a simple analytic solution and is widely used to test solvers for self-gravitating fluids (Motl et al. 2002).

In this test, we place a sphere with a radius of 0.25 code units at the centre of a cubic grid with a length of 1 code units. The total mass inside this sphere is 1 code unit. We fill the grid outside the sphere with a low density of 1×10^{-10} code units. We used a uniform grid with four resolutions: $64^3,\ 128^3,\ 256^3$, and 512^3. To demonstrate the effect of changing θ, the opening-angle parameter (see Section 2.2), we carried out two additional simulations with resolution of 128^3, and 256^3, where $\theta = 0.35$ instead of the default value of 0.5. The OCTO-TIGER simulations reproduce very accurately the analytic gravitational potential. The residuals of the potential on the equatorial plane

$$\epsilon = \left| \frac{\phi - \phi_{\rm analytic}}{\phi_{\rm analytic}} \right|. \tag{42}$$

are plotted in Figure 4 for four of the OCTO-TIGER simulations. A square pattern is apparent for the residuals in the OCTO-TIGER simulations with the maximum residuals appearing at the corners of a square that encloses the sphere. These larger residuals do not decrease with higher resolution, something that is expected of the FMM method. The FMM computes the multipole expansion of the potential between grid cells at the same refinement level as each other. Adding extra refinement to those levels only increases the quality of the solution between cells at the extra refinement level. The cells at the coarser levels still interact with each other in the same way, and the same expansions are passed to the more refined levels. The only way to cause cells to interact with each other at finer levels of refinement is to decrease the opening criterion.

We tested whether a smoother transition of the sphere to the background density can lower the residuals by carrying out simulations of two other density distributions (which have known analytic solutions), one where the density of this sphere decreases to the ambient value by a parabolic dependence with radius (continuous but not differentiable), and a second where the density distribution obeys an $n = 1$ polytropic profile (continuous and differentiable). We find that regardless of the smoothness of the density decrease, high residuals values remain at the corners. To decrease the high residuals at the corners, a lower value of θ needs to be set. We also tried to simulate an off-centre sphere and a range of additional



(a) OCTO-TIGER, outflow bc (b) FLASH, outflow bc

(c) OCTO-TIGER, diode bc (d) FLASH, diode bc

Figure 2. OCTO-TIGER vs. FLASH for a resolution of 256^3 and an angle of 45 deg for the shock tube. Difference in the x-velocities between simulations and analytic solution at the end of the simulation, time $t = 0.2$. The boundary condition in the top row is outflow without material inflow (diode), while in the lower row it is an outflow condition that allows material to inflow back to the simulation domain (outflow)

sphere sizes. Changing the sphere location does not affect the residuals. However, increasing the sphere radius does result in lower maximum residual values. We conclude that the maximum values are related to how much mass is concentrated in a given volume.

For comparison, in Figure 4, we present equatorial maps of several 128^3 and 256^3 cell FLASH simulations that used two gravity solvers: multigrid, panel (e) and BHTree, panel (f). Here we see that each solver has a different residual shape.

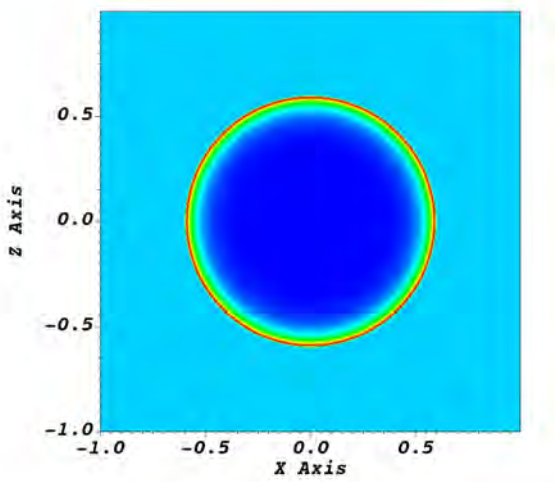
In Figure 5, we plot histograms of the residuals by volume (panels (a) and (c)) and by mass (panels (b) and (d)). Figure 5(a) shows that the residuals' maxima do not decrease by increasing the resolution; however, the residuals' mean and minimum values do. By increasing the resolution, the distribution is stretched to lower residual values. The notable second peak of the histogram for the lower resolution simulations practically disappears for the two higher resolutions. The most common residual value is consistently around 10^{-4} irrespective of resolution.

In Figure 5(c) and (d), we present the residuals' histograms and mass distributions of the FLASH simulations, where we used three commonly-used FLASH gravity solvers: Multigrid, BHTree, and Multipole. The FLASH Multipole gravity solver has the lowest maximum residuals. The 128^3 simulation that used the multipole solver has a lower mean value, $\simeq 2.23 \times 10^{-4}$, than every one of the other FLASH simulations (including the high resolution simulations). Only the OCTO-TIGER simulations have lower mean residuals: 1.93×10^{-4}, 1.67×10^{-4} and 1.65×10^{-4} for the 128^3, 256^3 and 512^3 simulations, respectively. The FLASH multipole solver easily produces an accurate solution to this problem because it has a spherical symmetry. Including additional terms, besides the monopole, does not contribute to the numeric solution in the FLASH multipole solver. The maximum and mean values of the residuals slightly increase in the high resolution FLASH simulations. The OCTO-TIGER simulations show overall low mean residuals and low most common residuals (the peak in the distributions both with respect to volume and mass).

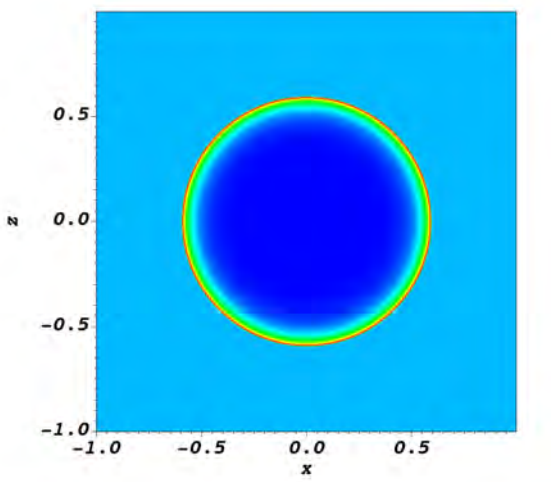
We summarize the residuals' minimum, maximum, and mean values for the OCTO-TIGER and FLASH simulations in Tables 1 and 2, respectively. Table 1 shows that in OCTO-TIGER, a higher resolution decreases the mean residual value, up to a certain resolution, where the mean residual value converges to some low value. To decrease the mean residual further, one must set a lower θ value.

From the mass distributions (Figure 5(b)), we see that the cells which have the highest residual values actually contain a negligible amount of mass. Namely, the cube corners which have high residuals reside outside the uniform density sphere, where the density is very low. In addition, increasing the resolution results in having most of the mass in lower residuals. The residual value, that has the maximum mass, shifts from 3×10^{-3} in the 64^3 simulation to 10^{-4} in the 512^3 simulation, and the mass distribution becomes flatter towards smaller residuals with higher resolution. We infer that these regions with high values of the potential residuals will have minimum effect on the evolution of a simulation, once we evolve it, because of their low total mass.

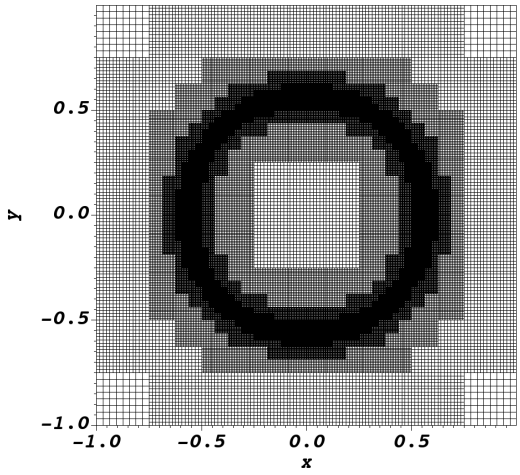
As can be seen from Figures 4 and 5, by decreasing the opening-angle parameter, θ, we can increase the level of accuracy in the OCTO-TIGER simulations even more. The residuals altogether are reduced, including the maximum residual values at the corners. With such low value of θ, we get the lowest mean residual values among all the simulations even at the lower resolution of the 128^3 cells. As with the simulations with $\theta = 0.5$, improving the resolution does not reduce the residual maximum, but does improve the minimum and mean values. By increasing the resolution, the main peak is shifted to lower residual values of around 4×10^{-5}. The mean residual value in the high resolution simulation



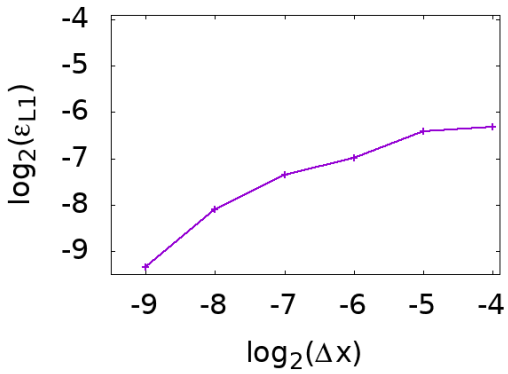
(a) Density slice, 6 levels of refinement



(b) Density slice, 7 levels of refinement



(c) Grid structure with 7 levels of refinement



(d) Density residuals

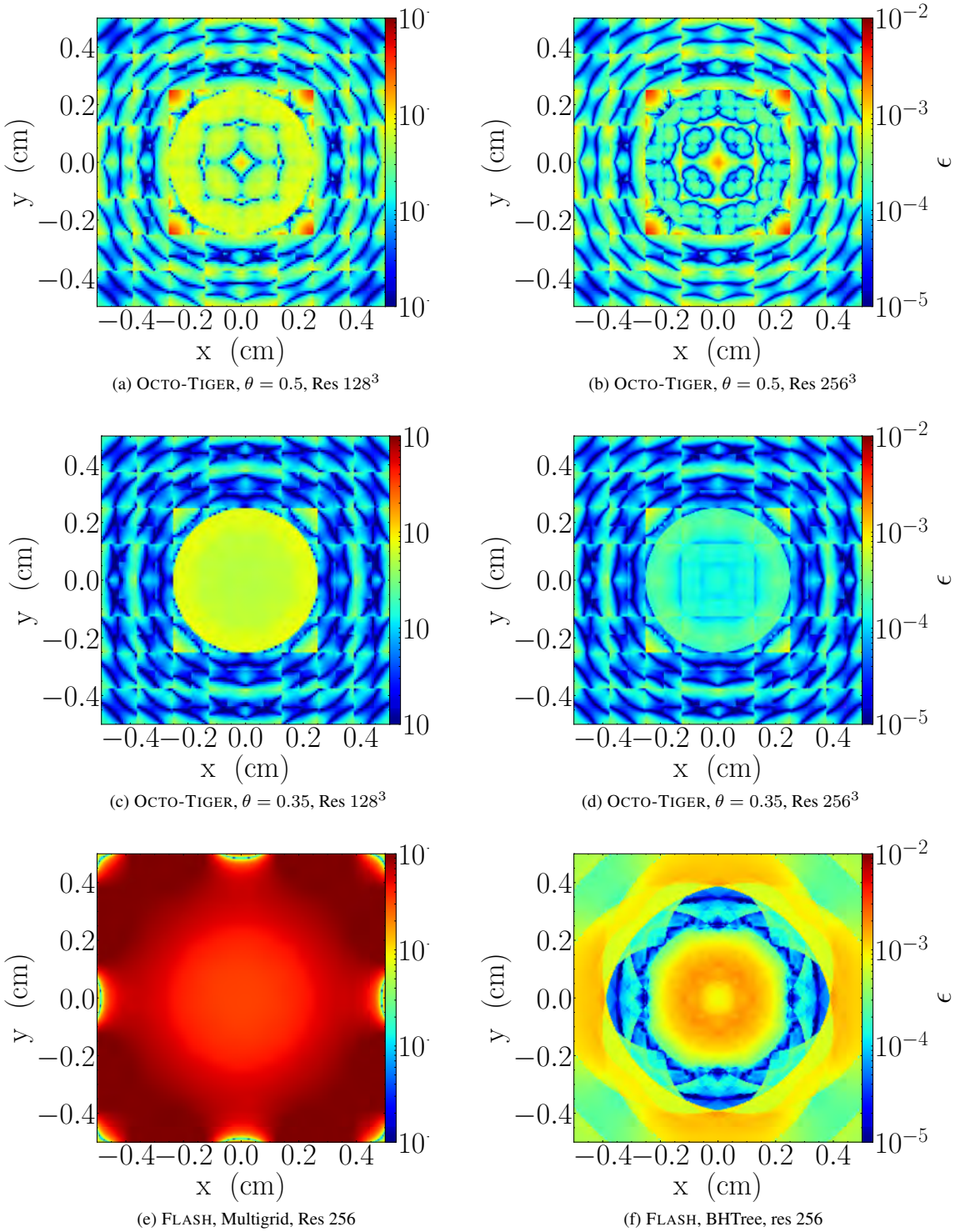
Figure 3. The Sedov-Taylor blast wave test. Density slices using OCTO-TIGER with 6 (a) and 7 (b) levels of refinement at $t = 0.25$ code units. Panel (c): the AMR grid structure of the simulation with 7 levels; Panel (d): the absolute deviation of the density from the analytic solution as a function of grid cell size for 6 simulations carried out with a maximum of 7 levels of refinement to a minimum of 2

N_{cells}	64^3	128^3	256^3	512^3	128^3	256^3
θ	0.5	0.5	0.5	0.5	0.35	0.35
max	2.66×10^{-2}	3.23×10^{-2}	3.49×10^{-2}	3.61×10^{-2}	0.82×10^{-2}	0.90×10^{-2}
mean	3.13×10^{-4}	1.93×10^{-4}	1.67×10^{-4}	1.65×10^{-4}	9.6×10^{-5}	6.5×10^{-5}
min	3×10^{-8}	1×10^{-9}	1×10^{-11}	1×10^{-11}	1×10^{-10}	1×10^{-11}

Table 1. Gravitational potential residuals of the static uniform sphere in OCTO-TIGER. OCTO-TIGER uses the fast mutipole method to solve for the gravity, which has an opening angle parameter θ. Full distributions appear in Figure 5

Solver	BHTree	BHTree	Multigrid	Multigrid	Multipole	Multipole
N_{cells}	128^3	256^3	128^3	256^3	128^3	256^3
max	0.18×10^{-2}	0.20×10^{-2}	1.81×10^{-2}	1.96×10^{-2}	2.6×10^{-4}	2.3×10^{-4}
mean	5.21×10^{-4}	5.38×10^{-4}	7.691×10^{-3}	8.759×10^{-3}	2.23×10^{-4}	2.24×10^{-4}
min	1×10^{-8}	5×10^{-9}	4×10^{-8}	4×10^{-7}	1.1×10^{-4}	1.9×10^{-4}

Table 2. Gravitational potential residuals of the static uniform sphere in FLASH. Full distributions appear in Figure 5



(a) OCTO-TIGER, $\theta = 0.5$, Res 128^3

(b) OCTO-TIGER, $\theta = 0.5$, Res 256^3

(c) OCTO-TIGER, $\theta = 0.35$, Res 128^3

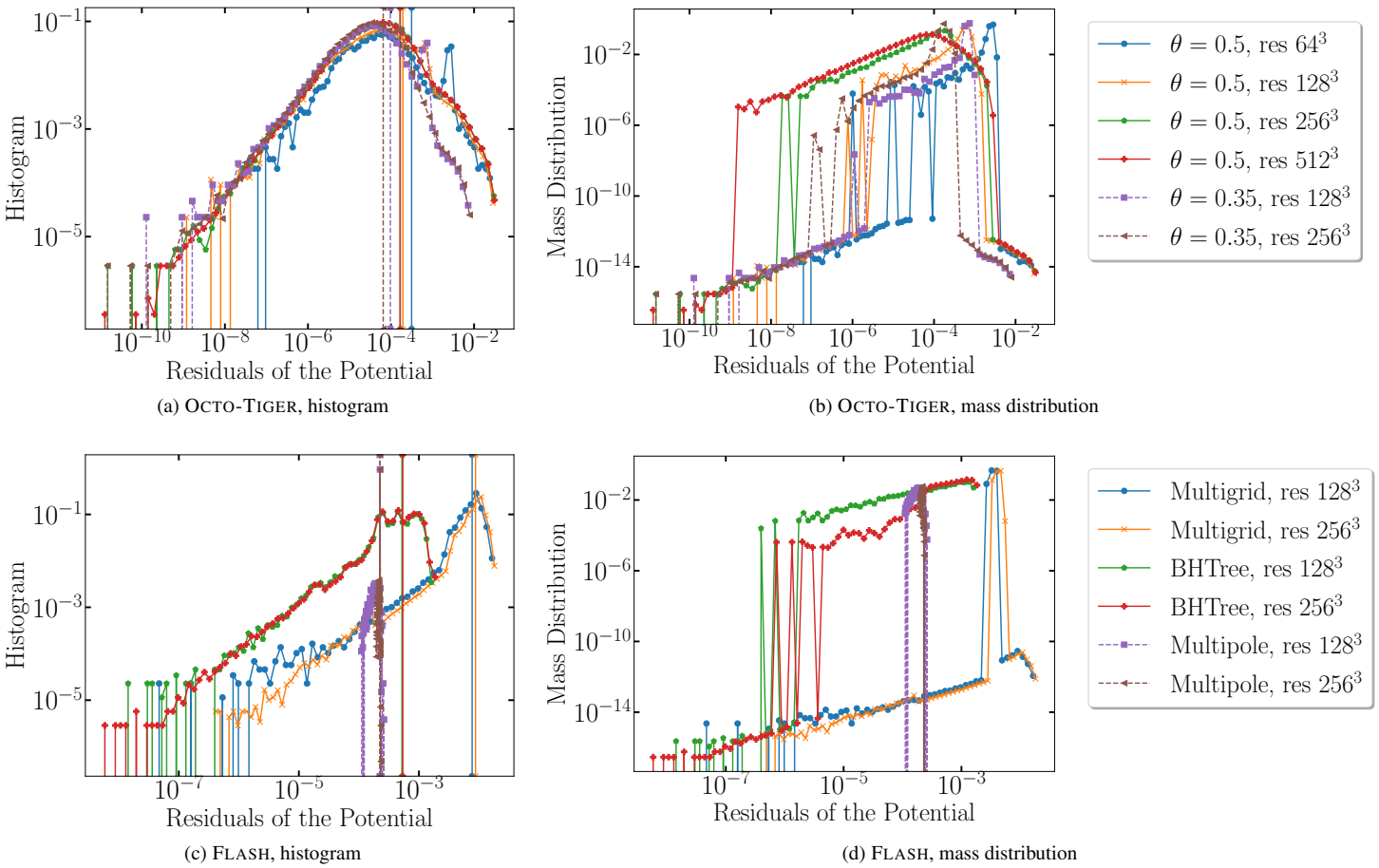
(d) OCTO-TIGER, $\theta = 0.35$, Res 256^3

(e) FLASH, Multigrid, Res 256

(f) FLASH, BHTree, res 256

Figure 4. Uniform static sphere test. Maps of the residuals of the gravitational potential, ϵ, at the equatorial plane $z = 0$ as a function of resolution, modelling code and opening angle θ

indicates a deviation of 0.0065 percent from the analytical value. We note, though, that the peak in the mass distribution of the residuals remains approximately the same for the $\theta = 0.35$ simulations and for the $\theta = 0.5$ simulations (compare the peaks of the purple squares line and the yellow crosses line or the peaks of the green circles line with the brown triangles line in Figure 5(b)). Reducing θ has a higher computational cost (see Section 5). This compromise between the accuracy of the simulation and the running time should be taken into account when simulating a problem.



(a) OCTO-TIGER, histogram

(b) OCTO-TIGER, mass distribution

(c) FLASH, histogram

(d) FLASH, mass distribution

Figure 5. Uniform static sphere test. Histograms and mass distributions of the potential's residuals of OCTO-TIGER and FLASH simulations. The vertical lines in panels (a) and (c) mark the residuals' mean value for each simulation

3.5 Stationary Star

In this test, we set up a polytrope with an index of $n = 3/2$. The stellar diameter is the same as we have used in the uniform static sphere test, equal to half of the domain size, and the polytrope's centre coincides with the centre of the domain. A polytrope can be scaled to model different types of stars. We show two such scalings in Table 3, one for a low-mass, fully convective, main-sequence star and a second one for a low-mass white dwarf. Outside the star we fill the domain with gas with a density that is 10 orders of magnitudes smaller than the star's central density. While the star itself is in good hydrostatic equilibrium, the outside medium is not in equilibrium and it starts free falling onto the surface of the star as the simulation starts. The falling of the gas together with the diode boundary condition, which prevents inflow, create outer regions with very low densities. To avoid the creation of a vacuum, whenever cell densities become lower than the threshold floor density of 10^{-15} times the central density, we set the density to be that value. In addition, to reduce shocks due to supersonic in-falling gas we give the ambient medium a high internal energy (and hence temperature). We list the ambient medium's properties in Table 3.

Any small perturbation from hydrostatic equilibrium will induce an oscillation with polytrope eigenfrequencies and modes. Hurley et al. (1966) computed numerically the fundamental modes, as well as the first and second harmonic modes of several polytropes with different adiabatic indices. They found that a pulsating $n = 3/2$ polytrope has a fundamental frequency of

$$\omega_F^2(n = 3/2;\ \gamma_{\rm ad} = 5/3) = 0.3764\frac{8\pi G\rho_c}{5}. \qquad (43)$$

The pulsation periods are of the order of the free-falling time of the star. For the low mass main sequence star model, for example, the fundamental period, T_F, is 24 minutes, while for the WD model the fundamental period is 18 seconds.

We compare the oscillations observed in the simulations with the analytical solutions. We also monitor the diffusion of the outer layers of the star and the behaviour of the low-density medium that surrounds the structure and we verify the conservation of some basic physical quantities.

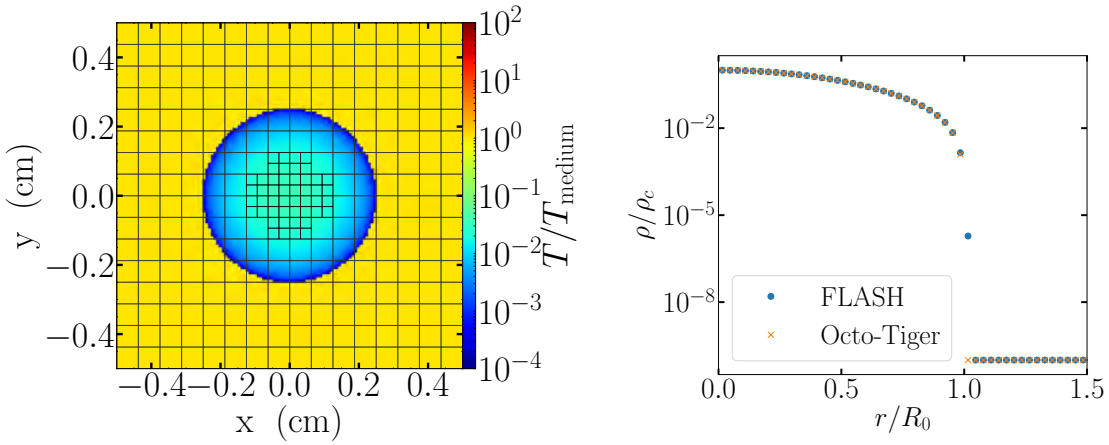
We ran six simulations in total, two OCTO-TIGER uniform grid simulations with resolutions of 128^3 and 256^3 cells and one OCTO-TIGER AMR simulation with base resolution of 128^3 and one level of refinement that increases linearly the resolution by a factor of 2. We refine based on a criterion of density (see Section 2.4). These three simulations use the OCTO-TIGER FMM gravity solver, with an opening angle parameter $\theta = 0.5$. We also ran a fourth uniform grid OCTO-TIGER simulation with resolution 128^3, and $\theta = 0.35$.

These four simulations used an ideal gas EoS. The fifth simulation, a uniform grid of 128^3 resolution and $\theta = 0.5$, used a polytropic EoS (see Equation 14). The sixth simulation is carried-

model	M ($M_\odot$)	R	ρ_c (g cm^{-3})	p_c (dyne/cm^2)	T_c (K)	$c_{s,c}$ (km/s)	T_s (K)	$c_{s,s}$ (km/s)	$T_{\rm medium}$ (K)	$c_{s,\rm medium}$ (km/s)	$T_{\rm f}$ (sec/min)	K (erg cm^2/g$^{5/3}$)
MS	0.27	$0.25R_\odot$	150	1.7×10^{17}	1.8×10^7	430	6300	9.3	1.9×10^8	2300	24 min	4×10^{13}
WD	0.35	10^9 cm	10^6	2.5×10^{22}	6.1×10^8	2000	7.3×10^4	44	4.3×10^9	11 000	18 sec	2.5×10^{12}

Legend: MS = main sequence; WD = white dwarf; M = mass; R = radius; ρ_c = central density; p_c = central pressure; T_c = central temperature; $c_{s,c}$ = central sound speed; T_s = surface temperature; $c_{\rm s,s}$ = surface sound speed; $T_{\rm medium}$ = medium temperature; $c_{\rm s,medium}$ = medium sound speed; $T_{\rm f}$ = fundamental period; K = polytropic constant.

Table 3. Examples of how our polytope model of Section 3.5 to 3.7 can be scaled to represent stars of different types. We assume $\mu_{\rm He^{+2}} = 4/3$, and $\mu_{\rm CO} = 2$, $\mu_{\rm H} = 1$, $\mu_{\rm H^+} = 1/2$, and for calculating the temperature for the main sequence star center, WD centre, main sequence star surface, and WD surface, respectively. To calculate the medium's temperature we assume $\mu_{\rm H^+} = 1/2$. Note that T_f is the fundamental period, while T_s, T_c and $T_{\rm medium}$ are temperatures



(a) AMR grid

(b) Density profile

Figure 6. Stationary star initial state. (a) AMR grid in a temperature slice at the equatorial plane of the AMR OCTO-TIGER simulation. Each cube represents a sub-grid containing 8^3 equal volume cubic cells; (b) the initial density profile of the stars in OCTO-TIGER and FLASH. Small differences are present due to different interpolation schemes

out with FLASH on a uniform grid resolution of 128^3, using the BH-tree gravity solver.

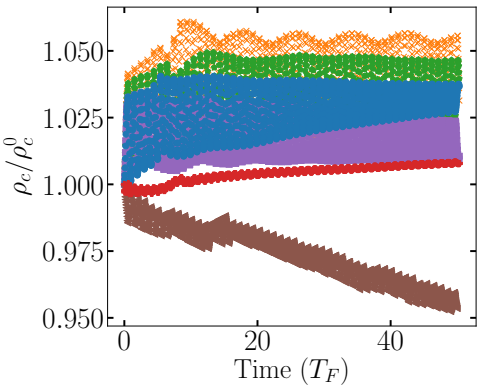
In OCTO-TIGER we solve for each cell in the domain of the Lane-Emden equation to obtain the polytrope, while in FLASH we interpolated a one-dimensional polytropic solution into the three-dimensional grid. This results in small differences between the initial states of the OCTO-TIGER and FLASH simulations. In Figure 6(a), we show the AMR grid in a temperature slice at the equatorial plane of the AMR OCTO-TIGER simulation. Each cube represents a subgrid containing 8^3 equal volume cubic cells. As we refined by the density criterion, only the inner region of the star is refined to the maximum level. In Figure 6(b), we plot the initial density profile of the stars in both OCTO-TIGER and FLASH.

In Figure 7(a), we plot the central density of the star, divided by the initial central density, over time (in units of T_F, the fundamental pulsation period of our polytope). As expected, in all simulations, the central density of the star shows clear oscillations. In the OCTO-TIGER simulations, the central density oscillates around some converging value, while in FLASH, there is a slow decrease with time overlaid on the central density oscillations. By setting a lower value for the opening angle parameter of the gravity solver ($\theta = 0.35$, red crosses in Figure 7), we can lower the amplitude for these oscillations.

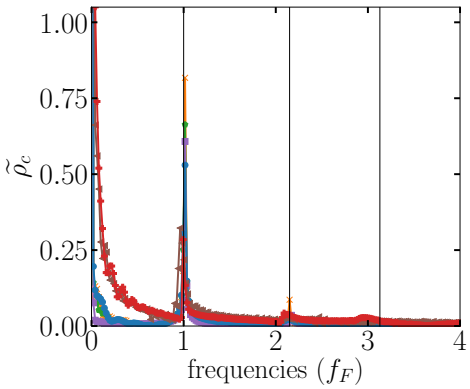
Although the oscillation amplitude might affect the amount of noise in the solution, the clear-cut test is whether the oscillation frequencies are aligned with the fundamental frequency predicted by theory. The assumption is that both the initial state and the numerical solving scheme inaccuracies will induce random noise which will oscillate at the fundamental frequency of the polytrope. For that, we plot the Fourier transform of the stars' central density (Figure 7(b)), normalised by the initial pulsation amplitude, in units of the fundamental frequency, f_F. The vertical lines in Figure 7(b) show the fundamental frequency and the first and second harmonic frequencies, $2.15 f_F$ and $3.13 f_F$, respectively. The Fourier transform of the densities of all the simulations notably peak close to the fundamental frequency. Again, the star, that is evolved with a $\theta = 0.35$, reproduced most accurately the fundamental mode, with a frequency peak that deviates by only 0.5 percent from the theoretical fundamental frequency. The other OCTO-TIGER simulations deviate by 1.5 percent, while FLASH deviates by 3.8 percent. The star that is evolved with a polytropic EoS, naturally suffers least from low-frequency noise.

We next plot (Figure 7(c)), the deviation from conservation of mass, $\Delta M/M_0$, in the OCTO-TIGER simulations. $\Delta M = M(t) + M_{\rm out} - M_0$, where $M(t) = \int_V \rho dV$, $M_{\rm out}$ and M_0 are the mass inside the simulation domain, the mass that outflowed from the grid and the initial mass, respectively. OCTO-TIGER calculates automatically, for every time step, the outflow of quantities such as mass, energy, and entropy, which simplify following of every small change of those quantities. Therefore, we plot only the conservation of mass in OCTO-TIGER simulations. We see conservation at the 10^{-13} level up to a time when the density flooring adds mass to the grid. Even then the conservation, by the end of the simulation is still at the level of a few $\times 10^{-12}$.

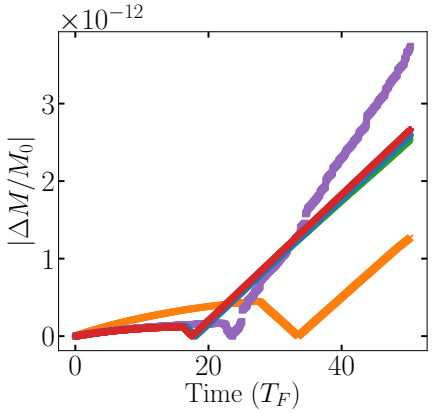
In these simulations, we set a density floor which causes a deviation from conservation at a machine precision level. The floor



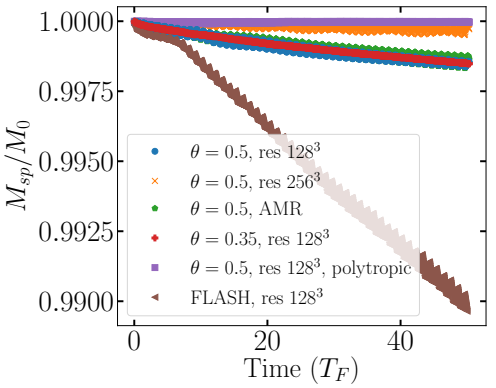
(a) Central density



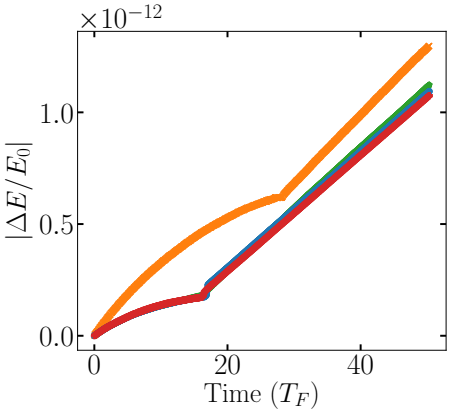
(b) Fourier transform of the central density



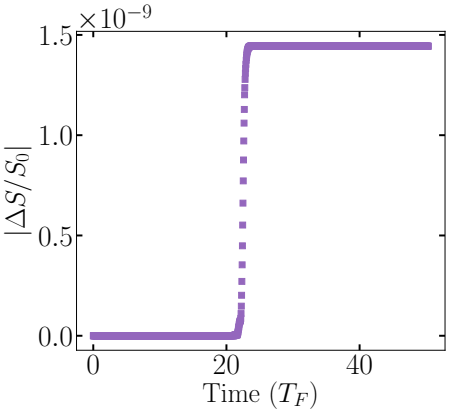
(c) Total mass conservation



(d) Mass within the polytrope boundary



(e) Total energy conservation



(f) Total entropy conservation

Figure 7. The stationary polytrope benchmark test. (a) Central density over time divided by the initial central density. (b) The Fourier transform of the central density pulsations normalised by the initial pulsation amplitude. The vertical lines show the fundamental frequency and the first and second harmonic frequencies. (c) Conservation of mass (only for OCTO-TIGER simulations). (d) Total mass enclosed in a sphere with a radius of the initial polytropic radius. (e) Conservation of energy (only for OCTO-TIGER simulations with ideal gas EoS). (f) Conservation of entropy (only for the polytropic EoS OCTO-TIGER simulation). T_F is the fundamental period of a polytrope with $n = 3/2$ and $\gamma = 5/3$

density is 5 orders of magnitude less than the initial density of the ambient medium, and usually will not affect mass conservation. However, at machine precision level, a deviation that grows approximately linearly with time appears. This can be explained by considering the following. As a consequence of ambient medium gas falling onto the star surface, the density outside the star decreases with time. Eventually, the density will decrease below the density floor threshold at the outer regions, which is then filled with density at the floor value. This happens after approximately 20, 25, and 35 fundamental periods in the low-resolution atmosphere with ideal gas EoS simulations, polytropic EoS simulation, and the high-resolution simulation, respectively. If the region that is being filled with a floor density is a fraction α of the domain volume, we will get a deviation from mass conservation of roughly $\alpha V \rho_{\rm floor}/M_0 = \alpha \rho_{\rm floor}/\overline{\rho}_0$, where $\overline{\rho}_0$ is the initial mean density in the simulation's domain. In our simulations $\rho_{\rm floor}/\overline{\rho}_0 \simeq 9 \times 10^{-14}$. If only one layer of cells at the grid boundary is being filled with the floor density then $\alpha \simeq 0.05$. If, in addition, the cells will be refilled every 5 time steps, after $\sim$20 000 time steps a deviation of 2.2×10^{-11} from conservation of mass will occur. In the simulations themselves, fewer cells are refilled less frequently so the deviations are smaller. Additionally, as shock heating is eliminated in the polytropic EoS simulation, the ambient gas continuously falls onto the star's surface without disturbance, which results in a bigger volume of flooring, and in a somewhat higher deviation from conservation of mass (purple line).

In Figure 7(d), we show the total mass enclosed in a sphere with a radius of the initial star's radius. Here we see that FLASH loses 1 percent of the stellar mass in 40 pulsations, while the OCTO-TIGER simulations lose at most 0.2 percent of the mass flowing out. The lower resolution, polytropic EoS as well as the higher resolution simulations perform best.

In Figure 7(e), we show the deviation from conservation of energy $\Delta E/E_0$, where $\Delta E = E\left(t\right) + E_{\rm out} - E_0$, and $E\left(t\right) = \int_V \left(E + \frac{1}{2}\rho\phi\right) dV$, $E_{\rm out}$ and E_0 are the energy inside the simulation domain, the energy that outflows from the grid and the initial energy, respectively. In Figure 7(f), we show the deviation from conservation of entropy $\Delta S/S_0$, where $\Delta S = S\left(t\right) + S_{\rm out} - S_0$, and $S\left(t\right) = \int_V \tau dV$, $S_{\rm out}$, and S_0 are the entropy inside the simulation domain, the entropy that outflows from the grid and the initial entropy, respectively. Beside the small effect of the flooring, the ideal gas EoS simulations conserve energy at machine precision, while the polytropic EoS simulation conserves entropy at a machine precision.
Click on the link[1] to view a movie of the OCTO-TIGER, 256^3, $\theta = 0.5$ simulation.

3.6 Star Moving Linearly in the Grid

We next simulate a star as in Section 3.5 except we initialize the star with a non-zero bulk velocity, allowing it to translate at a given constant velocity through the grid. This exercise tests the degree to which the surface layers of our structure shear away due to interaction with the low density medium permeating the background. It also tests how well a spherical star moves through our cubical grid and how well our code conserves the total, non-zero linear momentum.

We test three velocities, (a) a subsonic velocity, $v_1 = 5.2 \times 10^{-4} c_{s,\rm medium}$, where $c_{s,\rm medium}$ is the speed of sound of the ambient hot medium (see Table 3); (b) an intermediate velocity, $v_2 = 5.2 \times 10^{-2} c_{s,\rm medium}$; and (c) a high, supersonic velocity, $v_3 = 5.2 c_{s,\rm medium}$. We list these three velocities scaled to a main sequence and WD models, as well as the simulation time (shorter for the faster stars) and the distance the star travels in Table 4. The distance the star travels equals to 1.1 times the star's initial radius for all simulations. We placed the star initially at (-0.125, -0.125, 0) and gave it a velocity in the direction towards grid position (1,1,0).

Each velocity regime contains a set of six simulations (as we have done for the stationary star): two uniform grid, 128^3 and 256^3 cell simulations, one AMR simulation (128^3 cells and one level of refinement), with ideal gas EoS and $\theta = 0.5$; a uniform grid 128^3 simulation with ideal gas EoS and $\theta = 0.35$; a uniform grid 128^3 simulation with polytropic EoS and $\theta = 0.5$; and a FLASH uniform grid 128^3 simulation. In Sections 3.6.1 – 3.6.3, we describe the results of the simulations.

3.6.1 Star Translating at Low Velocity

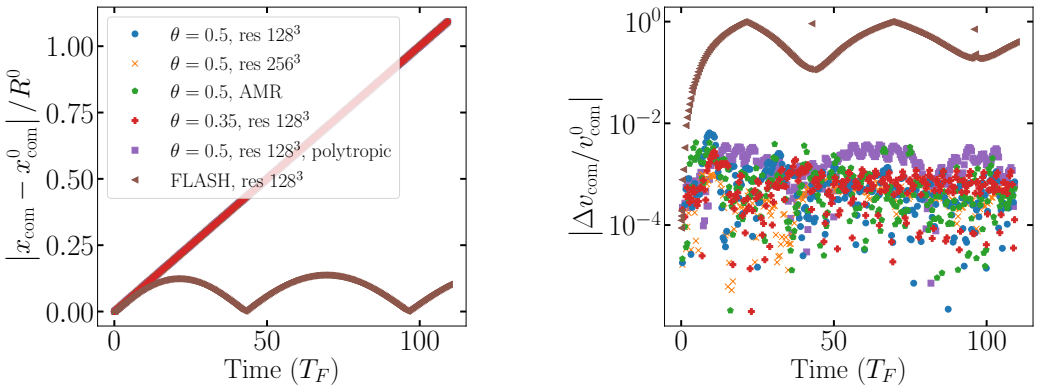
In this regime, the star translates at a very low velocity, equivalent to a Mach number of 5.2×10^{-4}, not only with respect to the ambient medium speed of sound, but also with respect to the speed of sound of the coldest regions at the star's surface. Such low bulk velocities are a challenge for hydrodynamic codes. Low level subsonic noise that develops can advect the stellar momentum to the surrounding low density gas, diminishing the star's initial low velocity. In addition, the star's low velocity allows us to run this test for over an hundred fundamental pulsations periods, the longest time we have run a polytrope simulation. During this time, the star should keep pulsating at its Eigenmodes as it translates through the grid.

The star in the OCTO-TIGER simulations slowly moves from the bottom left corner to the upper right corner (in the xy plane). As in the stationary star simulations, the star remains in hydrostatic equilibrium, while the ambient medium is free falling onto the stellar surface. This slowly cools the ambient medium and creates shocks on the star's surface that heat the gas for ideal gas simulations. We performed the same analysis as we did for the stationary star. We find that the star's movement does not affect the star's pulsations. Almost identically to the stationary star in Figure 7(a), the central density oscillates at the fundamental frequency in all OCTO-TIGER simulations, while the mean central density of the star simulated with FLASH decreases with time. Additionally, the star's mass ($M_{\rm sp}$) remarkably decreases at the same rate as for the stationary star in Figure 7(d), decreasing by 2 percent in the FLASH simulation and by less than 0.4 percent with OCTO-TIGER, after 100 pulsations.

In Figure 8(a), we plot the centre of mass position as a function of time, while in panel (b) we show the deviation of the centre of mass velocity from its initial value. The star in the OCTO-TIGER simulations moves through the grid at a constant velocity, while in the FLASH simulations, the star slows down, starts moving in the opposite direction and then oscillates around the star's initial position. The OCTO-TIGER simulations deviate from the initial velocity by less than 1 percent throughout the entire evolution.

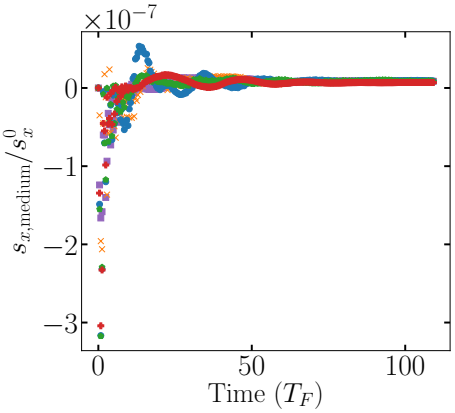
We exploited OCTO-TIGER's capability to track the provenance of gas, to calculate the diffusion of linear momentum from the star to its environment. In Figure 8(c), we plot the x-momentum of the diffuse medium divided by the initial *total* x-momentum in the OCTO-TIGER simulation. The initial oscillation is due to the sharp density gradient at the star's surface. Soon after the value settles at values close to zero. We mention though, that the mass ratio

[1] https://youtu.be/4-ra6fY982Q

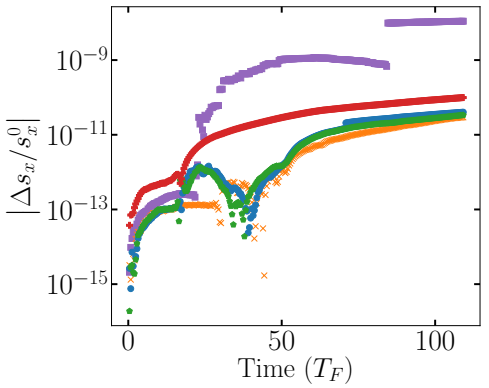


(a) Centre of mass position accuracy

(b) Centre of mass velocity accuracy



(c) Ambient medium total x-linear momentum



(d) Total x-momentum conservation

Figure 8. The translating star benchmark test with the star moving at Mach 5.2×10^{-4} (with respect to the diffuse medium). We plot various physical quantities of interest over time. Time is in units of T_F, the fundamental pulsation period

Model	v_1 (km s^{-1})	v_2 (km s^{-1})	v_3 (km s^{-1})	$t_{\rm sim1}$	$t_{\rm sim2}$ (sec/min/hr)	$t_{\rm sim3}$	$R_{\rm travel}$ ($R_\odot$)
MS	1.20	120	12 000	44 hr	27 min	16 sec	0.28
WD	5.6	560	56 000	33 min	20 sec	0.2 sec	0.016

Table 4. Translating polytrope benchmark test. Scaled quantities: motion velocities, total simulation times and displacement. For other scaled quantities refer to Table 3

between the ambient medium gas and the star is of the order of 10^{-8}. No comparison with FLASH is possible due to FLASH not tagging gas provenance.

We find, similarly to the stationary star test, that mass, energy, and entropy are conserved in the OCTO-TIGER simulations to excellent precision, except for small deviations due to adding mass when a cell's density drops below the minimum floor value (see Sec. 2.6). We additionally follow the conservation of x-linear momentum (Figure 8(d)), finding a machine precision level until flooring starts operating at which point we observe a slow linear growth. The deviation, though, is minimal, less than 10^{-10} after 100 pulsation periods for all simulations except the polytropic EoS. The polytropic simulations suffers more from flooring (because shock heating is absent) and the deviation settles on a value of 8×10^{-9}.

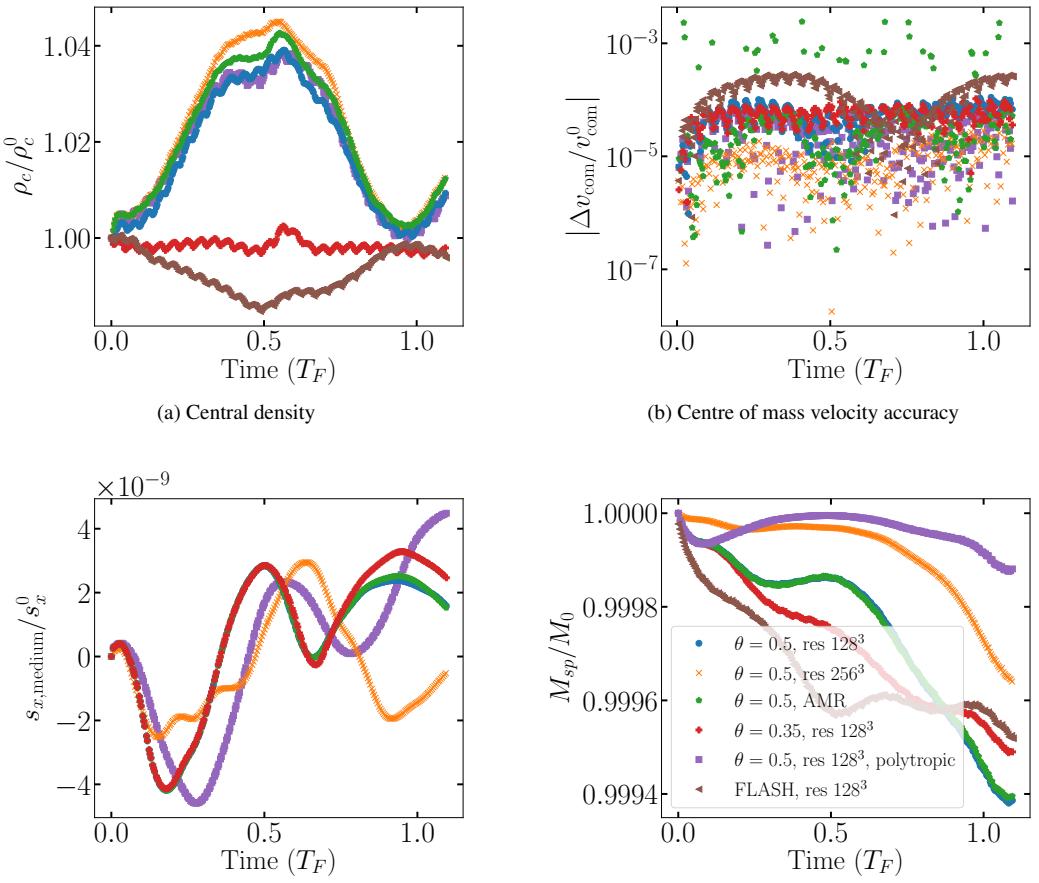
Click on the link[2] to view a movie of the low velocity translating star.

3.6.2 *Star Translating at Intermediate Velocity*

With a speed of Mach 5.2×10^{-2} with respect to the outside medium, the star translates at a subsonic velocity, but at a supersonic speed with respect to the star's surface. The simulation time is comparable to the star's fundamental period. This allows the star to pulsate only once while it translates at a higher speed than the previous test.

The star's gas diffuses out in the wake of the star, but then

[2] https://youtu.be/8ArZqP9F93Y



(a) Central density

(b) Centre of mass velocity accuracy

(c) Ambient medium total x-linear momentum

(d) Mass within polytrope boundary

Figure 9. The translating star benchmark test with the star moving at Mach 5.2×10^{-2} (with respect to the diffuse medium). We plot various physical quantities of interest over time. Time is in units of T_F, the fundamental pulsation period

reverses motion and trails the star moving in a flow that resembles accretion. Shocks are apparent at this wake in the ideal EoS which increases the diffusing out of material from the star.

In Figure 9, we plot only the salient quantities that demonstrate the level of accuracy of the simulations. In panel (a) we demonstrate how the simulation time is approximately the time of the fundamental period of pulsation and the amplitude is similar to the stationary star simulations of Figure 7(a). The error on the star's velocity is less than 10^{-3} for *all* simulations (Figure 9(b)), where OCTO-TIGER simulations have an error that is about 10 times smaller than the OCTO-TIGER simulations in the case of the slow moving star of Figure 8(b) (Section 3.6.1). In Figure 9(c) we see that the ambient medium momentum in OCTO-TIGER simulations remains negligible. After an oscillatory period, we expect it to converge to a value that is of the order of 10^{-9}, which is over 10 times smaller than for the equivalent, slow moving test. Finally, in Figure 8(d) the mass in the original polytrope is also retained with a precision that is 10 times that of the slow moving polytrope. An amount that is relatively high if we consider the short simulation time.

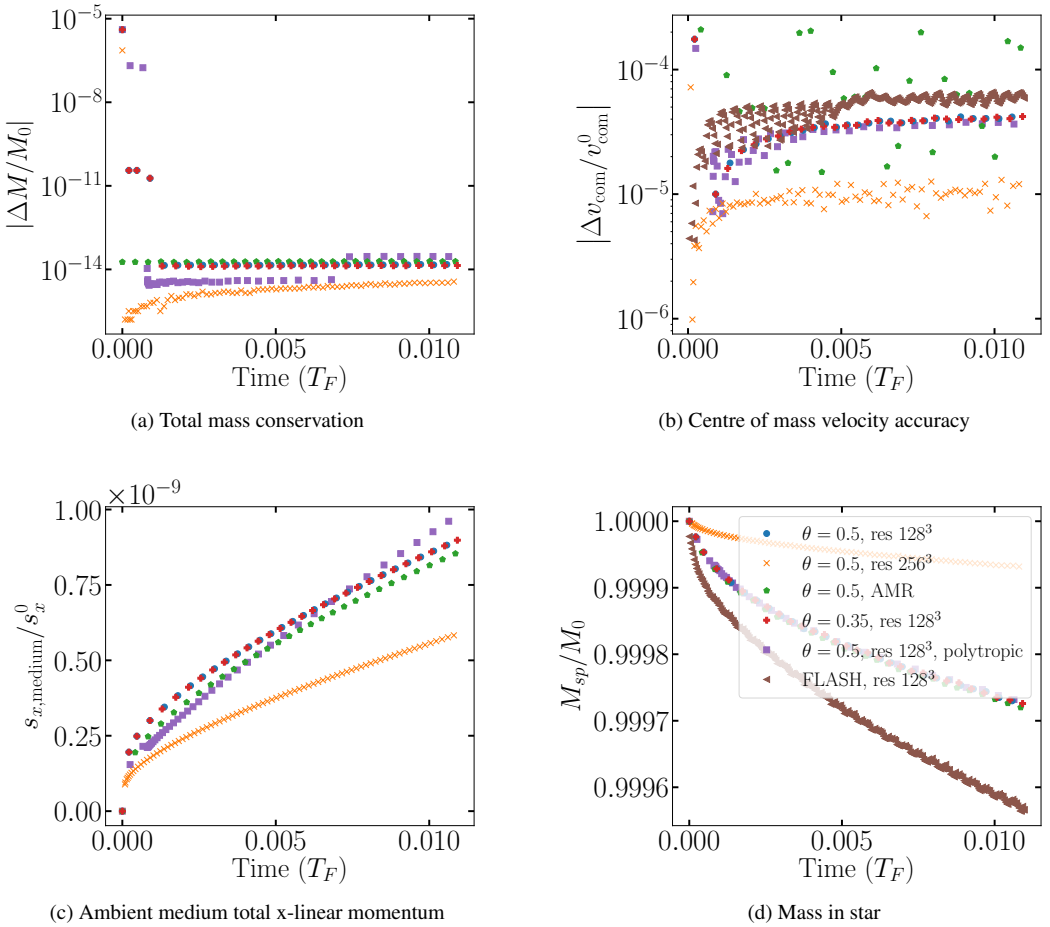
Click on the link[3] to view a movie of the intermediate velocity translating star.

3.6.3 *Star Translating at a High Velocity*

For the highest velocity simulation, at Mach 5.2 with respect to the sound speed of the ambient medium, the running time is shorter than the dynamical time of the star so the star does not relax. There is a slight overall mass increase in the two unigrid simulations, but not for the AMR one (Figure 10(a)) between the first and fifth time steps in the simulation. This is due to more mass leaving a cell than the mass in that cell at the sharp stellar edges of the polytrope. Since the mass in a cell is not allowed to be zero nor negative, mass is added to reach the density floor value, hence introducing mass in the grid. At subsequent time steps when numerical diffusion has softened all edges, this does not happen. This is not a large problem and can be resolved with a reduction of the Courant time to values ~ 0.14 (see justification in Section 2.7).

In Figure 10(b-d) we see that the velocity error, ambient x-momentum and mass in sphere retain their values to a precision that is comparable to that of the polytrope moving at intermediate

[3] https://youtu.be/b3MMyDCPY60



(a) Total mass conservation

(b) Centre of mass velocity accuracy

(c) Ambient medium total x-linear momentum

(d) Mass in star

Figure 10. The translating star benchmark test with the star moving at Mach 5.2 (with respect to the diffuse medium). We plot various physical quantities of interest over time. Time is in units of T_F, the fundamental pulsation period

speed (Section 3.6.2). The best simulation is the high resolution unigrid with $\theta = 0.5$ (yellow line), something that is also true but to a lesser extent for the other two velocity tests.
Click on the link[4] to view a movie of the high velocity translating star.

3.7 Star Rotating in the Grid

Here we simulate a rotating star in equilibrium with OCTO-TIGER and FLASH. Its rotation profiles, $\Omega(r)$, should remain flat whether the simulation is run in the inertial frame or in a rotating frame of reference, over a certain amount of time.

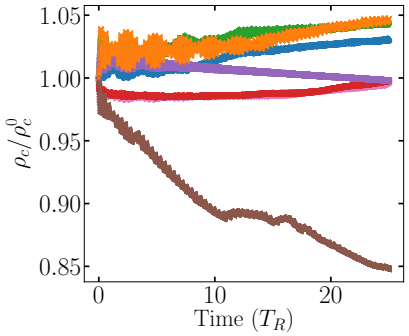
We constructed an equilibrium profile of a rotating polytrope using the Self Consistent Field method that will be described in Section 4.1. We then interpolated the profile in the OCTO-TIGER and FLASH grids using the same method. The fast-spinning star has an oblate shape with a ratio of 2/3 between polar and equatorial radii. The angular velocity is 0.52 in code units (0.0023 rad s^{-1} and 0.19 rad s^{-1} for the MS and the WD models, respectively), and the rotational period is $\approx$ 12 code units (45 min and 32.3 sec for the MS and the WD models, respectively).

We carry out simulations in the inertial frame (both with OCTO-TIGER and FLASH) and in the rotating frame (only with OCTO-TIGER). For each of these frames of reference, we carry out six simulations with the same combinations of resolution, θ parameter and EoS as done for the stationary star in Section 3.5.

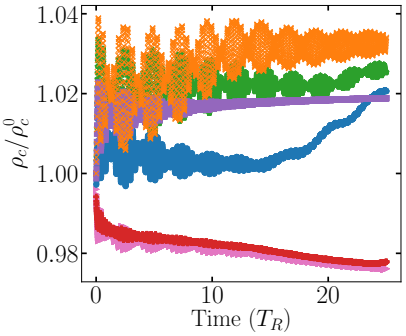
In Figures 11(a) and (b) we plot the familiar evolution of the core density scaled to the initial value. With an exception of the FLASH simulation that has a declining trend, reaching 85 percent of its initial values over 25 rotation periods, all the OCTO-TIGER simulations retain the initial core density value within 5 percent with no noticeable difference between rotating and inertial frame. In Figure 11(c), we plot the diffusion of the stellar mass out of the original boundary in the inertial frame. Once again FLASH performs worse with a loss of 0.5 percent of the mass over 25 rotation periods, but most of the OCTO-TIGER profiles are a close second with the polytropic EoS simulation performing distinctly better. Comparing this with panel (d) we see how all OCTO-TIGER simulations perform much better in the rotating frame, losing at most 0.05 percent of the mass. Similarly, Figure 11(e) shows that in the inertial frame, the high density inner star diffuses out moving to twice its original radius. The polytropic EoS simulation is by far the best, with FLASH also performing well. In this case the rotating frame, panel (f), does not perform a great deal better than the inertial frame.

The acid test is the evolution of the rotation profile. All simu-

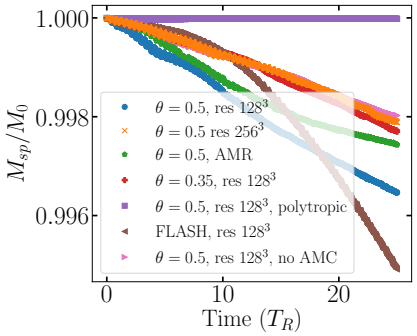
[4] https://youtu.be/rZiLMJ6Cxc4



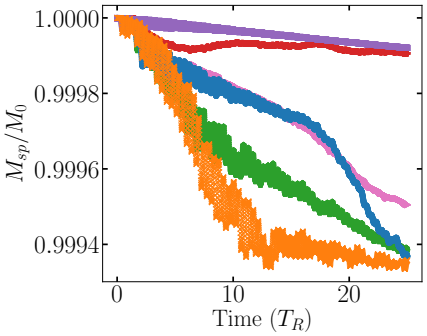
(a) Central density, inertial frame



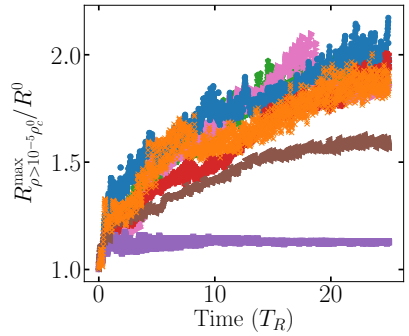
(b) Central density, rotating frame



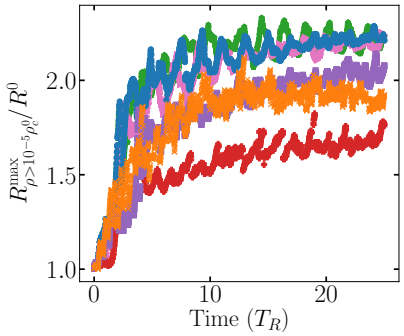
(c) Mass in star, inertial frame



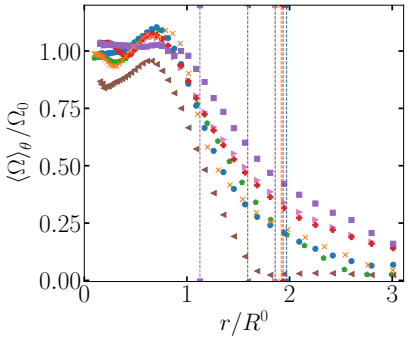
(d) Mass in star, rotating frame



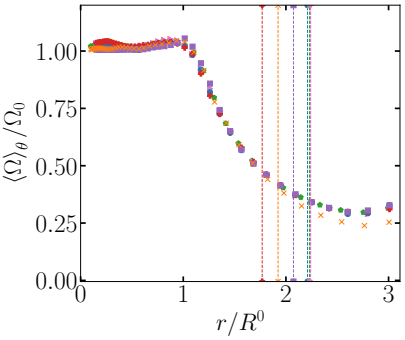
(e) Diffusion of high density gas, inertial frame



(f) Diffusion of high density gas, rotating frame

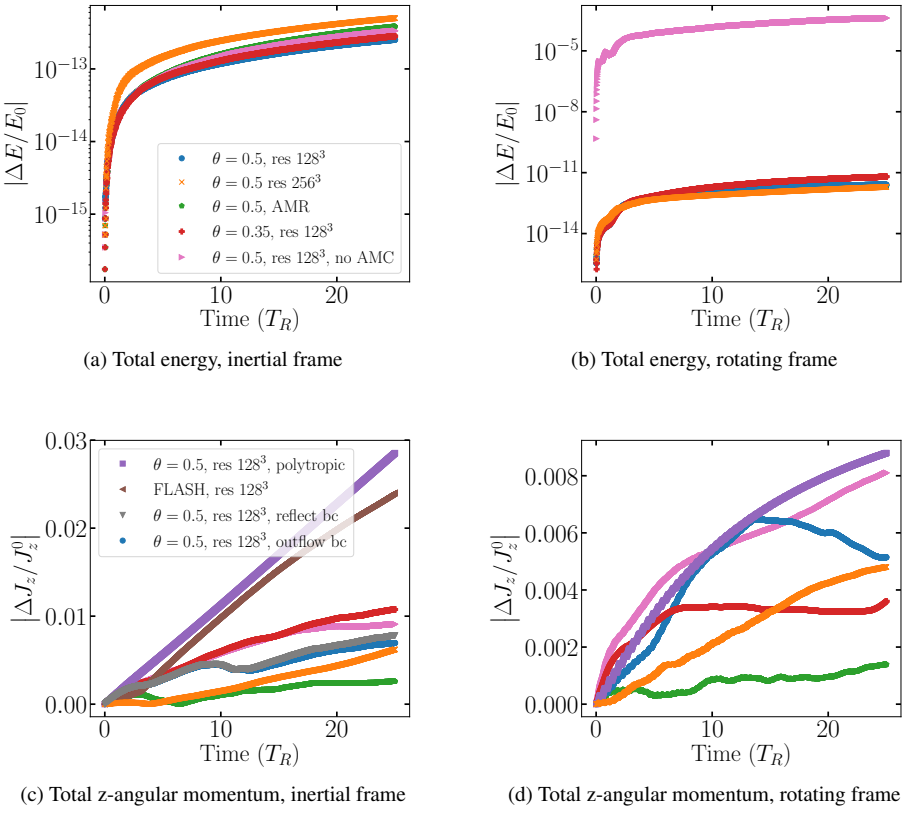


(g) Final rotation profile, inertial frame



(h) Final rotation profile, rotating frame

Figure 11. The rotating star test with $\Omega_0 = 0.52$ code units (0.0023 rad s^{-1} and 0.19 rad s^{-1} for the MS and the WD models, respectively). Several quantities are compared. The vertical lines in panels (g) and (h) are the maximum distance of high density gas at the end of the simulation (rightmost values of the curves in panel e and f, respectively)



(a) Total energy, inertial frame

(b) Total energy, rotating frame

(c) Total z-angular momentum, inertial frame

(d) Total z-angular momentum, rotating frame

Figure 12. The rotating star test with $\Omega_0 = 0.52$ code units (0.0023 rad/s and 0.19 rad/s for the MS and the WD models, respectively). Conservation of energy and angular momentum. All simulations used an ideal gas equation of state except the simulation shown in purple squares, which used a polytropic equation of state

lations in the inertial frame (Figure 11(g)) do not retain a flat profile (with the notable exception of the polytropic EoS one), while those in the rotating frame (Figure 11 (h)) systematically do. However, in the inertial frame, by the end of the simulations, the spin of the gas that has diffused out of the original volume is distinctly less for all simulations than in the rotating frame. The FLASH simulation (in the inertial frame) does not retain a flat profile, but it has the least diffusion of gas out of the boundary of the sphere of all simulations.

Finally, in Figure 12 we show the degree of conservation we achieve in the OCTO-TIGER simulations. This highlights the importance of including the angular momentum correction (AMC) of the gravity solver for the conservation of energy (upper panels) as outlined in Section 3. In the lower panels we plot the deviation from angular momentum conservation $\Delta J_z/J_z^0$, where $\Delta J_z = J_z(t) - J_z^0$, and $J_z(t) = \int_V (xs_y - ys_x)\, dV$ and J_z^0 are the z-angular momentum inside the simulation domain, and the initial z-angular momentum, respectively, and where s_x and s_y are the x and y components of the inertial momenta, respectively. As we do not take into account outflows we can compare between OCTO-TIGER and FLASH, showing OCTO-TIGER outperforms FLASH in ideal gas simulations. Additionally, by comparing between simulations with outflow (blue circles) and reflect (gray downward pointing triangles) boundary conditions, we find that outflows only very slightly affect angular momentum conservation. Overall, the AMR simulation conserves angular momentum the best, and simulations that include AMC conserve angular momentum better than the simulation without this correction.

In conclusion this test gives an excellent idea of the type of numerical diffusion we can expect. The rotating frame outperforms the inertial frame, and the polytropic EoS seems to be doing best in almost all cases. FLASH performs the worst except in retaining a stable high density sphere inside the star and it can also be seen that a higher gravity accuracy ($\theta = 0.35$) improves the performance. Click on the link[5] to view a movie of the rotating star in the rotating frame.

4 BINARY MERGER BETWEEN TWO POLYTROPES WITH A MASS RATIO OF 0.5

Here we present the results of two binary merger simulations performed in the rotating frame, starting from identical initial conditions, but differing in the adopted EoS. The donor star is half the mass of the accretor and initially fills its Roche lobe. Both stellar spins are synchronized to the orbital frequency. In the first simulation, the two stars are constructed using the same polytropic EoS ($\epsilon_1 = \epsilon_2 = 1$), with polytropic index $n = \frac{3}{2}$. In the second simulation, we use an ideal gas EoS (with dual energy parameters $\epsilon_1 = 0.001$ and $\epsilon_2 = 0.1$ from Equations 11 and 13).

We are motivated to model such a system for two reasons. First, by appropriate choice of units, this system approximates a

[5] https://youtu.be/7ZrxGJW2J_Y

binary consisting of two low mass WDs. We have set the unit conversion factors so that the accretor has a mass of $0.6\ \mathrm{M}_\odot$, the donor a mass of $0.3\ \mathrm{M}_\odot$ (1 code mass unit is 1 solar mass). The computational domain is 40 code length units on a side, or 1.7×10^{11} cm (1 length units is 4.36×10^9 cm). With this scaling the separation between the two stars is $6 \times 10^{-2}\ \mathrm{R}_\odot$, and the period is 2.7 minutes. The second reason is that a similar system was modelled by Motl et al. (2017) providing us with a comparison simulation as a further verification of OCTO-TIGER.

We use 9 levels of refinement, with the highest level having a grid spacing of 4.2×10^7 cm. The accretor is 50 cells across along its longest axis (the spinning stars have slightly larger equatorial than polar radii) and the donor is 79 cells.

Following Motl et al. (2017), we initially drive the stars into deep contact by systematically removing angular momentum. This is accomplished by adding source terms to the evolution equations for the x and y components of the momentum (Equation 2),

$$\begin{aligned} S_{s_x,\mathrm{driving}} &= -\tfrac{y}{x^2+y^2}\left(xu_y - yu_x\right) f_{\mathrm{driving}} \\ S_{s_y,\mathrm{driving}} &= +\tfrac{x}{x^2+y^2}\left(xu_y - yu_x\right) f_{\mathrm{driving}} \end{aligned} \tag{44}$$

where x and y are the x and y components of the position vector $\mathbf{x}$, u_x and u_y are the x and y component of the inertial frame velocity, $\mathbf{u}$, and f_{driving} is the driving rate in units of inverse time. This has the effect of reducing the z-angular momentum component of a given cell at a logarithmic rate of f_{driving}. It does not affect the cylindrical radial or z linear momenta. Similarly to what was done by Motl et al. (2017), we drive the system together at a rate of 1 percent per orbit for the first 2.7 orbits.

4.1 The Initial Stellar Model

The initial conditions for our binary simulations were produced using our own variant of the "Self Consistent Field" (SCF) method (Hachisu (1986a,b); Even & Tohline (2009); Kadam et al. (2016)). Given a barotropic EoS, $p = p(\rho)$, the SCF method allows one to construct an equilibrium model of a binary system with synchronously rotating stars. OCTO-TIGER's SCF can be used to construct systems with a polytropic or bi-polytropic EoS or a cold white dwarf EoS. The user selects the masses of each star and the Roche lobe filling factor for the donor. When each star has a different EoS, the Roche lobe filling factor for the accretor must also be specified. Here we document OCTO-TIGER's SCF as it pertains to the construction of the initial conditions for the binary models in this paper. These models use the same structural polytropic EoS for each star. One of the models is evolved with the same polytropic EoS, and the other with an ideal gas EoS.

The effective potential is defined as

$$\phi_{\mathrm{eff}} = \phi - \frac{1}{2}\Omega_{\mathrm{orb}} r^2, \tag{45}$$

where ϕ is the gravitational potential, Ω_{orb} is the rotational frequency of the binary, and r is the distance to the axis of rotation. Using the effective potential, the hydrostatic equilibrium equation in the rotating frame can be written

$$\frac{1}{\rho}\boldsymbol{\nabla} p + \phi_{\mathrm{eff}} = 0. \tag{46}$$

For the EoS we use:

$$p_{\mathrm{poly}} = K\rho^{1+\frac{1}{n}}, \tag{47}$$

where K is the polytropic constant and n is the polytropic index. Combining Equations 46 and 47 and integrating we arrive at:

$$K\left(1+n\right)\rho^{\frac{1}{n}} + \phi_{\mathrm{eff}} = C_{1|2}, \tag{48}$$

where the constant on the RHS is either C_1 for the primary, accreting star or C_2 for the secondary, donor star. While in general the polytropic constant, K, may be different for each star, in this paper, they are the same.

The SCF solves for the initial conditions iteratively. The user selects the mass of each star, the polytropic index, the initial separation, and the Roche filling factor of the donor. (Note that while the user specifies the initial separation, it is the orbital angular momentum that is held constant.) The Roche filling factor is defined as

$$f = \frac{\phi_{\mathrm{edge}} - \phi_C}{\phi_{L1} - \phi_C}, \tag{49}$$

where ϕ_{edge} is the effective potential at the edge of the donor ϕ_C is the effective potential at the centre of mass of the donor, and ϕ_{L1} is the effective potential at L_1, the first Lagrange point.

For the initial iteration we place two polytropic stars in the grid at the desired separation and small enough that they are within their respective Roche lobes. The gravity solver is also called before the iterations begin. For each iteration we: (1) multiply the densities of each star by a constant factor for each star, such that the result yields the desired mass of each star; (2) advect the entire grid such that the centre of mass lies at the centre of coordinates; (3) set the new value for Ω_{orb} using

$$\Omega_{\mathrm{orb}} \rightarrow \frac{J_{0,\mathrm{orb}}(M_1 + M_2)}{M_1 M_2 a^2}, \tag{50}$$

where $J_{0,\mathrm{orb}}$ is the initial orbital angular momentum, M_1 and M_2 are the fixed masses of the accretor and donor, and a is the orbital separation for the current iteration; (4) compute the integration constants C_1 and C_2,

$$C_1 = K\left(1+n\right)\rho_1^{\frac{1}{n}} + \phi_1, \tag{51}$$

and

$$C_2 = \left(1-f\right)\phi_C + f\phi_{L1}, \tag{52}$$

where ρ_1 is the maximum density of the accretor and ϕ_1 is the effective potential at the centre of the accretor; (5) compute a new value for K,

$$K = \frac{C_2\phi_2}{\left(n+1\right)\rho_2^{\frac{1}{n}}}, \tag{53}$$

where ρ_2 is the density at the centre of the donor, and ϕ_2 is the effective potential at the centre of the donor ($\phi_2 = \phi_C$); (6) compute a new density value at each point on the grid using

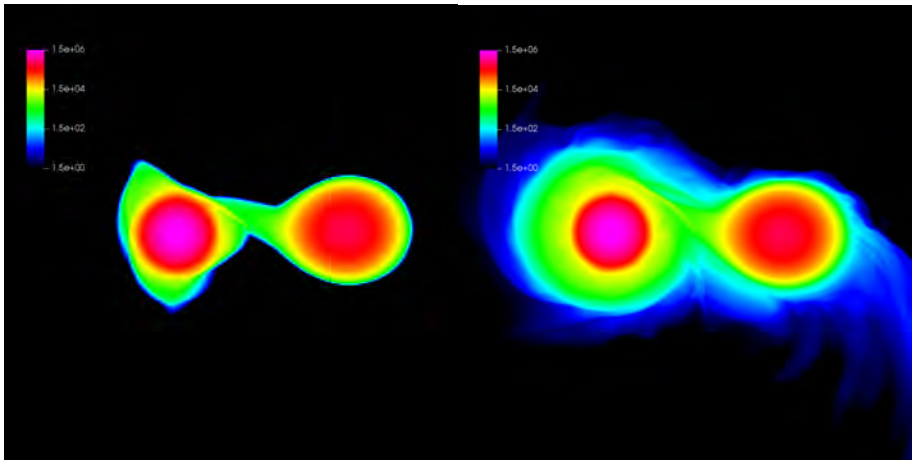
$$\rho = \left(\frac{\phi_{\mathrm{eff}} - C_{1|2}}{K\left(n+1\right)}\right)^n, \tag{54}$$

for the accretor or donor region as appropriate; (7) solve for the potential and repeat the process from step (1).

4.2 Simulation Results

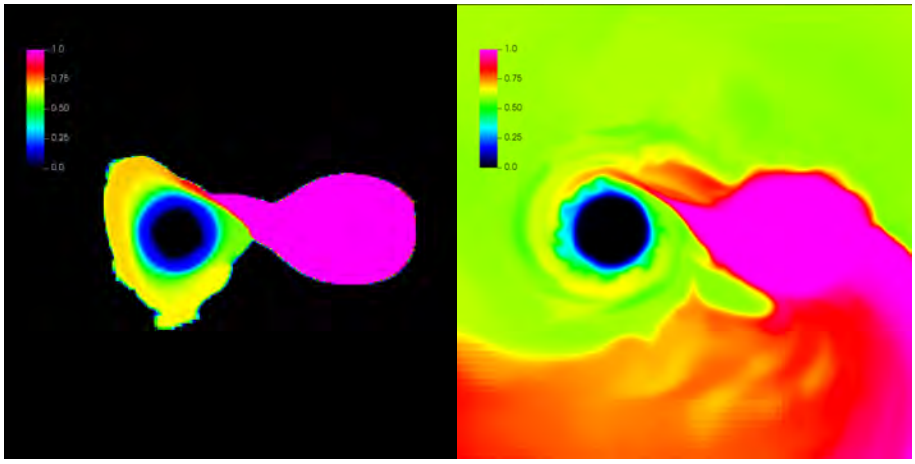
In each model, mass transfer begins within the first couple of orbits. The transfer rate grows and the donor is tidally disrupted at approximately $21.2P_0$ for the polytropic and $19.6P_0$ for the ideal gas model. Here P_0 is the initial orbital period, which is the same for each model. The merged remnant is then evolved for a couple of additional P_0.

In Figure 13, we show various images at $t = 11.7P_0$ for the polytropic simulation and $t = 12.6P_0$ for the ideal gas simulation.



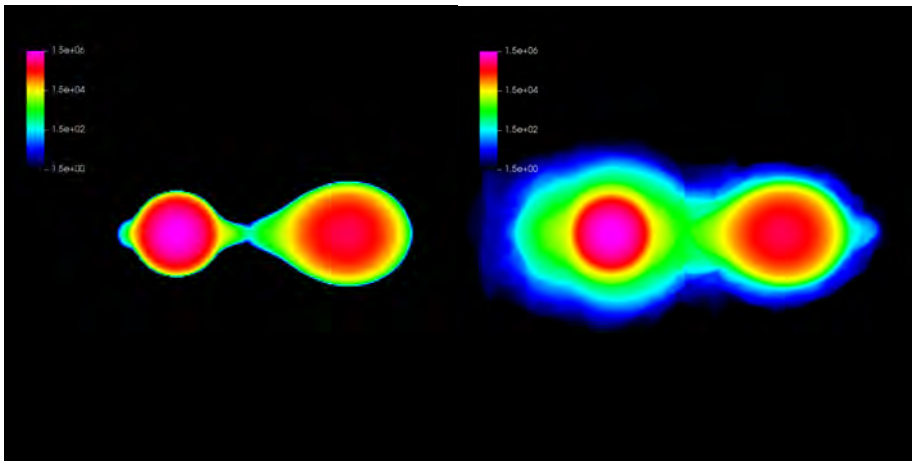
(a) Orbital density slice, polytropic EoS

(b) Orbital density slice, ideal gas EoS



(c) Orbital donor fraction slice, polytropic EoS

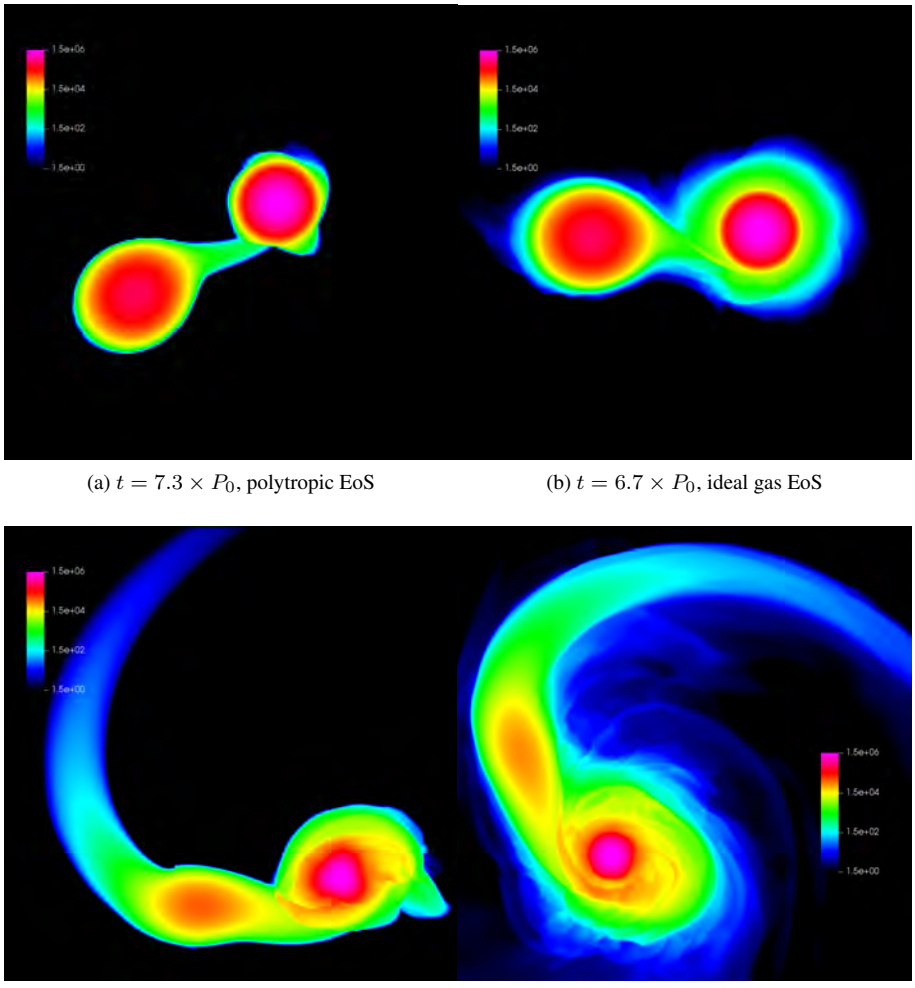
(d) Orbital donor fraction slice, ideal gas EoS



(e) Vertical density slice, polytropic EoS

(f) Vertical density slice, ideal gas EoS

Figure 13. Snapshots from our $q = 0.5$ simulations with a polytropic EoS (left column, at time $11.7 \times P_0$, where P_0 is the initial orbital period) or an ideal gas simulation (right column, at time $12.6 \times P_0$). The box size is 11×10^9 cm in all cases. The density scale is logarithmic and runs from 1.5 g/cm^3 to 1.5×10^6 g/cm^3



(a) $t = 7.3 \times P_0$, polytropic EoS (b) $t = 6.7 \times P_0$, ideal gas EoS

(c) $t = 21.2 \times P_0$, polytropic EoS (d) $t = 19.6 \times P_0$, ideal gas EoS

Figure 14. Density slices on the orbital plane for our $q = 0.5$ simulations with a polytropic EoS (left column) or an ideal gas EoS (right column). The box size is 11×10^9 cm (top row) or 16×10^9 cm (bottom row). The density scale is logarithmic and runs from 1.5 g/cm^3 to 1.5×10^6 g/cm^3. The snapshots are taken at times 7.3 (a), 6.7 (b), 21.2 (c) and $19.6 \times P_0$ (d). See Table 1 for description

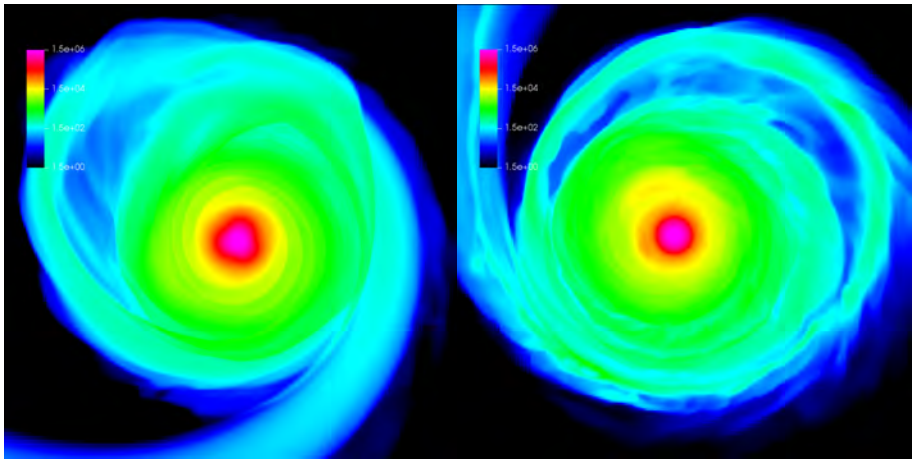
Panels (a) and (b) show density slices in the equatorial plane for the two EoS choices. OCTO-TIGER is able to track the material that was originally part of the donor, and the donor mass fraction is displayed in panels (c) and (d). The polytropic model clearly exhibits the same polygonal resonances at high mass transfer rates that were first reported in D'Souza et al. (2006) and were later confirmed in the SPH simulations reported in Motl et al. (2017). As in Motl et al. (2017), these resonance patterns do not appear in the ideal gas model because the accretion structure is thicker due to shock heating. In Figure 13(e)–(f) we show a density cut at the same times, showing that while the ideal gas EoS simulation produces L_2 and L_3 outflow, the polytropic EoS simulation only shows a hint of L_2 outflow by that time. The level of detail clearly visible near L_1 is consistent with the Roche geometry.

Figure 14(a)–(b) are from earlier in the evolution at approximately $t = 7P_0$ in each model. The resonance patterns in the polytropic model progress from pentagonal shaped, to box shaped, to triangular shaped (clearly seen in in Figure 14(a)). Figure 14(c)–(d) show the system as tidal disruption is occurring. As the donor is disrupted, significant mass flows through the L_2 Lagrange point, resulting in the "tails" seen in the figure.

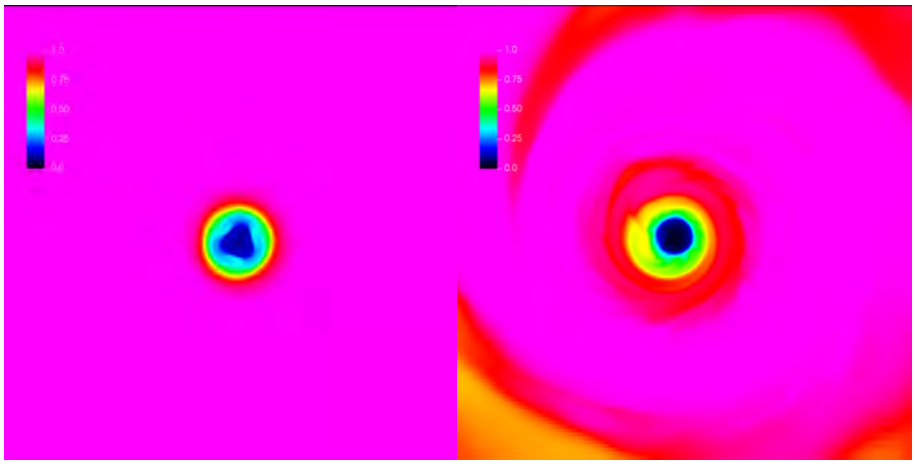
The images in Figure 15 are taken from two orbits after tidal disruption and merger has occurred. Panels (a) and (b) show equatorial density slices, while panels (c) and (d) show the donor fraction. The triangular shaped resonance pattern is still evident near the core in the polytropic model. In the ideal gas model, there is still a remnant of the donor above and just to the left of centre. The bottom images are shown with a larger spatial and density range to reveal the large scale low density structures that form from the merger.

4.2.1 Simulation Diagnostics

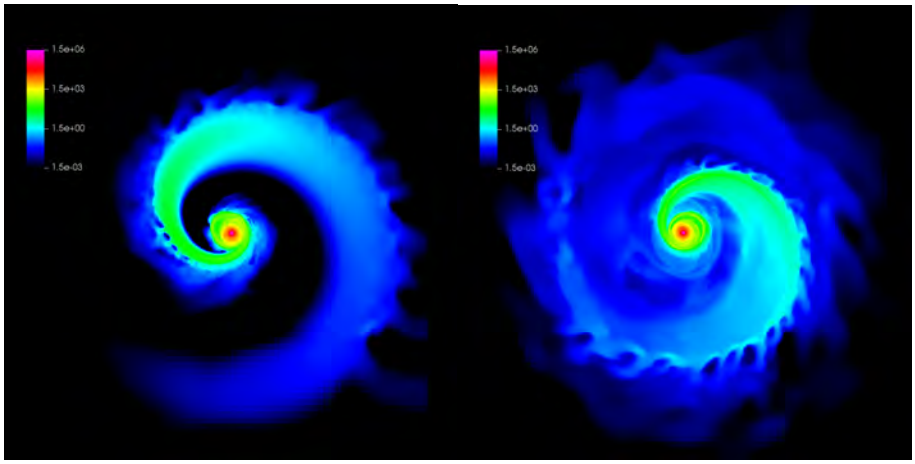
In order to measure the simulated properties of each star in the binary, we must first establish how to locate the stellar surface of each star. OCTO-TIGER accomplishes this iteratively. For each iteration OCTO-TIGER updates the centres of mass of each star and the orbital frequency. Summations are done over the grid cells within the regions defined for each star, and indices i, j, and k referring to grid cells are implied on the RHS of the next two equations. We



(a) Density, $t = 25.0 \times P_0$, polytropic EoS (b) Density, $t = 23.2 \times P_0$, ideal gas EoS



(c) Donor fraction, $t = 25.0 \times P_0$, polytropic EoS (d) Donor fraction, $t = 23.2 \times P_0$, ideal gas EoS



(e) Density, $t = 25.0 \times P_0$, polytropic EoS (f) Density, $t = 23.2 \times P_0$, ideal gas EoS

Figure 15. Slices on the orbital plane for our $q = 0.5$ simulations with a poytropic EoS (left column) or an ideal gas EoS (right column). The box size is 20×10^9 cm (top two rows; top row: logarithmic density scale with range $1.5 - 1.6 \times 10^6$ g/cm^3) or 200×10^9 cm (bottom row; logarithmic density scale with range $1.5 \times 10^{-3} - 1.6 \times 10^6$ g/cm^3). The snapshots are taken at times 25.0 (a, c, e), and $23.2 \times P_0$ (b, d, f). See Table 1 for description.

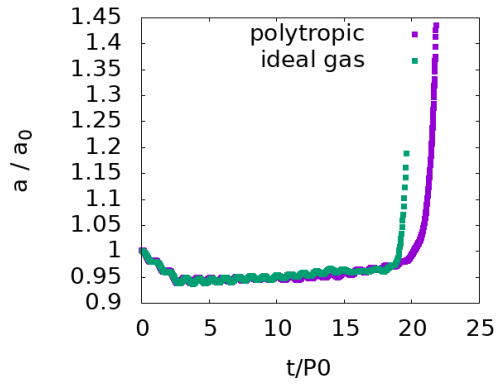


Figure 16. Diagnostic plots from the $q = 0.5$ polytropic and ideal gas simulations. Orbital separation is given as a function of the initial value.

thus define the centres of mass of each star as

$$\mathbf{x}_{1|2} = \frac{\sum^{1|2} \rho \, \mathbf{x} \, \Delta x^3}{M_{1|2}}. \tag{55}$$

The velocity of the centre of mass of each star is

$$\mathbf{u}_{1|2} = \frac{\sum^{1|2} \rho \mathbf{u} \Delta x^3}{M_{1|2}}. \tag{56}$$

Given $\mathbf{x}_{1|2}$ and $\mathbf{u}_{1|2}$, we define the orbital angular frequency as

$$\Omega_{\rm orb} = \frac{\left[(\mathbf{x}_1 - \mathbf{x}_2) \times (\mathbf{u}_1 - \mathbf{u}_2)\right]}{|\mathbf{x}_1 - \mathbf{x}_2|^2} \cdot \mathbf{e}_z, \tag{57}$$

where $\mathbf{e}_z$ is the unit vector in the z direction, out of the orbital plane.

For the first iteration, we compute the centres of mass and orbital frequency based on an initial guess for the bounding surface of each star. OCTO-TIGER evolves the original accretor and original donor mass densities as passive scalars, allowing it to track donor material within the accretor. We use the volumes defined by these densities to seed the first iteration, with cells containing majority accretor (donor) material flagged for the accretor (donor).

With the stellar volumes defined, we can then calculate the centres of mass and orbital frequency. Beginning with the second iteration (out of a total of five), we use the acceleration from the effective potential to find the regions of each star. This acceleration is,

$$\mathbf{g}_{\rm eff} = \mathbf{g} + \mathbf{r}\Omega_{\rm orb}^2, \tag{58}$$

where $\mathbf{r}$ is the radial vector distance to the axis of rotation. We define the quantities

$$q_{1|2} = \mathbf{g}_{\rm eff} \cdot \frac{\mathbf{x} - \mathbf{x}_{1|2}}{|\mathbf{x} - \mathbf{x}_{1|2}|}. \tag{59}$$

If $\min(q_1, 0) < \min(q_2, 0)$, the cell belongs to the accretor. If $\min(q_1, 0) > \min(q_2, 0)$, the cell belongs to the donor. For the case where $q_1 \geq 0$ and $q_2 \geq 0$, the cell is in neither star. This definition by itself may miss some cells in the centres of each star where the effective gravitational force is not guaranteed to point towards the centre of mass. For this reason, any material within a certain critical radius of the centre of mass of a star is included as part of that star regardless of the values of $q_{1|2}$. This critical radius is $\frac{1}{4} R_{L,1|2}$, where $R_{L,1|2}$ is the Roche lobe radius of each star taken from the point mass approximation.

With the stellar volumes defined, we can measure the properties of each star. The mass of each star is thus defined as

$$M_{1|2} = \sum^{1|2} \rho \Delta x^3. \tag{60}$$

The orbital separation is,

$$a = |\mathbf{x}_2 - \mathbf{x}_1| \, . \tag{61}$$

We show the separation of each model in Figure 16. Once the driving phase is over, the orbital separation of the two simulations increases with time up until the final merger. This shows the donor does not plunge into the accretor but instead is tidally disrupted (e.g., Figure 14(c)–(d)). The matter from the disrupted donor then falls onto the accretor. Our diagnostic plots that measure properties of the individual stars do not extend to this phase because there is no longer a clearly defined donor.

Computing the time rate of change of the two stars' masses, $\dot{M}_{1|2}$, requires filtering out high frequency noise, here defined as anything greater than ($\frac{1}{2}\Omega_{\rm orb}$). The regions defining each star change with time and can vary slightly from one time step to the next. The high frequency part of this signal needs to be filtered out to measure a meaningful $\dot{M}_{1|2}$. We apply a windowed sinc filter using an exact Blackman window with frequencies of $\frac{1}{2}\Omega_{\rm orb,0}$ and $\frac{3}{20}\Omega_{\rm orb,0}$. We apply this filter to $M_{1|2}$ and then compute the time-derivatives with nearest neighbors by a first order discrete scheme.

In Figure 17 (top left panel) we show the donor mass loss rate, normalised to the donor's mass. The time derivative is in units of initial orbital periods. The donor initially loses less than one percent per orbit, increasing to $\sim$1 percent, and then ramping up towards the time when the donor is tidally shredded. In Figure 17 (top right panel) we show the amount of mass that is no longer a part of either star. The two star system loses only a small fraction of the donor's mass throughout most of the evolution. This can be mass unbound from the system or mass in a common envelope, but still bound to the system. The ideal gas model loses mass at a higher rate. This is because the accreted gas is shock heated and therefore has higher pressure. It builds up a thicker layer around the accretor, and some of this gas is able to escape.

In Figure 17 (bottom panels) we plot the mass that is outside the surface of either stars, and the mass that is unbound from the system. Unbound mass is that for which the inertial frame kinetic energy density exceeds the inertial frame potential energy. Further from the system centre of mass, where most of the unbound material is likely to be located, this indicator is more reliable than using rotating frame quantities. Most of the mass remains bound. In the case of the ideal gas model, about 10 percent of the mass leaves the stellar boundaries, but only about 1 percent of the mass actually becomes unbound from the system. In the case of the polytropic model these figures are even lower due to lack of shock heating.

The orbital angular momentum is

$$J_{\rm orb} = \frac{M_1 M_2}{M_1 + M_2} \Omega_{\rm orb} a^2. \tag{62}$$

We show $J_{\rm orb}$ for both models in the top left panel of Figure 18 as a function of the initial orbital angular momentum (1.5×10^{51} g m^2 s^{-1}). As expected, both simulations lose orbital angular momentum throughout their evolution.

Each star has a spin angular momentum

$$J_{1|2} = \sum^{1|2} \rho \left(\mathbf{x} - \mathbf{x}_{1|2}\right) \times \left(\mathbf{u} - \mathbf{u}_{1|2}\right) \Delta x^3, \tag{63}$$

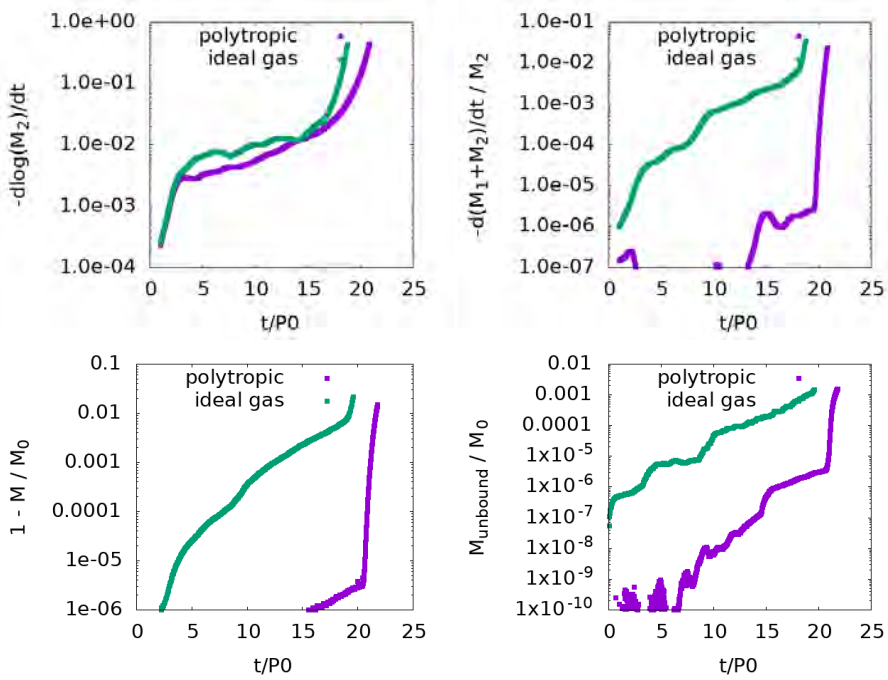


Figure 17. Diagnostic plots from the $q = 0.5$ polytropic and ideal gas simulations. Upper left: donor mass loss rate; upper right: mass loss rate from the both stars ($M = M_1 + M_2$, M_0 is the initial mass of the system); lower left: mass outside the boundary of the two stars; lower right: mass unbound from the binary

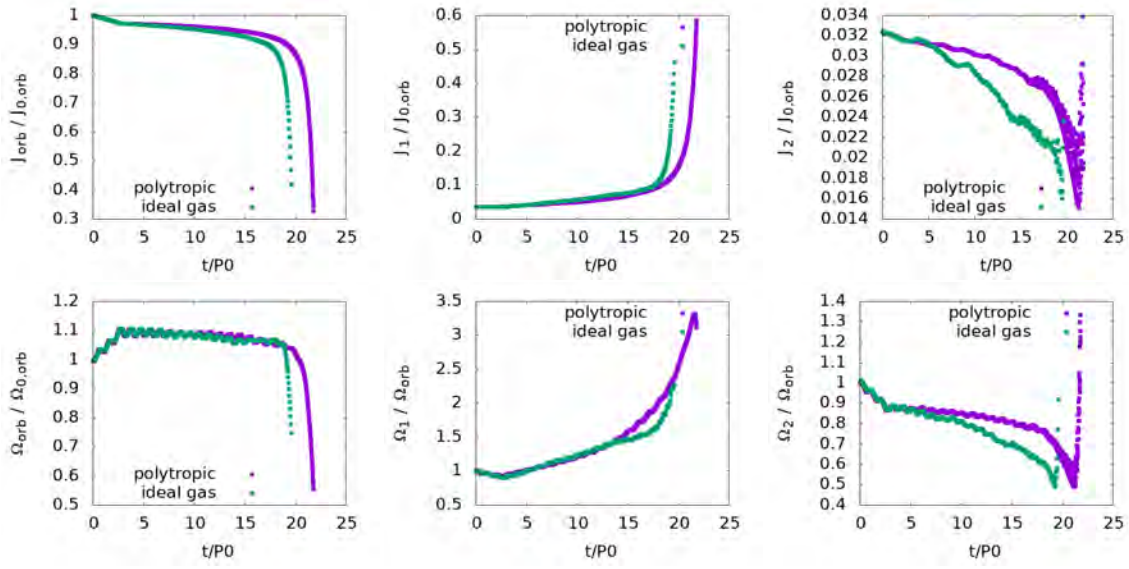


Figure 18. Diagnostic plots from the $q = 0.5$ polytropic and ideal gas simulations. Top left: the orbital angular momentum; top middle: the accretor angular momentum; top right: the donor angular momentum. All angular momenta are normalised to the initial value of the orbital angular momentum. Bottom left: the orbital frequency; bottom middle: the spin frequency of the accretor; bottom right: the spin frequency of the donor. All frequencies are normalised to the orbital frequency at $t = 0$

and an angular frequency assuming rigid body rotation

$$\Omega_{1|2} = \frac{J_{1|2}}{\sum^{1|2} \rho \left|(\mathbf{R} - \mathbf{R}_{1|2})\right|^2 \Delta x^3}, \tag{64}$$

where $\mathbf{R}_{1|2}$ is the projection of $\mathbf{x}_{1|2}$ on the xy-plane. We show $J_{1|2}$ in the middle right and lower right panels of Figure 18. In each model, the accretor gains angular momentum from the donor. We show $\Omega_{1|2}$ along with $\Omega_{\rm orb}$ in Figure 18.

The gravitational torques exerted on each star are

$$\mathbf{T}_{1|2} = \sum^{1|2} (\mathbf{x} - \mathbf{x}_{1|2}) \times \rho \mathbf{g}. \tag{65}$$

The gravitational torque exerted on the orbit is

$$\mathbf{T}_{\rm orb} = \mathbf{x}_1 \times \sum^{1} \rho \mathbf{g} + \mathbf{x}_2 \times \sum^{2} \rho \mathbf{g}. \tag{66}$$

The z-component of $\mathbf{T}_{\rm orb}$ plays an important role in whether or not the binary survives mass transfer. This represents the rate at which the tidal interaction can restore the orbital angular momentum lost to mass transfer (Figure 19 (a) to (c)).

D'Souza et al. (2006) gives an expression relating the time rate of change of the separation to systematic sources of orbital angular momentum, gravitational torque, and the mass transfer rate. This is

$$\frac{\dot{a}}{2a} = \left(\frac{\dot{J}}{J_{\rm orb}}\right)_{\rm sys} + \mathbf{T}_{z,\rm orb} - \frac{\dot{M}_2}{M_2}\left(1 - q - \sqrt{(1+q)r_h}\right). \quad (67)$$

The first term on the RHS, $\left(\frac{\dot{J}}{J_{\rm orb}}\right)_{\rm sys}$, is due to systematic changes in orbital angular momentum due to losses through L_2. In addition, there is a loss of orbital angular momentum during the first 2.7 orbits because of our artificial driving that needs to be accounted for when using this expression. The second term represents the contribution due to the gravitational torque on the orbit. The final term includes, r_h, the effective "circularization" radius. This is the radius from the accretor's centre of mass, which has the same specific angular momentum as the material at the L_1 Lagrange point, normalised to the orbital separation. While the first part of the final term, $\frac{\dot{M}_2}{M_2}(1-q)$, accounts for the change in orbital angular momentum due to the transfer of mass from one star to the other, the second part, $\frac{\dot{M}_2}{M_2}\left(-\sqrt{(1+q)r_h}\right)$, accounts for the orbital angular momentum that is converted to spin angular momentum in the accretor.

Frank et al. (2002) provide an analytical approximation to r_h,

$$r_{h,\rm analytic} = (1+q)\left(0.500 - 0.227\log_{10} q\right)^4. \quad (68)$$

A more accurate expression that includes the torquing of the stream of accreting gas by the donor is that of Verbunt & Rappaport (1988). However, for the current purpouse the expression in Equation 68 is sufficiently accurate. The orbital separation, angular momentum driving rate, gravitational torque on the orbit, the mass transfer rate, and mass ratio are all known or can be calculated from the data set, allowing us to compute a value for r_h using Equation 67. We show this quantity compared with $r_{\rm h,analytic}$ for the polytropic model, in the bottom right panel of Figure 19. In the beginning the value of $r_{\rm h}$ varies wildly. This is during the initial period of angular momentum driving. For the majority of the run, $r_{\rm h}$ is slightly less than $r_{h,\rm analytic}$, indicating the spin angular momentum actually transferred to the accretor is slightly less than predicted by the analytic theory. Toward the end, the computationally-derived $r_{\rm h}$ begins to exceed the analytical value $r_{h,\rm analytic}$. This is because angular momentum is now flowing through the L_2 point, making Equation 67 no longer valid.

The internal, kinetic, and potential energies of the stars are

$$E_{I,1|2} = \sum^{1|2} \rho\varepsilon, \quad (69)$$

$$E_{K,1|2} = \sum^{1|2} \frac{1}{2}\rho u^2, \quad (70)$$

and

$$\Phi_{1|2} = \sum^{1|2} \frac{1}{2}\rho\Phi, \quad (71)$$

respectively. Here the internal energy density, $\rho\varepsilon$, is obtained from the dual energy formalism. We show these energies in Figure 20. The donor loses internal and kinetic energy to the accretor. Although the ideal gas model includes shock heating, since the energy of the shock heated material around the accretor is very small compared to the total internal energy of the star, shock heating makes little apparent difference in the plots of total internal energy. As the accretor gains mass, its gravitational well deepens as its potential energy is lowered. The opposite is true of the accretor.

Grid based codes are subject to centre of mass drift, both physical, from the flow of matter out of the grid (which, by the end of the simulations, is 2.0×10^{29} g and 1.0×10^{28} g for the ideal gas and polytropic EoS, respectively), and numerical. The latter effect is greatly reduced by using a gravity solver that conserves linear momentum, as OCTO-TIGER does. There are still numerical viscosity effects that can push matter one way or another. In addition, the SCF code does not render initial conditions with the centre of mass perfectly at the coordinate centre. Figure 21 (top left panel) shows the magnitude of cylindrical radial location of the system's centre of mass,

$$|\mathbf{R}_{\rm com}| = \sqrt{x_{\rm com}^2 + y_{\rm com}^2} \quad (72)$$

in length units normalised to the finest cell width.

As discussed in Section 2.1, OCTO-TIGER evolves the angular momentum (Equation 7) for the purposes of obtaining the error in the angular momentum conservation. The evolved angular momenta, $\mathbf{l}$, can be compared with the angular momenta obtained from the evolved linear momenta, $\mathbf{x} \times \rho\mathbf{u}$, (Equation 2) to obtain the error. We define the global angular momentum conservation violation error, $\mathbf{l}_{\rm err}$, as:

$$\mathbf{l}_{\rm err} = \sum^{\rm V} \left(\mathbf{l}_i - \mathbf{x}_i \times \rho_i \mathbf{u}_i\right), \quad (73)$$

where the i index refers to a given computational cell and the summation is over the entire computational domain. We show this value over time in Figure 21 (top right panel). In Figure 21, bottom left panel, we plot the total angular momentum non-conservation in the computational domain, scaled to the initial total angular momentum. Here we see that the polytropic EoS conserves momentum to approximately 1 percent over the entire simulation, while for the ideal gas EoS this is considerably worse at 7 percent (the initial drop in angular momentum in the right panel is due to the artificial driving phase and should be ignored). The reason for the difference is that the ideal gas EoS simulation expands more due to shock heating moving gas into the outskirts of the domain where lower resolution dominates with a worse performance of angular momentum conservation. Poorly resolved, sharp momentum density gradients, such as those seen in the outskirts of the simulations (see Figure 15, bottom panels, where we plot density slices) will always disfavour good conservation.

In Figure 21, bottom right panel, we show the energy conservation. The value in the y-axis is scaled to the energy after the driving, at 2.7 orbits. Hence data points before $x = 2.7$ are not meaningful. After that x value, however, we see that the energy conservation is excellent, going from 10^{-14} to 10^{-11} for the ideal gas EoS, or to 10^{-5} for the polytropic EoS.

Without doubt the polytropic EoS simulation is superior in general, because the ideal gas EoS with more heating results is an inflated gas distribution, a higher rate of mass transfer from donor to accretor, and any quantities that derive from that, such as angular momentum transfer or the speed of the merger are also affected.

These simulations were run on 400 cores over 20 nodes, of QueenBee2 (see Section 5) and took approximately 200 hours of wall-clock time, or just over 8 days for a total modest cost of approximately 80 000 CPU-hours. They have between 2 and 8 million cells. If the scaling test in Section 5 are consulted, we would con-

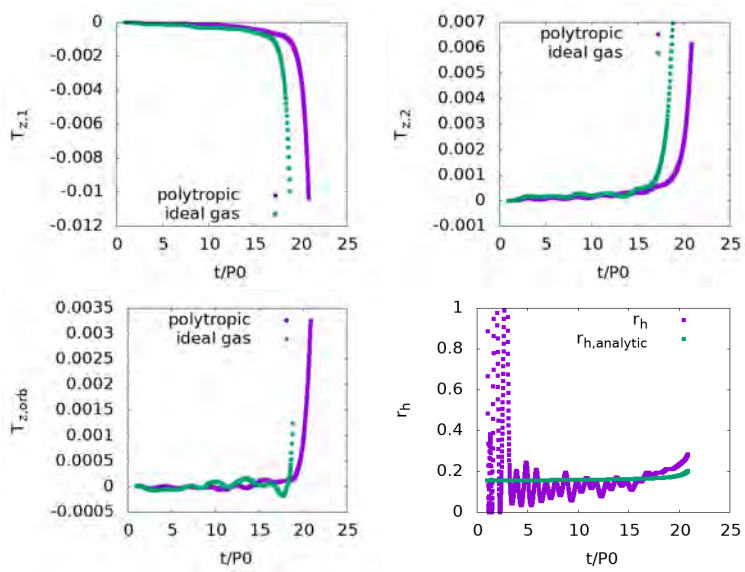


Figure 19. Diagnostic plots from the $q = 0.5$ polytropic and ideal gas simulations. Top left: the gravitational torque on the accretor; top right: the gravitational torque on the donor; bottom left: the gravitational torque on the orbit; bottom right: the circularization radius for the polytropic simulation calculated using Equation 67 and for the analytical approximation using Equation 68

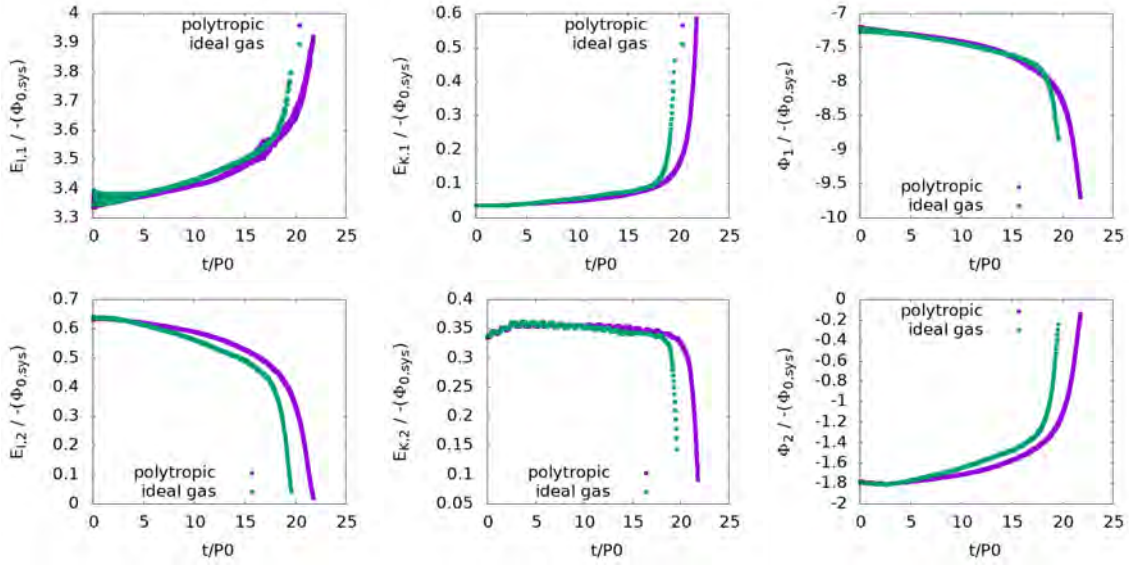


Figure 20. Diagnostic plots from the $q = 0.5$ polytropic and ideal gas simulations. Top left: the accretor's internal energy; top middle: the accretor's kinetic energy; top right: the accretor's potential energy. Bottom left: the donor's internal energy; bottom middle: the donor's kinetic energy; bottom right: the donor's potential energy. All energies are normalised to the absolute value of the total potential energy at $t = 0$

clude that a doubling of the number of cores would reduce the wall clock time between 40 and 60 percent, while any further increase of the number of cores would not be particularly advantageous.

Naturally, this run could be repeated with substantially larger resolution (and larger number of cores), likely leading to an expansion of discovery space. However, increasing resolution with the aim of determining convergence is not straightforward because it is not clear what quantities can be reasonably expected to converge. An increase in resolution automatically lowers the mass transfer rate in the early interaction and lengthens the pre-merger time beyond what can be simulated (see for instance Motl et al. (2017) who compared a 4-million cell WD merger cell simulation to a 47-million cell one, or Reichardt et al. (2019) who compared the early mass transfer before a common envelope interaction between a ∼100 000 SPH particle simulation to one with 1.3 million particles). What quantities, then, should we expect to converge to a given value with increasing resolution? It is possible that, for example, the increase in mass transfer rate over one orbital period, measured at a specific initial separation may be a quantity that is expected to converge. The determination of how to measure the accuracy of simulation results of this level of complexity is an urgent, on-going area of research.

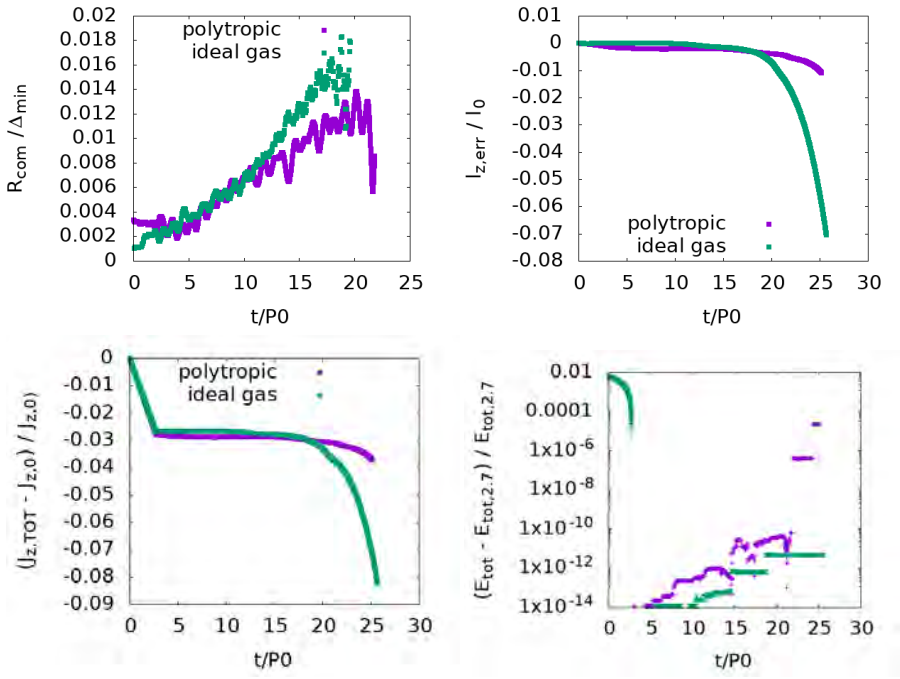


Figure 21. Diagnostic plots from the $q = 0.5$ polytropic and ideal gas simulations. Top left: the location of the system centre of mass (Equation 72) in units of the finest grid cell; top right: the relative angular momentum error (Equation 73); bottom left: the total angular momentum loss from the computational domain (the initial decrease corresponds to the angular momentum that is artificially subtracted from the system during the driving phase); bottom right: the total energy conservation error of the system (the values before $x = 2.7$ orbits are meaningless because of the driving)

Click on the link[6] to view a movie of the $q = 0.5$ ideal gas simulation. Click on the link[7] to view a movie of the $q = 0.5$ polytropic EoS simulation.

5 CODE PERFORMANCE AND SCALING

In this section, we present a series of scaling tests to demonstrate the performance of OCTO-TIGER and we make comparisons with equivalent tests carried out with FLASH. In order to compare the two codes, we set the CFL coefficient to be the same in both ($\eta_{\rm CFL} = 0.4$).

We carried out two types of scaling tests. The first one was performed by executing a number of time steps ($N_{\rm steps}$) in the simulation of a stationary star. The second test included a binary system of stars and was designed to stimulate some level of regridding. Tests were performed on three supercomputers, BigRed3 (BR3), Queen-Bee2 (QB2) and Gadi, with variable resolutions (128^3, 256^3, 512^3, and 1024^3) and variable values of θ (0.35 and 0.5), the gravity resolution parameter. We describe the supercomputer hardware in Table 5. The code and its dependencies' versions used with the three sets of scaling runs are reported in Table 6.

In Figures 22 and 23, we present two measures of code speed: the computational time per time step and the wall-clock time per time step. We show this including and excluding the initialization time, regridding time, and the time to write out the output. While the computational time is a feature of the code, the time to write out the output is a feature of the machine and file system used and can therefore vary among otherwise similar computers. These quantities are a measure of the strong scaling properties of OCTO-TIGER. We also show the number of time steps ($N_{\rm steps}$) times the number of cells ($N_{\rm cells}$) processed per second, which measures the amount of work that the cores have available to do. If we had perfect scaling, these curves should be horizontal, indicating the cores have sufficient work, but they tend instead to curve downwards as the addition of more cores results in cores not having sufficient work to do. The reason why the curves tilt downwards is also the introduction of additional overheads, such as message sending, due to the addition of nodes. Note that above a certain high number of cores, each core has computational work of only several subgrids, in which case the communication between cores dominates the performance, and the scaling of the computational time becomes flat as well.

The values used for the figures are reported in Tables 7–9, organised by computer. In brackets, next to the values used for the plots, we calculate the "speedup" ($S_n = (t_N/t'_N)/(N'_{\rm core}/N_{\rm core})$, where t_N and t'_N are the time step lengths for runs that use $N_{\rm core}$ and $N'_{\rm core}$ cores, respectively). These efficiencies are calculated with respect to the run that uses an entire node in each machine (48 cores on Gadi, 20 cores on QB2 and 24 cores on BR3), when available or with the run with the least number of cores ($N_{\rm cores}$), when not. On the RHS panels of Figures 22 and 23, we mark the value of 0.5 efficiency by short horizontal segments over plotted on each curve.

In the tables, we have reported on the *total time* also known as wall-clock time, which includes the initialization, regridding and the writing to disk part of the computation. If we had reported only the computational time the efficiencies would increase.

We start by comparing the code performance for different values of the gravity solver parameter θ. This comparison was made

[6] https://www.youtube.com/watch?v=0JD5E7DUImw
[7] https://www.youtube.com/watch?v=MfArAQPPHss

Cluster	CPU	Memory	Interconnect	# nodes	# of cores
QB2	2 × 10 Intel E5-2680v2 Xeon processors (+2 GPUs[1])	64 GB	56 Gb/sec (FDR) InfiniBand	504	10800
BR3	2 × 12 Intel Xeon Processort E5-2690 v3	64 GB	Cray Aries	930	22 464
Gadi	2 × 24 core Intel Xenon Scalable 'Cascade Lake'	190 GB	200 Gb/sec (HDR) InfiniBand	3024	145 152

[1] The GPUs are NVIDIA Tesla K20; NVIDIA-SMI 450.51.05, driver version: 450.51.05, CUDA version: 11.0 with 6GB of memory per device.

Table 5. Computational infrastructure used in this work. QB2 is Louisiana's Optical Network Infrastructure's QueenBee2; BR3 is the University of Indiana BigRed3, while Gadi is the Australian National Computational Infrastructure peak machine

Software	QB2	BR3	Gadi
HPX	1.4	1.4	1.4
Vc	1.4.1	1.4.1	1.4.1
Boost	1.69/1.68	1.68	1.72
hwloc	1.11.1	1.11.12	1.11.12
jemalloc	5.1.0	5.1.0	5.1.0
gcc	8.3/7.4	8.3	8.3
hdf5	1.12/1.8	1.8.12	1.8.12
silo	4.10.2	4.10.2	4.10.2
cmake	3.13	3.13.2	3.16.2
CUDA	10.2	-	-
MPI	MVAPICH2 2.3.2/ OpenMPI 4.0	cray-mpich 7.7.10	intel-mpi 2019.8.254

Table 6. Software dependencies of OCTO-TIGER (version 0.8). Note that we used a customized version of Vc. Note that we used the pre-compiled MPI version on the cluster and therefore have some variation there. If the gcc compiler was recent enough to compile HPX and OCTO-TIGER we used the pre-compiled version on the cluster

.

with OCTO-TIGER with 2 million (solid lines) and 17 million cells (dashed lines) using BR3, QB2, QB2+GPUs, and Gadi (red, blue, black and magenta lines, respectively, Figure 22). As expected, the smaller the value of θ (the more precise the gravity solver solution) the longer the simulations take. At 2 million cells, the QB2 tests show an increase in the computational time by increasing the number of cores between 1280 and 2560, while this is less pronounced for the other two computers. This is due to a node-communication problem. The $\theta = 0.35$ simulations are doing more communication between sub-grids and hence nodes. The use of CPU+GPU improves the wall clock time, but also displays the upturn. This behaviour is less pronounced for tests with more than 17 million cells (Figure 23).

Next we compare the total time taken for the test for any code on any machine. For the 2 and 17-million cell simulations, the Gadi simulations are the fastest, although we note that for a range of core number between ∼30 and ∼300 it is the QB2 with GPU that is the fastest. This is particularly evident at 17 million cells, looking at the work per core per time plots (Figure 23, RHS panels). Whether runs with GPUs are faster depends on whether the run is dominated by computation or by communication. GPU are computationally faster, but Gadi has a 4-time faster network communication than any other network used in this paper. Further information on OCTO-TIGER runs with GPUs can be found in Daiß et al. (2019), and will be the subject of future papers.

We now come to the actual scaling properties. For the 2-million cell simulations, the efficiency of increasing the number of cores drops once we move past 1536 cores: on Gadi from 768 to 1546 to 3072 cores the efficiency drops from 34, to 19 to 10 percent. This is similar on QB2 and BR3 (note that on QB2 we are comparing slightly different number of cores: 640, 1280 and 2560). At 17 million cells, the same three steps show efficiencies of 99, 86, and 72 percent on Gadi, while for the other computers, they are markedly smaller, with, for instance, 47, 37, 26 percent on BR3, and slightly better but similar values for QB2, making the runs on Gadi the fastest and most scalable.

Using GPUs with the 2- and 17-million cell simulations reduces the time per step between a factor of 1.5 and 2; the scalability of the hybrid CPU+GPU runs is slightly inferior to those of the pure CPU runs for the 2-million cell simulations, for which processors have less work to do, but almost identical to the non-GPU runs for the 17-million cell simulations.

Next, we look at scalability vs. problem size by comparing runs with 2, 17, 134 million and 1 billion cells on Gadi. Looking at the efficiencies when increasing the number of cores between 1536 and 3072, we find 10, 72, and 75 percent for 2, 17, and 134 million cells, respectively which would imply that increasing the size of the problem from 17 to 134 million has not afforded us a much improved scaling. On the other hand, going from 6144 to 12 736 to 20 735 to 41 472 cores and finally to 82 944 cores, the efficiencies improve markedly upon doubling the number of cores for the 1 billion cell run compared to the 17-million cell run. In fact, it is remarkable that our 1 billion cell run is still scaling very well once we move to ∼80 000 cores, giving hope that we can carry out very high resolution simulations with reasonable wall-clock times (though the actual cost of the simulation in terms of CPU-hours would remain very high).

The scalability properties of FLASH were tested on QB2 at 2- and 17-million cell resolutions. They are very comparable to those of OCTO-TIGER. The computational time per step is intermediate between the $\theta = 0.35$ and 0.5, but tends to be longer than even the more accurate $\theta = 0.35$ OCTO-TIGER simulations when using a higher number of cores.

An additional scaling test was carried out using the first 30 time steps of a binary interaction simulation similar to that of Section 4. The tests was to determine code scalability in view of AMR regridding. We performed a series of simulations on QB2, with increasing resolution by virtue of adding additional levels of refinement, see Table 10 and Figure 24. The code performance is aligned

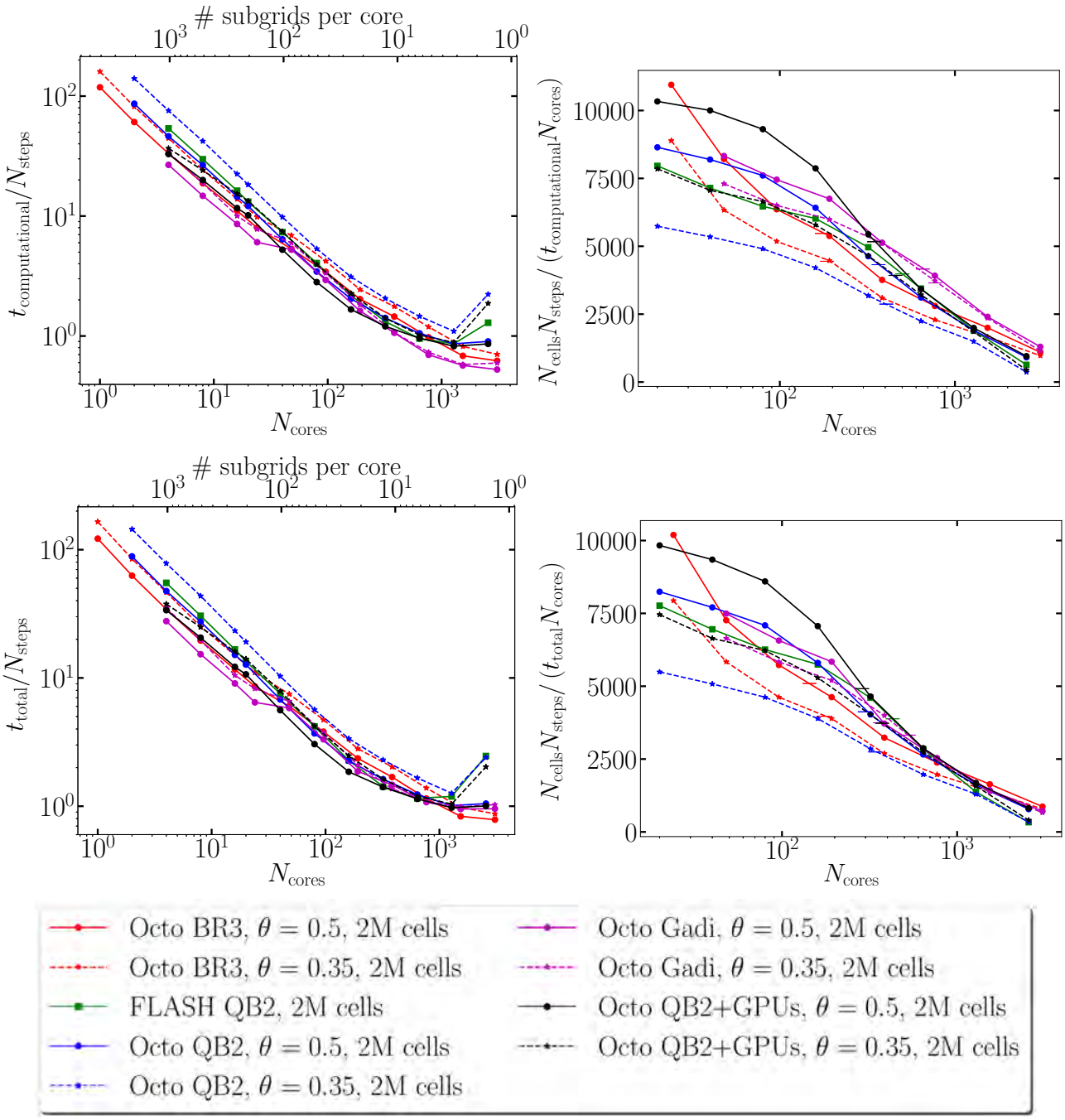


Figure 22. Scaling test carried out on the pulsating polytrope problem of a small size. QB2 refers to QueenBee2, BR3 refers to the BigRed3. The simulations contain $128^3 \approx 2M$ cells or $16^3 = 4096$ subgrids. The short horizontal segments mark the 0.5 efficiency

with that we have observed for the static polytrope on the same computer cluster (Table 8), although a side by side comparison is not possible in virtue because the number of cells was not exactly the same.

We finally carried out a last scaling test, where twelve regridding events took place in 100 time stpes, and where we started the simulation from an intermediate stage of the $q = 0.5$ simulation in order to trigger substantial regridding (Table 7, last column). This scaling runs were carried out on Gadi. Although the number of AMR boundaries varied by ≈ 30 percent in these runs, the scaling was similar to a uniform grid with no regridding taking place (see red-dotted triangle line in Figure 23 and the last column in Table 7). We conclude that our regridding scheme scales well to high number of cores.

6 SUMMARY AND CONCLUSIONS

We have presented a thorough suite of benchmark tests and a comprehensive and detailed description of the AMR hydrodynamics code OCTO-TIGER. Although OCTO-TIGER has been used in tandem with other codes in previous publications in astrophysics

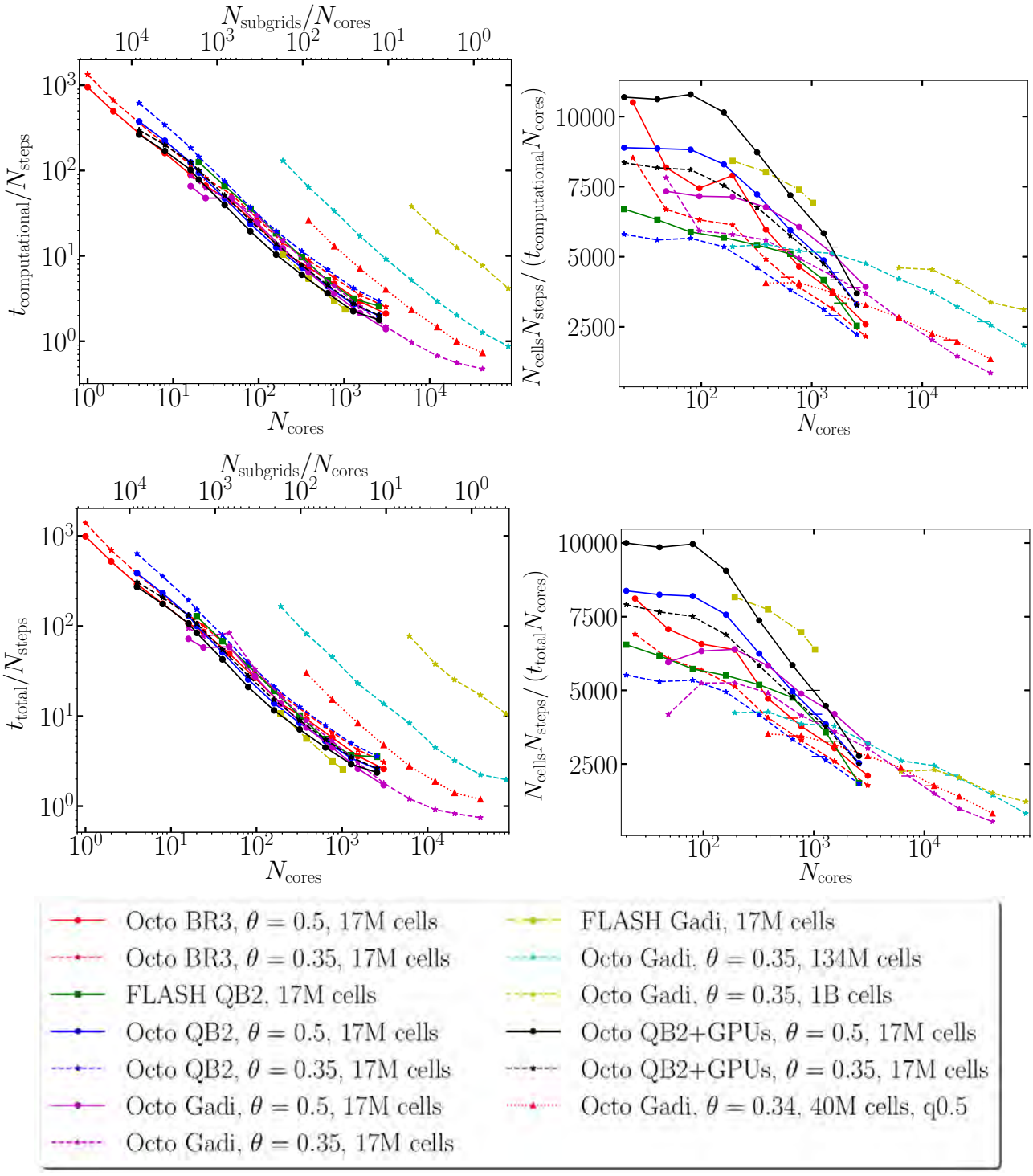


Figure 23. Scaling test carried out on the pulsating polytrope problem of big sizes. QB2 refers to QueenBee2, BR3 refers to the BigRed3. The simulations contain 256^3 (17M), 512^3 (134M), and 1024^3 (1B) cells or 4096, 32 768, 262 144, subgrids, respectively. One subgrids per core of the 17M cells simulations (top axis on left panels) is 8 subgrids per core of the 134M cells, and 64 subgrids per core of the 1B cells simulations. The short horizontal segments mark the 0.5 efficiency

(Kadam et al. 2016; Staff et al. 2018) as well as in computer science (e.g. Pfander et al. 2018; Daiß et al. 2019; Heller et al. 2019b), there has never been an actual "method paper".

OCTO-TIGER joins a small suite of codes specifically designed to study the merger of white dwarf stars while also modelling the earlier, pre-merger phase. OCTO-TIGER is designed to be a scalable, accurate code that is able to exploit CPU and GPU computer architectures. It is currently optimised to model stellar mergers for similar-sized stars that can be modelled using a star setup via a barotropic EoS, although for the evolution one can choose

Code	OCTO-TIGER	OCTO-TIGER	FLASH	OCTO-TIGER	OCTO-TIGER	OCTO-TIGER	OCTO-TIGER	OCTO-TIGER[a]
$N_{\rm cells}$	2M	2M	17M	17M	17M	134M	1000M	40M
θ	0.5	0.35	–	0.5	0.35	0.35	0.35	0.34
$N_{\rm cores}$								
4	27.7	35.0	–	–	–	–	–	–
8	15.3	19.7	–	–	–	–	–	–
16	9.06	10.5	–	72.1	94.7	–	–	–
24	6.45	8.26	–	57.7	77.1	–	–	–
48	5.83 (–)	6.58 (–)	–	58.7 (–)	83.5 (–)	–	–	–
96	3.32 (88%)	3.74 (88%)	–	27.6 (125%)	33.3 (125%)	–	–	–
192	1.87 (78%)	2.10 (78%)	10.7 (–)	13.7 (126%)	16.6 (125%)	165 (–)	–	–
384	1.47 (50%)	1.36 (60%)	5.64 (95%)	7.48 (117%)	8.90 (117%)	81.8 (101%)	–	30.1 (–)
768	1.08 (34%)	1.09 (38%)	3.13 (85%)	4.47 (99%)	5.27 (99%)	45.3 (91%)	–	15.3 (98%)
1024	–	–	2.57 (78%)	–	–	–	–	–
1536	0.951 (19%)	0.988 (21%)	–	2.60 (86%)	3.03 (86%)	23.0 (90%)	–	8.37 (90%)
3072	0.955 (10%)	1.03 (10%)	–	1.71 (72%)	1.80 (72%)	13.6 (75%)	–	4.77 (79%)
6144	–	–	–	–	1.20 (55%)	8.36 (62%)	77.7 (–)	2.78 (68%)
12 288	–	–	–	–	0.915 (36%)	4.47 (58%)	38.0 (102%)	1.88 (50%)
20 736	–	–	–	–	0.829 (23%)	3.20 (48%)	25.3 (91%)	1.40 (40%)
41 472	–	–	–	–	0.745 (13%)	2.24 (34%)	17.1 (67%)	1.19 (23%)
82 944	–	–	–	–	–	1.97 (19%)	10.6 (54%)	–

[a] Test using a binary simulation with $q = 0.5$.

Table 7. Wall clock time per time step in seconds for different scaling tests carried out on Gadi. In parenthesis we report the speedup of a given calculation with respect to the calculation that uses one node (48 cores), interpreted as the fractional reduction of the time step compared to the fractional increase in core count

Code	FLASH	OCTO	OCTO	OCTO	OCTO	FLASH	OCTO	OCTO	OCTO	OCTO
$N_{\rm cells}$	2M	2M	2M	2M	2M	17M	17M	17M	17M	17M
θ	–	0.5	0.35	0.5	0.35	–	0.5	0.35	0.5	0.35
GPUs	–	–	–	✓	✓	–	–	–	✓	✓
$N_{\rm cores}$										
2	–	88.7	144	–	–	–	–	–	–	–
4	55.1	47.6	78.1	33.7	37.7	–	388	637	272	308
8	30.5	27.4	43.6	20.6	24.9	–	231	356	175	207
16	16.7	15.1	23.4	12.2	16.0	–	131	193	107	132
20	13.5 (–)	12.7(–)	19.1 (–)	10.7 (–)	14.0 (–)	128 (–)	100 (–)	152 (–)	83.9 (–)	106 (–)
40	7.54 (90%)	6.80 (93%)	10.3 (93%)	5.61 (95%)	7.89 (89%)	67.9 (94%)	50.8 (98%)	79.2 (96%)	42.6 (99%)	54.7 (97%)
80	4.19 (81%)	3.70 (86%)	5.67 (84%)	3.05 (87%)	4.21 (83%)	36.6 (87%)	25.6 (98%)	39.2 (97%)	21.0 (100%)	27.9 (95%)
160	2.28 (74%)	2.26 (70%)	3.37 (71%)	1.85 (72%)	2.48 (71%)	19.0 (84%)	13.8 (90%)	21.2 (90%)	11.6 (91%)	15.2 (87%)
320	1.42 (59%)	1.62 (49%)	2.30 (52%)	1.41 (47%)	1.63 (54%)	10.1 (79%)	8.39 (75%)	12.6 (75%)	7.11 (74%)	8.98 (74%)
640	1.15 (37%)	1.23 (32%)	1.66 (36%)	1.14 (29%)	1.20 (37%)	5.51 (73%)	5.28 (59%)	7.87 (60%)	4.48 (59%)	5.49 (60%)
1280	1.19 (18%)	1.01 (20%)	1.26 (24%)	0.975 (17%)	1.04 (21%)	3.66 (55%)	3.40 (46%)	4.99 (48%)	2.93 (45%)	3.48 (48%)
2560	2.46 (4%)	1.05 (9%)	2.40 (6%)	1.00 (8%)	2.03 (5%)	3.54 (28%)	2.58 (30%)	3.57 (33%)	2.36 (28%)	2.63 (32%)

Table 8. Wall clock time per time step in seconds for different scaling tests carried out on one stationary polytrope on QueenBee2. In parenthesis we report the speedup of a given calculation with respect to the calculation that uses one node (20 cores), interpreted as the fractional reduction of the time step compared to the fractional increase in core count

among a number of analytical EoS options. It does not currently include nuclear reactions such as for example the code CASTRO (Katz et al. 2016). OCTO-TIGER's gravity accuracy outperforms FLASH, with overall smaller residuals in the constant density sphere test (Section 3.4), although the comparison may not be with exactly the same parameters. The static and translating polytrope tests also show OCTO-TIGER's superior conservation of the peak density and overall mass of the structure, as well as the centre of mass position and overall velocity (for the translating polytropes; Sections 3.5 and 3.6).

In the rotating polytrope test, simulations in the rotating frame distinctly outperform those calculated in the inertial frame, although even in the inertial frame the polytrope retains its mass, position and radius to excellent precision. OCTO-TIGER demonstrates a superior conservation of angular momentum and energy in this test, although the hydrodynamics is not well exercised in single grid tests where there is little gas motion (Section 3.7).

We have carried out scaling tests using a static polytrope, where the entire grid is fully refined with resolutions from 2 million to 1 billion cells. We have used three different computer clusters, two university-based mid-tier computers, QueenBee2 (Louisiana State University) and BigRed3 (Indiana University) and one peak facility, Gadi, at the Australian Government's National Computational Infrastructure Centre. Strong scaling properties of the code are excellent up to $\sim 82\,000$, the largest number of cores tested, but only for the largest problems of 1 billion cells. With smaller problems of 134 million cells, performance improvement is obtained up to a maximum of $\sim$20 000 cores (after which the speedup is modest). For the 17 million cell test, similar to the number used in the binary simulation in Section 4, the optimum number of cores is $\sim$6000.

Code	OCTO-TIGER	OCTO-TIGER	OCTO-TIGER	OCTO-TIGER
$N_{\rm cells}$	2M	2M	17M	17M
θ	0.5	0.35	0.5	0.35
$N_{\rm cores}$				
1	122	165	987	1390
2	62.7	84.8	521	694
4	34.3	46.5	2938	382
8	19.5	25.2	177	219
16	11.7	15.0	107	132
24	8.57 (–)	11.0 (–)	86.11 (–)	101 (–)
48	6.01 (71%)	7.48 (74%)	49.4 (87%)	57.5 (88%)
96	3.81 (56%)	4.72 (58%)	26.6 (81%)	30.7 (82%)
192	2.36 (45%)	2.81 (49%)	13.8 (79%)	17.0 (74%)
384	1.69 (32%)	2.03 (34%)	9.25 (58%)	10.7 (59%)
768	1.15 (23%)	1.38 (25%)	5.77 (47%)	6.61 (48%)
1536	0.834 (16%)	0.980 (18%)	3.60 (37%)	4.21 (38%)
3072	0.785 (9%)	0.872 (10%)	2.59 (26%)	3.07 (26%)

Table 9. Wall clock time per time step in seconds for different scaling tests carried out on one stationary polytrope on BigRed3. In parenthesis we report the speedup of a given calculation with respect to the calculation that uses one node (24 cores), interpreted as the fractional reduction of the time step compared to the fractional increase in core count

Code	OCTO-TIGER	OCTO-TIGER	OCTO-TIGER	OCTO-TIGER	OCTO-TIGER
$N_{\rm cells}$	0.51M	1.5M	3.0M	10.5M	45M
θ	0.34	0.34	0.34	0.34	0.34
$N_{\rm cores}$					
20	6.73	18.3	34.18	103	–
40	3.98 (85%)	9.89 (92%)	18.1 (95%)	53.1 (97%)	–
80	2.50 (67%)	5.53 (83%)	9.76 (88%)	27.4 (94%)	128
160	1.86 (45%)	3.47 (66%)	5.67 (76%)	15.0 (86%)	65.1 (98%)
320	1.53 (27%)	2.50 (46%)	3.93 (54%)	9.04 (71%)	37.9 (84%)
640	1.31 (16%)	1.91 (30%)	2.77 (30%)	5.83 (55%)	21.4 (75%)
1280	1.19 (9%)	1.56 (18%)	2.11 (25%)	4.02 (40%)	12.6 (63%)

Table 10. Wall clock time per time step in seconds for different scaling tests carried out on QueenBee2 on a binary interaction simulation similar to the one of Section 4. In parenthesis we report the speedup of a given calculation with respect to the calculation that uses one node (20 cores), except for the 45 million cell run, which is given with respect to the smallest number of cores tested (80 cores)

We have also carried out somewhat limited scaling tests of a binary problem using QueenBee2 and Gadi to measure the performance when regridding takes place and found that the scaling properties of the code are similar to those assessed using the stationary polytrope with no AMR regridding.

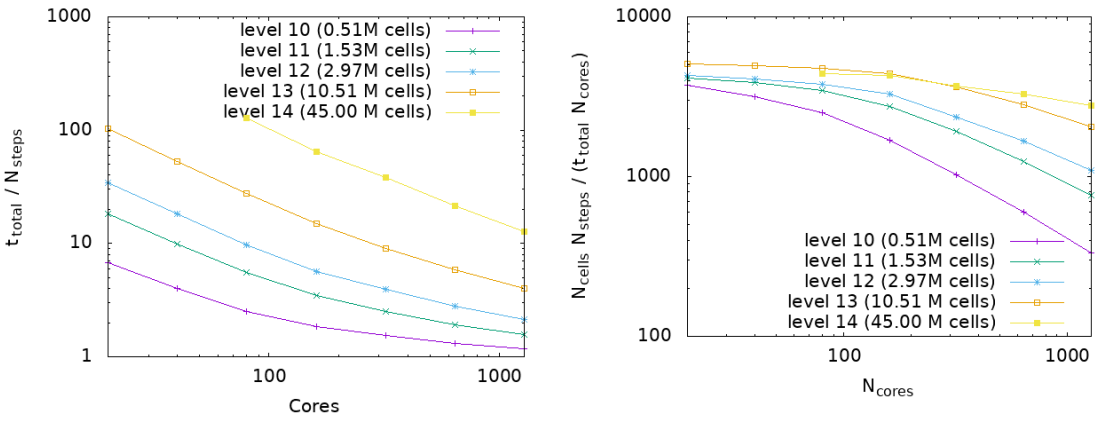
While the emphasis in this paper is not OCTO-TIGER's ability to use mixed CPU-GPU architectures, we have carried out a small test that showed that the use of 2 GPU units per node improves the step time by a factor of 1.5-2. Given the increased emphasis on the use of GPUs for this type of computation (see references in Section 2.10) and we will further address this capability in future work.

Finally, we have used OCTO-TIGER to carry out intermediate resolution WD merger simulations with an ideal gas and a polytropic EoS and compared the results to a similar simulations carried out by Motl et al. (2017) using two different codes, the grid code FLOWER, and the SPH code SNSPH (Fryer et al. 2006). The total run time of this 2-8 million cell simulation was approximately 8 days and it used 80 000 CPU-hours on 400 cores of the computer cluster QueenBee2. By comparison the high resolution cylindrical simulation in Motl et al. (2017), with an approximately similar resolution, ran for over a year. Because of the relatively small wall-clock times achieved by OCTO-TIGER thanks to its excellent scalability, higher resolution runs are possible.

Our improvements to OCTO-TIGER include (i) applying high resolution only to the binary through adaptive mesh refinement (ii) exploiting the more favorable Courant condition from Cartesian coordinates while maintaining acceptable conservation and (iii) leveraging a parallel framework that enables simulations on a very large number of computing elements. We have effectively reduced the wall clock time for a merger simulation to complete from over a year to essentially one day. These developments enable a wide variety of numerical experiments that would not be possible or conclusive if run only at lower resolution.

An interesting question is how to best exploit resolution improvements. We know from Motl et al. (2017) that simply increasing the resolution will lower the initial mass-transfer rate and greatly increase the early, pre-merger phase of the simulation. This would extend enormously the pre-merger time, making the simulation impossibly long. Under such circumstances we would be forced to increase the artificial angular momentum extraction (the driving) to speed up the simulation to a time when the mass-transfer rate is high enough that the merger takes place. As such, the mass transfer rate and the pre-merger timescale cannot be measured with this type of simulation. On the other hand, there are other parameters of interest that would benefit from higher resolution. An example is the actual structure of the flow during the low mass-transfer rate time, or the details of the flow at or after the time of merger, including the determination of dredged-up elements.

In future papers we will implement a number of additions. First we will experiment with increasing the photospheric resolution, while at the same time implementing a radiation transport



(a) (b)

Figure 24. Scaling results for the first 30 time steps of a binary simulation on LONI's QB2. The left panel shows the total wall-clock time per time step as a function of core count, while the right panel shows the total throughput per core in terms of $N_{\rm cells} N_{\rm timesteps}/(t_{\rm wallclock} N_{\rm cores})$

algorithm. With this the immediate aim is to model the light signature, something that would greatly benefit from sufficient resolution at the photosphere as thoroughly explained by Galaviz et al. (2017). After that, the following improvement will be to use model stars imported from 1D stellar structure simulations, instead of being constructed using barotropic solutions, and that can use a tabulated EoS. A follow-up study is already underway to simulate a $q = 0.1$ binary which mimics the V1309 Sco system - this would be only the second simulation of this system, after the 100 000 SPH particle hydrodynamic simulation of Nandez et al. (2014), and a $q = 0.7$ WD binary thought to be a typical progenitor system for the R Coronae Borealis stars (e.g., Clayton et al. 2011).

ACKNOWLEGEMENTS

This work was supported by National Science Foundation Award 1814967. The numerical work was carried out using the computational resources (QueenBee2) of the Louisiana Optical Network Initiative (LONI) and Louisiana State University's High Performance Computing (LSU HPC). Our use of BigRed3 at Indiana University was supported by Lilly Endowment, Inc., through its support for the Indiana University Pervasive Technology Institute. This research was undertaken with the assistance of resources (Gadi) from the National Computational Infrastructure (NCI Australia), an NCRIS enabled capability supported by the Australian Government. We wish to thank Dale Roberts at NCI and Richard Miller at Macquarie University for helping the team to run scaling tests on an exorbitant number of Gadi cores. Finally we thank Macquarie University for funding the Gadi computer time from the university allocation.

DATA AVAILABILITY

The source code of OCTO-TIGER (Marcello et al. 2021) is available on GitHub[8] released under the Boost Software License Version 1. The build scripts to build OCTO-TIGERand its dependencies are available on GitHub[9] as well. The input files to run the OCTO-TIGER simulations are available on Zenodo (Marcello et al. 2020). Table 6 lists the software version used on the different clusters. Note that we used the pre-compiled MPI version on the cluster and therefore have some variation there. If the gcc compiler was recent enough to compile HPX and OCTO-TIGER we used the pre-compiled version on the cluster.

For movies of the simulations and download of selected simulation data see our YouTube channel[10].

[8] https://github.com/STEllAR-GROUP/octotiger

[9] https://github.com/diehlpk/PowerTiger

[10] https://youtube.com/playlist?list=PL7vEgTL3FalaLHjQQH5UtqEVR9-WkFeZp

REFERENCES

Amini P., Kaiser H., 2019, in 2019 IEEE/ACM Third Annual Workshop on Emerging Parallel and Distributed Runtime Systems and Middleware (IPDRM). pp 26–33

Bordner J., Norman M. L., 2012, in Proceedings of the Extreme Scaling Workshop. BW-XSEDE '12. University of Illinois at Urbana-Champaign, USA

Bryan G. L., Norman M. L., Stone J. M., Cen R., Ostriker J. P., 1995, Computer Physics Communications, 89, 149

Bulla M., Sim S. A., Pakmor R., Kromer M., Taubenberger S., Röpke F. K., Hillebrandt W., Seitenzahl I. R., 2016, MNRAS, 455, 1060

Burdge K. B., et al., 2020, ApJ, 905, L7

Clayton G. C., 2012, Journal of the American Association of Variable Star Observers (JAAVSO), 40, 539

Clayton G. C., et al., 2011, ApJ, 743, 44

Colella P., Woodward P. R., 1984, Journal of Computational Physics, 54, 174

Copik M., Kaiser H., 2017, in Proceedings of the 5th International Workshop on OpenCL. pp 1–7

D'Souza M. C. R., Motl P. M., Tohline J. E., Frank J., 2006, The Astrophysical Journal, 643, 381

Daiß G., et al., 2019, in Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. pp 1–37

Daiß G., et al., 2019, From Piz Daint to the Stars: Simulation of Stellar Mergers using High-Level Abstractions (arXiv:1908.03121)

De Marco O., Long J., Jacoby G. H., Hillwig T., Kronberger M., Howell S. B., Reindl N., Margheim S., 2015, MNRAS, 448, 3587

Dehnen W., 2000, ApJ, 536, L39

Després B., Labourasse E., 2015, Journal of Computational Physics, 290, 28

Diehl P., Seshadri M., Heller T., Kaiser H., 2018, in 2018 IEEE/ACM 4th International Workshop on Extreme Scale Programming Models and Middleware (ESPM2). pp 19–28, doi:10.1109/ESPM2.2018.00006

Duquennoy A., Mayor M., 1991, Astronomy and Astrophysics, 248, 485

Even W., Tohline J. E., 2009, Astrophys. J. Suppl., 184, 248

Frank J., King A., Raine D. J., 2002, Accretion Power in Astrophysics: Third Edition

Fryer C. L., Rockefeller G., Warren M. S., 2006, ApJ, 643, 292

Fryxell B., et al., 2000, ApJ Supplement Series, 131, 273

Galaviz P., De Marco O., Passy J.-C., Staff J. E., Iaconi R., 2017, ApJ Supplement Series, 229, 36

Hachisu I., 1986a, ApJ Supplement Series, 61, 479

Hachisu I., 1986b, ApJ Supplement Series, 62, 461

Heller T., et al., 2019a, The International Journal of High Performance Computing Applications, 33, 699

Heller T., et al., 2019b, The International Journal of High Performance Computing Applications, 33, 699

Hillebrandt W., Niemeyer J. C., 2000, Annual Review of Astronomy and Astrophysics, 38, 191

Hurley M., Roberts P. H., Wright K., 1966, ApJ, 143, 535

Ivanova N., et al., 2013, The Astronomy and Astrophysics Review, 21, 59

Ivezić Ž., et al., 2007, in Valsecchi G. B., Vokrouhlický D., Milani A., eds, IAU Symposium Vol. 236, Near Earth Objects, our Celestial Neighbors: Opportunity and Risk. pp 353–362 (arXiv:astro-ph/0701506), doi:10.1017/S1743921307003420

Jetley P., Gioachin F., Mendes C., Kale L. V., Quinn T., 2008, in 2008 IEEE International Symposium on Parallel and Distributed Processing. pp 1–12

Jha S. W., 2017, Type Iax Supernovae. p. 375, doi:10.1007/978-3-319-21846-5˙42

Kadam K., Motl P., Frank J., Clayton G., Marcello D., 2016, Monthly Notices of the Royal Astronomical Society, 462, stw1814

Kadam K., Motl P. M., Marcello D. C., Frank J., Clayton G. C., 2018, MNRAS, 481, 3683

Kaiser H., Brodowicz M., Sterling T., 2009, in 2009 International Conference on Parallel Processing Workshops. pp 394–401

Kaiser H., Heller T., Adelstein-Lelbach B., Serio A., Fey D., 2014, in Proceedings of the 8th International Conference on Partitioned Global Address Space Programming Models. pp 1–11

Kaiser H., et al., 2020, Journal of Open Source Software, 5, 2352

Kale L. V., Krishnan S., 1993, Technical report, CHARM++: A Portable Concurrent Object Oriented System Based on C++. USA

Kashyap R., Haque T., Lorén-Aguilar P., García-Berro E., Fisher R., 2018, ApJ, 869, 140

Katz M. P., Zingale M., Calder A. C., Swesty F. D., Almgren A. S., Zhang W., 2016, ApJ, 819, 94

Kippenhahn R., Kohl K., Weigert A., 1967, Zeitschrift für Astrophysik, 66, 58

Kippenhahn R., Thomas H.-C., Weigert A., 1968, Zeitschrift für Astrophysik, 69, 265

Kurganov A., Noelle S., Petrova G., 2000, SIAM J. Sci. Comput, 23, 707

MacLeod M., Loeb A., 2020, ApJ, 893, 106

Marcello D. C., Tohline J. E., 2012, ApJ Supplement Series, 199, 35

Marcello D., Kadam K., Clayton G. C., Frank J., Kaisar H., Motl P., 2016, in Accretion Processes in Cosmic Sources. p. 55

Marcello D. C., Shiber S., Marco O. D., Frank J., Clayton G. C., Motl P. M., Diehl P., Kaiser H., 2020, Files for reproducing results in Octo-Tiger: a new, 3D hydrodynamic code for stellar mergers that uses HPX parallelisation, doi:10.5281/zenodo.4393374, https://doi.org/10.5281/zenodo.4393374

Marcello D., et al., 2021, STEllAR-GROUP/octotiger: Benchmark paper, doi:10.5281/zenodo.4432574, https://doi.org/10.5281/zenodo.4432574

Mason E., Diaz M., Williams R. E., Preston G., Bensby T., 2010, A&A, 516, A108

Meyer F., Meyer-Hofmeister E., 1979, A&A, 78, 167

Motl P. M., Tohline J. E., Frank J., 2002, The Astrophysical Journal Supplement Series, 138, 121

Motl P. M., Frank J., Tohline J. E., D'Souza M. C. R., 2007, ApJ, 670, 1314

Motl P. M., Frank J., Staff J., Clayton G. C., Fryer C. L., Even W., Diehl S., Tohline J. E., 2017, ApJ Supplement Series, 229, 27

Nandez J. L. A., Ivanova N., Lombardi Jr. J. C., 2014, ApJ, 786, 39

Pakmor R., Kromer M., Taubenberger S., Sim S. A., Röpke F. K., Hillebrandt W., 2012, ApJ, 747, L10

Pejcha O., Metzger B. D., Tomida K., 2016, MNRAS, 461, 2527

Pfander D., DaißG., Marcello D., Kaiser H., Pflüger D., 2018, in Proceedings of the International Workshop on OpenCL. IWOCL '18. ACM, New York, NY,

USA, pp 19:1–19:8, doi:10.1145/3204919.3204938, http://doi.acm.org/10.1145/3204919.3204938
Reichardt T. A., De Marco O., Iaconi R., Tout C. A., Price D. J., 2019, MNRAS, 484, 631
Ricker P. M., Timmes F. X., Taam R. E., Webbink R. F., 2019, IAU Symposium, 346, 449
Rucinski S., 2010, in Kalogera V., van der Sluys M., eds, Vol. 1314, American Institute of Physics Conference Series. pp 29–36, doi:10.1063/1.3536391
Schneider F. R. N., Ohlmann S. T., Podsiadlowski P., Röpke F. K., Balbus S. A., Pakmor R., Springel V., 2019, Nature, 574, 211
Sedov L. I., 1946, Journal of Applied Mathematics and Mechanics, 10, 241
Shu C.-W., Osher S., 1989, Journal of Computational Physics, 83, 32
Smith R. C., 1984, Quarterly Journal of the Royal Astronomical Society, 25, 405
Sod G. A., 1978, Journal of Computational Physics, 27, 1
Solheim J.-E., 2010, Publications of the Astronomical Society of the Pacific, 122, 1133
Staff J. E., et al., 2018, ApJ, 862, 74
Tauris T. M., van den Heuvel E. P. J., 2014, ApJ, 781, L13
The C++ Standards Committee 2017, Technical report, ISO International Standard ISO/IEC 14882:2017, Programming Language C++. Geneva, Switzerland: International Organization for Standardization (ISO).
The C++ Standards Committee 2020, Technical report, ISO International Standard ISO/IEC 14882:2020, Programming Language C++. Geneva, Switzerland: International Organization for Standardization (ISO).
Thoman P., et al., 2018, The Journal of Supercomputing, 74, 1422
Tylenda R., et al., 2011, A&A, 528, A114
Verbunt F., Rappaport S., 1988, ApJ, 332, 193
Warner B., 2003, Cataclysmic Variable Stars, doi:10.1017/CBO9780511586491.
Webbink R. F., 1992, in van den Heuvel E. P. J., Rappaport S. A., eds, NATO Advanced Study Institute (ASI) Series C Vol. 377, X-Ray Binaries and Recycled Pulsars. pp 269–280

This paper has been typeset from a TEX/ LATEX file prepared by the author.