Developing a code for general relativistic hydrodynamics with application to neutron star oscillations.

Dissertation
zur Erlangung des Grades eines Doktors
der Naturwissenschaften
der Fakultät für Mathematik und Physik
der Eberhard-Karls-Universität zu Tübingen

vorgelegt von
Wolfgang Kastaun
aus Dinslaken

2007

Tag der mündlichen Prüfung: 2.4.2007
Dekan: Prof. Dr. Nils Schopohl
1. Berichterstatter: Prof. Dr. Wilhelm Kley
2. Berichterstatter: Prof. Dr. Konstantinos Kokkotas

Zusammenfassung

Im Zuge dieser Doktorarbeit wurde ein Programm zur numerischen Simulation idealer Flüssigkeiten innerhalb beliebiger gekrümmter Raumzeiten in einer bis drei Dimensionen im Rahmen der Allgemeinen Relativitätstheorie erstellt.

Das numerische Verfahren basiert auf einer HRSC (High Resolution Shock Capturing) Methode. Diese wurde dahingehend modifiziert, daß Druck- und Gravitationskräfte konsistent berechnet werden. Dadurch wird eine höhere Genauigkeit im Falle quasi-stationärer, isentroper Systeme erreicht, ohne dabei den Anwendungsbereich einzuschränken, wie dies z.B. beim linearisierten Ansatz der Fall wäre. Ferner wurde eine Reformulierung der allgemeinrelativistischen hydrodynamischen Zeitentwicklungsgleichungen hergeleitet, welche das Fundament des neuen numerischen Verfahrens bildet.

Am Beispiel eines nichtrotierenden und eines starr rotierenden stationären Neutronensternmodells unter Verwendung der Cowling-Näherung (d.h. mit zeitunabhängigem Gravitationsfeld) wurde das Verfahren erfolgreich getestet, insbesondere wurden umfangreiche Konvergenztests durchgeführt.

Durch Störung der Anfangsdaten wurden jeweils sieben verschiedene Schwingungsmoden angeregt, und mit Hilfe von Fourieranalysen deren Frequenzen und Eigenfunktionen bestimmt. Ein Vergleich mit Ergebnissen anderer Arbeiten ergab eine gute Übereinstimmung, beispielsweise stimmen die Frequenzen besser als 1.7%, überein, bis auf einen Fall besser als 0.8%.

Als problematisch erwies sich die Behandlung der Sternenoberfläche, d.h. des Übergangs zum Vakuum. Hierzu wurde eine neue Methode entwickelt, die zwar eine Verbesserung, aber noch keine endgültige Lösung des Problems darstellt: Mit Hilfe eines speziell entwickelten Testproblems ähnlich zum Neutronenstern, jedoch ohne Gebiete mit Vakuum, konnte gezeigt werden, daß die numerischen Fehler zum größten Teil durch die Behandlung der Sternoberfläche verursacht werden.

Es besteht die Möglichkeit, das Programm mit einem weiteren Programm zur Zeitentwicklung der Raumzeit bei gegebener Materieverteilung zu koppeln, so daß allgemeinrelativistische Simulationen ohne einschränkende Näherungen durchgeführt werden können. Erste derartige Simulationen eines nichtrotierenden Neutronensterns verliefen erfolgversprechend. Die extrahierten Frequenzen stimmen besser als 1.2% mit verfügbaren Literaturwerten überein, allerdings sind vor einer abschließenden Bewertung noch einige offene Fragen zu klären.

Contents

Introduction

The intention of this work is the development of a code for general relativistic ideal hydrodynamics in up to three dimensions, based on regular numerical grids and the Eulerian approach. The envisioned fields of application are scenarios involving compact objects where general relativistic effects are strong, in particular the simulation of single and binary neutron star models.

Numerical simulations of compact objects are a field of active research. The availability of gravitational wave detectors, such as Geo600 [2] and LIGO [3], creates a demand for theoretical models of gravitational wave sources. Potential sources include black hole and neutron star mergers, but possibly also oscillations of single (proto-) neutron stars. Future observations of gravitational waves emitted by neutron star oscillations could provide insights into the equation of state of cold nuclear matter at high densities, as detailed in [6]. Recently, there is also growing evidence for direct observations of magnetar oscillations in the soft gamma ray emission during giant flares, see [60], which are attributed to the solid stellar crust and possibly magnetic field modes, see [51, 52].

To meet this demand, a number of codes for general relativistic hydrodynamics has been developed in recent years, which can be divided into general purpose codes and codes specialised on single star oscillations in various approximations, e.g. the ones described in [5, 42, 24, 29, 31]. Currently there exist only a few general purpose codes capable of 3D simulations in General Relativity, e.g. [16, 23, 48], which are used in the context of supernova core collapse [14, 15], neutron star mergers [48], neutron star collapse to a black hole [8], as well as neutron star oscillations and instabilities [17, 22, 23, 49]. Besides those mesh based codes, there exists a relativistic smoothed particle hydrodynamics (SPH) code [37], which is used for neutron star mergers.

Usually such codes are split into a part evolving the spacetime and one evolving the hydrodynamic equations. The code developed during my thesis, called "`Pizza`", numerically evolves the hydrodynamic equations on an arbitrary spacetime. It is implemented using the Cactus computational toolkit, see [1], which is a framework for large scale numerical simulations. For this framework, there exists a generalised interface for the communication between spacetime and hydrodynamic evolution codes, in a way that both can be developed independently and different codes can be combined. This makes it possible to couple the `Pizza` code with any spacetime evolution code using that interface, and to perform hydrodynamic simulations in General Relativity.

In a general purpose code, no specialising assumptions should be made. However, it is possible to modify a numerical scheme such that higher accuracy is achieved in scenarios where certain assumptions *do* hold. The numerical scheme I developed in the course of this work is tuned in a way that a class of stationary solutions, containing rigidly rotating cold neutron stars, is evolved with particular accuracy. For this, I derived a special formulation of the hydrodynamic evolution equations.

Besides the advantage for single star simulations, the hope is that also quasi-stationary, nearly-isentropic systems will benefit, albeit to a lesser extent. Evolving stationary solutions where gravity and pressure forces cancel has always been slightly problematic, even in Newtonian simulations. In Newtonian hydrodynamics, there exist a number of approaches to deal with such systems without linearisation, see [30] and the references therein. In general relativistic hydrodynamics, the `Pizza` code presents, to the author's knowledge, the first attempt in that direction.

A natural testbed for the hydrodynamic part of a general relativistic code are oscillations of single neutron stars in the Cowling approximation, that is, with a fixed spacetime metric. Recent results from 2D axisymmetric simulations in Cowling approximation are presented in [21, 54]. Since those results are obtained with a different numerical scheme and coordinate system, the comparison to the ones presented here provides a meaningful validation.

This thesis is structured as follows: In the first part, some analytic background is reviewed and, more important, a special formulation of the hydrodynamic evolution equations is derived. In the second part, the numerical methods used by the `Pizza` code are described. In particular, I present a new high resolution shock capturing (HRSC) scheme based on the aforementioned formulation of the hydrodynamic equations. In the third part, results from numerical tests are shown, involving shock tubes, nonrotating and rigidly rotating neutron stars with a polytropic or ideal gas EOS in Cowling approximation, and a specially designed testbed similar to a neutron star but without a fluid-vacuum boundary. Finally, I give accurate frequencies and eigenfunctions for different oscillation modes of two neutron star models. Additionally, first simulations with coupled spacetime evolution are shown. Basic knowledge of General Relativity and numerical methods is assumed. Otherwise, I refer the reader to the textbooks [18, 36, 38, 46, 57, 58].

Chapter 1

Analytical background

In the following, the basic notation and fundamental equations used in this work are introduced. For all equations, geometric units are used, i.e. units for which $G = c = 1$ holds. The 4-metric of the spacetime is denoted by g_{ab}, and the signature of g_{ab} is $(-,+,+,+)$. Indices a, b, c, etc. generally range from 0 to 3. In geometric units, the Einstein equations read

$$G_{ab} = 8\pi T_{ab} \tag{1.1}$$

where T_{ab} is the stress-energy-tensor and

$$G_{ab} = R_{ab} - \frac{1}{2} R g_{ab} \tag{1.2}$$

is the Einstein-tensor. R_{ab} is the Ricci-tensor belonging to the spacetime metric g_{ab}. It contains derivatives of g_{ab} up to second order in space and time. From the field equations (1.1) and the contracted Bianchi identities, see [58], follows the local conservation of energy and momentum

$$\nabla_a T^{ab} = 0 \tag{1.3}$$

where ∇_a is the covariant derivative.

In the context of this work, the matter consists of an ideal fluid, which additionally satisfies the mass conservation law

$$\nabla_a (\rho u^a) = 0 \tag{1.4}$$

where u is the 4-velocity of the fluid and ρ the restmass density in the fluid's rest frame. The stress-energy tensor of an ideal fluid is given by

$$T^{ab} = \rho h u^a u^b + P g^{ab} \tag{1.5}$$

where P is the pressure and h is the specific relativistic enthalpy defined as

$$h = 1 + \epsilon + \frac{P}{\rho} \tag{1.6}$$

with the specific internal energy ϵ. The fluid is further assumed to possess either one or two internal degrees of freedom, satisfying an equation of state (EOS) of the form

$$P = P(\rho, \epsilon) \quad \text{or} \quad P = P(\rho) \tag{1.7}$$

The numerical solution of Eqs. (1.3) and (1.4) for an arbitrary given spacetime is the goal of this thesis, while the simultaneous integration of Eq. (1.1) for a given stress-energy tensor is left to another code.

1.1 3+1 split

The covariant formulation of the field and hydrodynamic equations is not suitable for numerical computations. What is needed instead is a formulation where a state defined on a 3-dimensional space is evolved in time. Unfortunately, the Newtonian concept of an universal lapse of time is not valid in General Relativity. However, one can choose a 4-dimensional coordinate system such that the coordinate lines of the “time” coordinate are timelike, and the coordinate lines of the “spatial” coordinates are spacelike. Hypersurfaces of constant coordinate time are denoted Σ_t in the following. Given such coordinates, it is possible to define quantities on Σ_t for which the covariant equations yield evolution equations with respect to coordinate time. This approach is called 3+1-split.

We will now introduce the 3+1-split variables which replace the spacetime metric. The coordinate time is denoted by x^0, the spatial coordinates by x^i. Indices i, j, k, etc. range from 1 to 3, and n is the unit vector field orthogonal to Σ_t, that is $n^a n_a = -1, n_i = 0$. Observers with worldlines tangential to n are called normal observers throughout this work. For normal observers, Σ_t locally coincides with what they would call “now”. Defining $t = (1, 0, 0, 0)$, we can write

$$t^a = \alpha n^a + \beta^a \tag{1.8}$$

where α is the so called lapse function, and β is the shift vector, which is tangential to Σ_t, i.e. $\beta^0 = 0$ and $n_a \beta^a = 0$.

The lapse function α gives the ratio between the lapse of physical time measured by a normal observer and the lapse of coordinate time. Given the spacetime and one hypersurface, one can construct the neighbouring hypersurfaces using the lapse function. The shift vector field β^i on the other hand tells us how the given 3-dimensional coordinate systems on neighbouring Σ_t are connected: After transporting the coordinate system on one hypersurface along the normal vector field n onto a neighbouring hypersurface, one has to shift the resulting coordinate system along the shift vector β^i.

Lapse and shift are gauge quantities, which can be freely specified. The lapse function is connected to the acceleration of the normal observers via

$$n^a \nabla_a n_i = \partial_i \ln \alpha, \quad n^b n^a \nabla_a n_b = 0 \tag{1.9}$$

This implies that for a constant lapse function, the normal observers follow a geodesic, i.e. they are free falling. Therefore a constant lapse is a bad choice in most situations.

The 4-metric g_{ab} can be expressed in terms of lapse, shift, and the 3-metric g_{ij} on Σ_t as

$$g_{ab} = \begin{pmatrix} -\alpha^2 + \beta^k \beta_k & \beta_j \\ \beta_i & g_{ij} \end{pmatrix} \tag{1.10}$$

Its determinant is given by

$$\sqrt{g} \equiv \sqrt{-\det(g_{ab})} = \alpha \sqrt{d} \tag{1.11}$$

where

$$\sqrt{d} \equiv \sqrt{\det(g_{ij})} \tag{1.12}$$

is the physical volume per coordinate volume.

Another fundamental quantity is the extrinsic curvature K_{ij}, which is a special case of the second fundamental form Π, which in turn is defined as follows: Given a submanifold S of some manifold M, and vector fields u, v tangential to S, one can split covariant derivatives as

$$v^a \nabla_a w^b = v^a \nabla^S_a w^b + v^a w^c \Pi^b_{ac} \tag{1.13}$$

The part of the covariant derivative tangential to S is given by ∇^S, which is the covariant derivative defined on the submanifold. The orthogonal part is given by the second fundamental form Π^b_{ac}, which is a symmetric tensor.

Inserting the spacetime for M and a hypersurface of constant coordinate time for S, one can write

$$v^a w^c \Pi^b_{ac} \equiv -v^i w^j n^b K_{ij} \tag{1.14}$$

The extrinsic curvature K_{ij} is a symmetric 3-tensor defined on Σ_t. It depends only on the choice of the hypersurfaces.

The extrinsic curvature specifies how strong the hypersurfaces of constant coordinate time are bend, in the following sense: If we assume that Σ_t contains a spacelike geodesic curve $c(s)$, it follows from the geodesic equation

$$\dot{c}^a \nabla_a \dot{c}^b = 0, \qquad \dot{c}^a \equiv \frac{dc^a}{ds} \tag{1.15}$$

together with Eqs. (1.13) and (1.14) that

$$\dot{c}^i \dot{c}^j K_{ij} = 0 \tag{1.16}$$

Dealing with the curvature of Σ_t, the extrinsic curvature tensor also contains information about the derivatives of the normal vector along Σ_t. Given two vector fields v, w tangential to Σ_t, one can write

$$\begin{aligned} v^a w_b \nabla_a n^b &= v^a(\partial_a(\underbrace{w_b n^b}_{=0}) - n_b \nabla_a w^b) &(1.17)\\ &= v^i w^j K_{ij} &(1.18)\\ \Rightarrow \nabla_i n_j &= K_{ij} &(1.19)\end{aligned}$$

Since K_{ij} specifies how the direction of the normal vectors n change along Σ_t, it also tells us how the physical distance of neighbouring normal observers changes along their trajectories. K_{ij} is therefore closely related to the derivative of the 3-metric with respect to coordinate time. One can show that

$$(\partial_0 - L_\beta) g_{ij} = -2\alpha K_{ij} \tag{1.20}$$

where $L_\beta g_{ij}$ is the Lie-derivative (see [38, 39]) of the 3-metric along the shift vector. The term $L_\beta g_{ij}$ accounts for the change of the metric components due to the shift of the coordinate system along β.

We now apply the 3+1-split to the hydrodynamic quantities. The 4-velocity of the fluid can be written as

$$u^a = \gamma \left(n^a + v^a\right), \quad v^0 = 0 \tag{1.21}$$

where v^i is the 3-velocity in the reference frame of a normal observer, and

$$\gamma = \frac{1}{\sqrt{1 - v^i v_i}} \tag{1.22}$$

is the corresponding Lorentz factor. Using Eq. (1.8), one can also write

$$u^a = \frac{\gamma}{\alpha}(1, w^i) \tag{1.23}$$

where

$$w^i = \alpha v^i - \beta^i \tag{1.24}$$

is the advection speed with respect to the coordinates, i.e. $w^i = \frac{u^i}{u^0}$.

The quantities that will be evolved in my simulations are energy, momentum, and restmass coordinate-density in the local reference frame of normal observers. Coordinate-density means amount per volume in coordinate space, in contrast to amount per physical volume. The mass (coordinate-) density is given by

$$D = \sqrt{d}\gamma\rho \tag{1.25}$$

The factor γ takes into account the Lorentz contraction, while the factor $\sqrt{d}$ converts physical density to coordinate-density. The energy density is given by

$$E = \sqrt{d} n^a n^b T_{ab} = \sqrt{d}\left(\gamma^2 \rho h - P\right) \tag{1.26}$$

For numerical simulations it is beneficial to substract the restmass density, defining

$$\tau = E - D = \sqrt{d}\left(\gamma^2 \rho h - P - \gamma\rho\right) \tag{1.27}$$

Finally, the momentum density is given by

$$S_i = -\sqrt{d} n^a T_{ai} = \sqrt{d}\gamma^2 \rho h v_i \tag{1.28}$$

1.2 Hydrodynamic evolution equations

As shown in [33, 10], the hydrodynamic evolution equations for an ideal fluid on a given curved spacetime can be written as a system of quasi-conservation laws

$$\partial_0 q = -\partial_i f^i + s \tag{1.29}$$

for the (quasi-) conserved quantities $q = (D, \tau, S_i)$ defined in Sec. 1.1. The flux functions $f^i = (f^i_D, f^i_\tau, f^i_{S_j})$ are given by

$$\begin{aligned} f^i_D &= w^i D && (1.30)\\ f^i_\tau &= w^i \tau + \sqrt{g} v^i P && (1.31)\\ f^i_{S_j} &= w^i S_j + \sqrt{g} P \delta^i_j && (1.32) \end{aligned}$$

The source terms $s = (0, s^\tau, s^{S_i})$ given in [10] can be written as

$$\begin{aligned} s^{S_i} &= -E\partial_i\alpha + \alpha P \partial_i \sqrt{d} + S_l \partial_i \beta^l + \frac{1}{2}\alpha\sqrt{d}\gamma^2 \rho h v^l v^k \partial_i g_{kl} && (1.33)\\ s^\tau &= \alpha\sqrt{d} P K - S^l \partial_l \alpha + \alpha\sqrt{d}\gamma^2 \rho h v^l v^k K_{lk} && (1.34) \end{aligned}$$

This formulation, with minor changes regarding the way in which the source terms are computed, is used in most recent general relativistic hydro codes. For a detailed review on general relativistic hydrodynamics, see [20].

In contrast to other formulations, e.g. [61], the usage of the conservative form makes it possible to apply modern HRSC schemes. However, such methods only compute the flux terms. The source terms have to be computed independently, e.g. by inserting finite difference approximations for derivatives of the field quantities in Eqs. (1.33) and (1.34).

In stationary cases, the pressure forces are contained in the flux terms and the gravitational forces in the source terms. Both contributions cancel, and the net forces present in oscillations around the equilibrium are much smaller than the equilibrium pressure or gravitational forces themselves. The numerical error on the other hand is roughly proportional to the pressure and gravitational forces. Since flux and source terms are computed with different methods, the errors usually do not cancel. The net forces are therefore computed with a big relative error.

To circumvent this problem, I constructed a new numerical scheme where source and flux terms are treated more consistent. The new scheme is build on a new formulation of the source terms in the evolution equations, which I derived especially for this purpose, and which is presented in the following.

1.2.1 Preparation

To derive the new formulation of the evolution equations, we start with the stress energy tensor of an ideal fluid:

$$T^a{}_b = \rho h u^a u_b + P\delta^a{}_b \equiv A^a{}_b + P\delta^a{}_b \tag{1.35}$$

The covariant divergence of A is given by

$$\nabla_a A^a{}_b = \partial_a A^a{}_b + \Gamma^a_{ac} A^c{}_b - \Gamma^c_{ab} A^a{}_c \tag{1.36}$$

The last term can be simplified to

$$\begin{aligned}\Gamma^c_{ab} A^a{}_c &= A^a{}_c \frac{1}{2} g^{cd} \left(\partial_a g_{db} + \partial_b g_{da} - \partial_d g_{ab}\right) &(1.37)\\ &= \frac{1}{2} A^{ad} \left(\partial_a g_{db} + \partial_b g_{da} - \partial_d g_{ab}\right) &(1.38)\\ &= \frac{1}{2} A^{ad} \partial_b g_{ad} &(1.39)\end{aligned}$$

where we have used $A^{ad} = A^{da}$ and the definition of the Christoffel symbols Γ^c_{ab}. Together with the identity

$$\Gamma^a_{ac} = \frac{1}{\sqrt{g}} \partial_c \sqrt{g} \tag{1.40}$$

we obtain

$$\begin{aligned}
\nabla_a T^a{}_b &= \partial_a A^a{}_b + \frac{1}{\sqrt{g}} A^c{}_b \partial_c \sqrt{g} \\
&\quad - \frac{1}{2} A^{ad} \partial_b g_{ad} + \partial_b P \qquad (1.41) \\
&= \frac{1}{\sqrt{g}} \partial_a \left(\sqrt{g} A^a{}_b \right) - \frac{1}{2} A^{ad} \partial_b g_{ad} + \partial_b P \qquad (1.42)
\end{aligned}$$

For solutions of the Einstein equations, $\nabla_a T^a{}_b = 0$ and hence

$$0 = \partial_a \left(\sqrt{g} \rho h u^a u_b \right) - \frac{1}{2} \sqrt{g} \rho h u^a u^d \partial_b g_{ad} + \sqrt{g} \partial_b P \qquad (1.43)$$

By applying a 3+1 split on this equation, one could obtain evolution equations for energy and momentum density. We will use a slightly different strategy here and rewrite the equations locally at one point using special functions.

1.2.2 Equilibrium functions

We now construct a set of functions labelled “equilibrium functions”, which will become the central part of the new scheme. For an arbitrary chosen point Q, they are defined on a neighbourhood of Q as follows:

- An advection speed which is constant in space

$$\hat{w}^i \equiv w^i(Q) \qquad (1.44)$$

- The corresponding 4-velocity and Lorentz factor

$$\hat{u}^a = \frac{\hat{\gamma}}{\alpha}(1, \hat{w}^i), \quad \hat{u}^a \hat{u}_a = -1 \qquad (1.45)$$

- A new set of fluid variables, which are identical to the original ones at point Q, i.e.

$$\hat{\rho} = \rho, \quad \hat{s} = s, \quad \hat{P} = P, \quad \hat{h} = h \quad \Big|_Q \qquad (1.46)$$

- To define the new fluid variables elsewhere, we demand

$$\begin{aligned}
\hat{h}\frac{\alpha}{\hat{\gamma}} &= \text{const} \qquad (1.47) \\
\hat{s} &= \text{const} \qquad (1.48)
\end{aligned}$$

  Further, the new fluid variables have to satisfy the same EOS as the original ones.

For later use we rewrite Eq. (1.47). Constant specific entropy (1.48) implies

$$\frac{d\hat{h}}{d\hat{P}} = \frac{1}{\hat{\rho}} \tag{1.49}$$

and hence

$$\frac{\partial_i \hat{P}}{\hat{\rho}\hat{h}} = \frac{d\hat{h}}{d\hat{P}}\frac{\partial_i \hat{P}}{\hat{h}} = \frac{\partial_i \hat{h}}{\hat{h}} = \partial_i \ln(\hat{h}) \tag{1.50}$$

Eq. (1.47) therefore becomes

$$\frac{\partial_i \hat{P}}{\hat{\rho}\hat{h}} = -\partial_i \ln \frac{\alpha}{\hat{\gamma}} \tag{1.51}$$

This definition is motivated by the case of a rigidly rotating isentropic star in corotating coordinates, where we have $\hat{w}^i = 0$ and $\hat{v}^i = \beta^i/\alpha$. In the Newtonian limit, $h \approx 1, \quad \alpha \approx 1, \quad |\hat{v}| = \Omega d \ll 1$ (where d is the distance to the rotation axis, and Ω the angular velocity), and the gravitational potential is $U \approx \alpha - 1$. In that limit Eq. (1.51) becomes

$$\partial_i \hat{P} = \hat{\rho}\partial_i \left(\frac{1}{2}\Omega^2 d^2 - U\right) \tag{1.52}$$

which is the condition for hydrostatic equilibrium in a centrifugal - gravitational potential. As shown in [12], Eq. (1.47) is indeed the necessary condition for rigidly rotating isentropic fluid bodies in general relativity.

In general, the functions defined here do not correspond to a stationary solution. However, in generic cases they still possess a weaker property we will exploit: As shown in the next subsections, the evolution equations (1.29) reduce at one instant of time (almost) to simple advection equations if applied to equilibrium functions.

1.2.3 Momentum equation

Using the previously defined equilibrium functions, Eq. (1.43) can be simplified. From Eq. (1.45) we have

$$\frac{\alpha}{\hat{\gamma}}\hat{u}^a = (1, \hat{w}^1, \hat{w}^2, \hat{w}^3) = \text{const} \tag{1.53}$$

and therefore

$$\begin{aligned} -\frac{1}{2}\hat{u}^a\hat{u}^d\partial_i g_{ad} &= -\frac{\hat{\gamma}^2}{2\alpha^2}\partial_i\left(\frac{\alpha^2}{\hat{\gamma}^2}\hat{u}^a\hat{u}^d g_{ad}\right) && (1.54)\\ &= \frac{\hat{\gamma}^2}{2\alpha^2}\partial_i\left(\frac{\alpha^2}{\hat{\gamma}^2}\right) && (1.55)\\ &= \partial_i \ln\left(\frac{\alpha}{\hat{\gamma}}\right) && (1.56) \end{aligned}$$

At point Q we have $u = \hat{u}$ and, setting index $b = i$, we can reformulate Eq. (1.43) as

$$0 = \partial_a \left(\sqrt{g}\rho h u^a u_i\right) + \sqrt{g}\rho h \partial_i \ln\left(\frac{\alpha}{\hat{\gamma}}\right) + \sqrt{g}\partial_i P \quad \Big|_Q \qquad (1.57)$$

With identities (1.51) and (1.46) we obtain

$$\begin{aligned} 0 &= \partial_a \left(\sqrt{g}\rho h u^a u_i\right) + \sqrt{g}\partial_i \left(P - \hat{P}\right) \quad \Big|_Q && (1.58)\\ &= \partial_a \left(\sqrt{g}\rho h u^a u_i\right) + \partial_i \left[\sqrt{g}\left(P - \hat{P}\right)\right] \quad \Big|_Q && (1.59)\\ &= \partial_a \left[\sqrt{g}\rho h u^a u_i + \sqrt{g}\left(P - \hat{P}\right)\delta^a_i\right] \quad \Big|_Q && (1.60)\end{aligned}$$

Inserting Eqs. (1.28), (1.11), (1.23), and $u_i = \gamma v_i$ then yields the final form of the momentum equation

$$\boxed{\partial_0 S_i = -\partial_j \left[S_i w^j + \sqrt{g}\left(P - \hat{P}\right)\delta^j_i\right] \quad \Big|_Q} \qquad (1.61)$$

The flux terms are identical to Eq. (1.32), but the source terms are now encoded in the derivatives of $\hat{P}$. Note also this equation is only valid at one point, unless the fluid variables are equilibrium functions themselves and hence identical to the ones constructed around Q. In that case all steps of the derivation above are valid everywhere and Eq. (1.61) reduces to a simple advection equation with constant advection speed, i.e.

$$\partial_0 S_i = -\hat{w}^j \partial_j S_i \qquad (1.62)$$

The same holds trivially for the evolution equation of the conserved rest mass density D, since it contains only advection terms.

Note that isentropic systems which are manifestly stationary in comoving coordinates, for example rigidly rotating isentropic stars, correspond to equilibrium functions. This can be shown directly from Eqs. (1.61) and (1.51) when using $w^i = 0$, the assumption of isentropy, and the fact that Q was arbitrary.

1.2.4 Energy equation

A similar treatment can be given for the energy equation, although it is not possible to rewrite the source terms completely using equilibrium functions, if one wants to avoid explicit dependence on time derivatives of lapse and shift. As shown below, the energy equation can be written as

$$\boxed{\partial_0 \tau = -\partial_i \left[\tau w^i + \sqrt{g}\left(v^i P - \hat{v}^i \hat{P}\right)\right] + r \quad \Big|_Q} \qquad (1.63)$$

where

$$r = -\frac{\alpha}{\gamma} u^a \left(\frac{1}{2} \sqrt{d} \gamma^2 \rho h v^i v^j \partial_a g_{ij} + P \partial_a \sqrt{d} \right) \tag{1.64}$$

Again, if the system corresponds to an equilibrium function, the derivation is valid everywhere and the equation reduces to an advection equation plus the remaining terms r. These terms vanish if the 3-metric components are constant along the fluid worldlines. For instance, this is the case for a stationary rotating star (without meridional motion) in suitable coordinates. In the Newtonian limit with Cartesian coordinates, they can also be neglected.

To derive Eq. (1.63), we make use of the expressions

$$n^0 = \frac{1}{\alpha}, \quad n^i = -\frac{1}{\alpha} \beta^i \quad , n_i = 0 \tag{1.65}$$

$$u^a = \gamma \left(n^a + v^a \right), \quad n^b u_b = -\gamma, \quad v^0 = 0 \tag{1.66}$$

as well as

$$\begin{aligned}
& n^c \left(n^b + 2v^b \right) \partial_a g_{cb} \\
= \;& \partial_a \left[n_b \left(n^b + 2v^b \right) \right] - n_b \partial_a \left(n^b + 2v^b \right) \\
& - \left(n_b + 2v_b \right) \partial_a n^b & (1.67) \\
= \;& \partial_a(-1) - 2 \left(n_b + v_b \right) \partial_a n^b & (1.68) \\
= \;& -\frac{2}{\gamma} u_b \partial_a n^b & (1.69)
\end{aligned}$$

The following calculation is valid only at point Q. By contracting Eq. (1.43) with n^b, we get

$$\begin{aligned}
0 \;=\;& \underbrace{n^b \partial_a \left(\sqrt{g} \rho h u^a u_b \right)}_{a_1} \underbrace{- n^b \frac{1}{2} \sqrt{g} \rho h u^a u^d \partial_b g_{ad}}_{a_2} \\
& \underbrace{+ n^b \sqrt{g} \partial_b P}_{a_3} & (1.70) \\
a_1 \;=\;& -\partial_a \left(\sqrt{g} \gamma \rho h u^a \right) - \sqrt{g} \rho h u^a u_b \partial_a n^b & (1.71) \\
=\;& \underbrace{-\partial_0 \left(\sqrt{d} \gamma^2 \rho h \right) - \partial_i \left(\sqrt{d} \gamma^2 \rho h w^i \right)}_{a_4} \\
& \underbrace{- \sqrt{g} \rho h u^a u_b \partial_a n^b}_{a_5} & (1.72)
\end{aligned}$$

Using Eqs. (1.51) and (1.56), a_2 becomes

$$\begin{aligned}
a_2 &= -\left(\frac{1}{\gamma}u^b - v^b\right)\frac{1}{2}\sqrt{g}\rho h u^a u^d \partial_b g_{ad} && (1.73)\\
&= \underbrace{-\frac{1}{2\gamma}u^b\sqrt{g}\rho h u^a u^d \partial_b g_{ad}}_{a_6}\underbrace{+\, v^i\sqrt{g}\partial_i \hat{P}}_{a_7} && (1.74)
\end{aligned}$$

With Eq. (1.69), we then find

$$\begin{aligned}
a_6 + a_5 &= -\sqrt{g}\rho h u^a\left(u_b\partial_a n^b + \frac{1}{2\gamma}u^c u^b \partial_a g_{cb}\right) && (1.75)\\
&= -\sqrt{g}\rho h u^a\Big[u_b\partial_a n^b \\
&\quad + \frac{\gamma}{2}\left(n^c n^b + 2n^c v^b + v^c v^b\right)\partial_a g_{cb}\Big] && (1.76)\\
&\overset{(1.69)}{=} -u^a\frac{1}{2}\sqrt{g}\gamma\rho h v^i v^j \partial_a g_{ij} && (1.77)
\end{aligned}$$

$$\begin{aligned}
a_3 + a_7 &= \frac{1}{\alpha}\sqrt{g}\partial_0 P - \frac{\beta^i}{\alpha}\sqrt{g}\partial_i P + v^i\sqrt{g}\partial_i\hat{P} && (1.78)\\
&= \sqrt{d}\beta^i\partial_i\left(\hat{P} - P\right) + w^i\sqrt{d}\partial_i\hat{P} + \sqrt{d}\partial_0 P && (1.79)\\
&= \partial_i\left[\sqrt{d}\beta^i\left(\hat{P} - P\right) + \sqrt{d}\hat{w}^i\hat{P}\right]\\
&\quad - w^i P\partial_i\sqrt{d} + \sqrt{d}\partial_0 P && (1.80)\\
&= \partial_i\left[\sqrt{d}\alpha\left(\hat{v}^i\hat{P} - v^i P\right) + \sqrt{d}w^i P\right]\\
&\quad - w^i P\partial_i\sqrt{d} + \sqrt{d}\partial_0 P && (1.81)\\
&= \partial_0\left(\sqrt{d}P\right) - P\partial_0\sqrt{d} - w^i P\partial_i\sqrt{d}\\
&\quad + \partial_i\left[\sqrt{d}\alpha\left(\hat{v}^i\hat{P} - v^i P\right) + \sqrt{d}w^i P\right] && (1.82)\\
&= \partial_0\left(\sqrt{d}P\right) - \frac{\alpha}{\gamma}u^a P\partial_a\sqrt{d}\\
&\quad + \partial_i\left[\sqrt{d}\alpha\left(\hat{v}^i\hat{P} - v^i P\right) + \sqrt{d}w^i P\right] && (1.83)
\end{aligned}$$

Recombining these terms yields

$$
\begin{aligned}
0 &= a_3 + a_7 + a_6 + a_5 + a_4 & (1.84)\\
&= -\partial_0 \left[\sqrt{d}\left(\gamma^2 \rho h - P\right)\right] \\
&\quad - \partial_i \Big[\sqrt{d}\left(\gamma^2 \rho h - P\right) w^i \\
&\qquad\qquad + \alpha\sqrt{d}\left(v^i P - \hat{v}^i \hat{P}\right)\Big] \\
&\quad - \frac{\alpha}{\gamma} u^a \left(\frac{1}{2}\sqrt{d}\gamma^2 \rho h v^i v^j \partial_a g_{ij} + P \partial_a \sqrt{d}\right) & (1.85)
\end{aligned}
$$

Using the rest mass conservation

$$
0 = \partial_0 \left(\sqrt{d}\gamma\rho\right) + \partial_i \left(w^i \sqrt{d}\gamma\rho\right) \tag{1.86}
$$

finally gives the desired result.

Up to this point, we have defined some "equilibrium" states for which the evolution equations simplify to an advection equation, and used these states to reformulate the source terms of the evolution equations. There is no one-to-one correspondence between the equilibrium functions and stationary solutions. However, isentropic systems which are manifestly stationary in comoving coordinates are given by equilibrium functions.

1.3 Spacetime evolution equations

As shown by [7], the Einstein equations can be formulated as evolution equations of the 3-metric g_{ij} and the extrinsic curvature K_{ij}, which are usually referred to as ADM (Arnowitt, Deser, Misner) variables. In detail,

$$
\begin{aligned}
\left(\partial_0 - L_\beta\right) g_{ij} &= -2\alpha K_{ij} & (1.87)\\
\left(\partial_0 - L_\beta\right) K_{ij} &= -\nabla_i \nabla_j \alpha + \alpha \left(R^{(3)}_{ij} + K K_{ij} - 2K_{ik}K^k{}_j - R_{ij}\right) & (1.88)
\end{aligned}
$$

where $R^{(3)}_{ij}$ is the Ricci-tensor belonging to the 3-metric on Σ_t, and R_{ij} are the spatial components of the 4-dimensional Ricci tensor. Using the field equations (1.1), R_{ij} can be written as

$$
R_{ij} = 8\pi \left(T_{ij} - \frac{1}{2} T^a_a g_{ij}\right) \tag{1.89}
$$

In addition to the evolution equations, the field equations yield the constraint equations (compare [23])

$$0 = H \equiv \frac{1}{16\pi}\left(R^{(3)} + K^2 - K_{ij}K^{ij}\right) - n^a n^b T_{ab} \tag{1.90}$$

$$0 = M_i \equiv \frac{1}{8\pi}\left(\nabla_j K_i^j - \nabla_i K\right) + n^a T_{ai} \tag{1.91}$$

which have to be satisfied on every hypersurface Σ_t. Analytically, the constraints are propagated by the evolution equations, given they are fulfilled initially. In numerical simulations however, small errors inevitably induce constraint violations. As soon as the constraints are violated, they can grow analytically. Wether they do depends on the specific formulation of the equations. For example, one can add various combinations of the constraints to the evolution equations. Although it is possible to enforce the constraint equations during evolution, it requires the solution of elliptic equations, see [27], which is numerically expensive and very complicated in conjunction with mesh refinement or excision techniques. Therefore, most 3D spacetime evolution codes use unconstrained evolution.

The search for a numerically stable formulation lead to alternative formulations of the original ADM system. The current state-of-the-art formulation is the so called BSSN (Baumgarte, Shapiro, Shibata, Nakamura) system described in [50, 11], where the evolved variables are the following:

- The conformal factor ϕ, defined as

$$e^{4\phi} = \det(g_{ij})^{1/3} \tag{1.92}$$

- The conformally rescaled 3-metric

$$\tilde{\gamma}_{ij} = e^{-4\phi} g_{ij} \tag{1.93}$$

- The trace K of the extrinsic curvature
- The conformally rescaled trace-free part of the extrinsic curvature

$$\tilde{A}_{ij} = e^{-4\phi}\left(K_{ij} - \frac{1}{3}K g_{ij}\right) \tag{1.94}$$

- Optionally, the so called conformal connection functions

$$\tilde{\Gamma}^i = -\tilde{\gamma}^{ij}_{,j} \tag{1.95}$$

There are some variants of the evolution equations for the BSSN variables, which differ mainly in the way the constraints are used. The evolution equations used for the coupled spacetime simulations shown in Sec. 4.2 can be found in [4]. Note that the promotion of the quantities ϕ, K to independent variables creates additional, albeit trivial, constraints. For the simulations in Sec. 4.2, they are enforced after each timestep, in contrast to the ADM constraints.

1.4 Spherical neutron stars

The main testcase for the `Pizza` code is a spherically symmetric, nonrotating static neutron star with a cold equation of state. In this work, I use simple star models where magnetic fields and the existence of a solid crust are neglected. The structure of such stars has been investigated by [40]. In the following I briefly review the basic equations, and provide the transformation formulas used to obtain initial data in various coordinate systems. For the more complicated case of rotating stars, I refer the reader to [53]. For a general introduction to the astrophysics of neutron stars, see [47].

In spherical coordinates $x^i \equiv (r, \vartheta, \varphi)$, the spacetime of a static spherical star is given by the line element

$$ds^2 = -e^{2\nu(r)}dt^2 + e^{2\lambda(r)}dr^2 + r^2\left(d\vartheta^2 + \sin^2(\vartheta)d\varphi^2\right) \qquad (1.96)$$

or, alternatively by

$$g_{ij} = \begin{pmatrix} e^{2\lambda} & 0 & 0 \\ 0 & r^2 & 0 \\ 0 & 0 & r^2\sin^2(\vartheta) \end{pmatrix} \qquad (1.97)$$

$$\alpha = e^{\nu} \qquad (1.98)$$

$$\beta^i = 0 \qquad (1.99)$$

Obviously, r is the circumferential radius. Specialising the argumentation in Sec. 1.2 to this case, it turns out that $\nu = \ln(\alpha)$ replaces the gravitational potential in Newtonian gravity. With the ansatz for the line element, the field equations yield the Tolman-Oppenheimer-Volkoff (TOV) equations

$$m' = 4\pi r^2\rho(1+\epsilon) \qquad (1.100)$$

$$\nu' = \left(\frac{m}{r^2} + 4\pi P r\right)\left(1 - \frac{2m}{r}\right)^{-1} \qquad (1.101)$$

$$P' = -\rho h \nu' \qquad (1.102)$$

where m is defined by

$$e^{2\lambda(r)} \equiv \left(1 - \frac{2m(r)}{r}\right)^{-1} \tag{1.103}$$

So far ν is determined only up to a constant, which can be found by matching the solution to the Schwarzschild metric in the exterior.

We now investigate the behaviour of the solution near the stellar surface, which is a problematic region for numerical computations. Near the surface, we have

$$\epsilon \to 0, \qquad \rho \to 0, \qquad P \to 0, \qquad h \to 1 \tag{1.104}$$

In this limit, the first two TOV equations become

$$m' = 0 \tag{1.105}$$

$$\nu' = \frac{m}{r^2}\left(1 - \frac{2m}{r}\right)^{-1} \tag{1.106}$$

Note Eq. (1.102) is the same as Eq. (1.51), and hence Eq. (1.47), for the case $\gamma = 1$. Therefore $h' = -\nu' \neq 0$ at the surface. For the polytropic case where $h = 1 + \Gamma\epsilon$, it follows that the specific energy density approaches zero linearly, i.e.

$$\epsilon' = -\frac{\nu'}{\Gamma} \tag{1.107}$$

The behaviour of pressure and density follows from

$$\rho \sim \epsilon^{\frac{1}{\Gamma-1}}, \quad P \sim \epsilon^{\frac{\Gamma}{\Gamma-1}} \tag{1.108}$$

For the case $\Gamma = 2$, the density has kink at the surface, whereas the pressure is smooth.

For the numerical simulations shown in Sec. 3 and 4, we also need the metric in cylindrical and Cartesian coordinates. The components of the 3-metric with respect to new coordinates $\bar{x}^i$ are obtained by the transformation rule

$$\bar{g}_{kl} = \frac{\partial x^i}{\partial \bar{x}^k} g_{ij} \frac{\partial x^j}{\partial \bar{x}^l} \tag{1.109}$$

For cylindrical coordinates $\bar{x}^i \equiv (s, z, \varphi) = (r\sin(\vartheta), r\cos(\vartheta), \varphi)$, a straightforward calculation yields

$$g^c_{ij} = \begin{pmatrix} 1 + s^2 f & szf & 0 \\ szf & 1 + z^2 f & 0 \\ 0 & 0 & s^2 \end{pmatrix} \tag{1.110}$$

where $f = \left(e^{2\lambda} - 1\right)/r^2$. The metric determinant becomes

$$\sqrt{d} = e^{\lambda} s \tag{1.111}$$

Note that the volume element $\sqrt{d}$ becomes zero at the axis. This could lead to numerical problems if no precaution is taken, as discussed later.

For Cartesian coordinates, we obtain

$$g^c_{ij} = 1 + f \begin{pmatrix} x^2 & xy & xz \\ xy & y^2 & yz \\ xz & yz & z^2 \end{pmatrix} \tag{1.112}$$

and

$$\sqrt{d} = e^{\lambda} \tag{1.113}$$

By reparameterising the radial coordinate, it would be possible to obtain a diagonal metric in Cartesian as well as in cylindrical coordinates. However, for the purpose of testing the code, it is desirable to use a metric tensor where all components are non-zero.

Chapter 2

Numerical methods

In this chapter the numerical methods for relativistic hydrodynamics developed or used for this work are described. For the sake of brevity, I focus on the aspects that one would need to know in order to reproduce my results, and discuss technical details of the `Pizza` code only on a general level.

2.1 Time integration via Method of Lines

The "Method of Lines" is a procedure to separate the time discretisation from the spatial discretisation for systems of partial differential equations of the general form

$$\partial_t q(x,t) = g(x, q, \partial_i q) \tag{2.1}$$

where q can be a scalar or vector, and g could as well contain higher spatial derivatives of q. Suppose we already performed a discretisation in space, using a finite difference or finite volume method, obtaining a system of the form

$$\partial_t q = L[q] \tag{2.2}$$

where q is a discrete state vector, and L is a numerical approximation for g. The equation above is nothing but a large system of ordinary differential equations, which can be solved with standard methods.

For the star simulations in Sec. 3, a 2nd order Runge-Kutta method (method of Heun, see [45]) is used, which works as follows: To evolve a state q^n in time over a timestep Δt, resulting in a new state q^{n+1}, one computes

$$q^a = q^n + \Delta t L[q^n] \tag{2.3}$$

$$q^b = q^a + \Delta t L[q^a] \tag{2.4}$$

$$q^{n+1} = \frac{1}{2}\left(q^n + q^b\right) \tag{2.5}$$

The Method of Lines has several advantages over traditional schemes which perform time and space discretisation simultaneously: The resulting scheme will be considerably simpler to implement due to the modular character; as a consequence, it is also easier to optimise; it is trivial to switch to another time integration scheme; last but not least, the evolution system can easily be split into several parts, for example a hydrodynamic and a spacetime part.

After choosing a time integrator, one still has to specify a suitable timestep. If explicit schemes are used for time integration, the timestep has to satisfy the condition

$$\Delta t < \sigma \frac{\Delta x}{v_m} \tag{2.6}$$

to allow for a stable evolution. Here, v_m is the maximal signal propagation speed, defined with respect to coordinate times and distances, Δx is the grid spacing, and σ is the Courant factor. The Courant factor depends on both the spatial discretisation and the time integrator. For complex schemes, it usually has to be determined empirically.

For the simulations in Cowling approximation, v_m is determined by the hydrodynamic evolution equations. The maximum signal propagation speed for this system is given in [23, 10]. Without Cowling approximation, the timestep is further restricted by the propagation speed of gravitational signals. In that case, a fixed timestep $\Delta t = \sigma \Delta x / c$ will be used. In principle, the coordinate velocity of gravitational signals is not exactly the speed of light, but in practice, the differences are absorbed in the Courant factor.

2.2 Space discretisation scheme

In this section, I give a prescription which allows one to modify a standard HRSC scheme such that source and flux terms in the hydrodynamic evolution equations are treated more consistently, utilising the formulation of the evolution equations shown in Sec. 1.2. This scheme is the unique feature of the `Pizza` code and a central part of this work.

2.2.1 Original scheme

First I describe the scheme to be modified, restricting the discussion initially to one dimension for simplicity. It is a standard finite volume HRSC scheme on a regular grid; see [57, 30] for an introduction to the field. The evolved quantities are the cell averages q_n of q. The time derivatives of the cell averages are written as

$$\partial_0 q_n = -\frac{1}{\Delta x}(F_{n+\frac{1}{2}} - F_{n-\frac{1}{2}}) + s_n \tag{2.7}$$

$$s_n = \left(0, s_n^\tau, s_n^{S_1}, s_n^{S_2}, s_n^{S_3}\right) \tag{2.8}$$

where lower indices n refer to quantities defined in cell n and indices $n-\frac{1}{2}$, $n+\frac{1}{2}$ to quantities defined at the left and right cell boundaries. The numerical fluxes $F_{n\pm\frac{1}{2}}$ are approximations to the flux functions f at the cell boundaries. $F_{n\pm\frac{1}{2}}$ and s_n are computed as described below, where $Y_n = (\rho, \epsilon, w^l)_n$ denotes a set of primitive variables obtained from q_n.

1. Assume constant states inside the cells and define left and right states at the cell boundaries

$$Y^r_{n-\frac{1}{2}} = Y^l_{n+\frac{1}{2}} \equiv Y_n$$

2. Refine this model by assuming piecewise linear states, where the slopes are computed with a slope limiter function L

$$\begin{aligned}
\bar{Y}^r_{n-\frac{1}{2}} &\equiv Y^r_{n-\frac{1}{2}} - \frac{1}{2}L(\Delta Y_{n-\frac{1}{2}}, \Delta Y_{n+\frac{1}{2}}) \\
\bar{Y}^l_{n+\frac{1}{2}} &\equiv Y^l_{n+\frac{1}{2}} + \frac{1}{2}L(\Delta Y_{n-\frac{1}{2}}, \Delta Y_{n+\frac{1}{2}}) \\
\Delta Y_{n-\frac{1}{2}} &\equiv Y^r_{n-\frac{1}{2}} - Y^l_{n-\frac{1}{2}}
\end{aligned}$$

We demand that a globally linear state is modelled exactly, i.e. L has to satisfy $L(a, a) = a$. The purpose of L is to avoid spurious oscillations near shocks, see [57]. Some choices for L are shown in Sec. 2.2.4.

3. Compute numerical fluxes at the resulting discontinuities using an approximative Riemann solver R

$$F_{n\pm\frac{1}{2}} = R(\bar{Y}^l_{n\pm\frac{1}{2}}, \bar{Y}^r_{n\pm\frac{1}{2}})$$

For the trivial Riemann problem, where both states are the same, R should be exact, i.e. $R(Y, Y) = f(Y)$.

4. Compute the source terms from

$$\begin{aligned}
s_n = {} & s(Y_n, (K_{ij})_n, \alpha_n, (\beta^i)_n, (D_j\sqrt{d})_n, \\
& (D_j g_{ik})_n, (D_j\alpha)_n, (D_j\beta^i)_n)
\end{aligned}$$

where D is a linear finite difference operator for the first derivative.

Values of lapse, shift and 3-metric needed at the cell boundaries are obtained by linear interpolation. If the fluid has only one internal degree of freedom, i.e. $P = P(\rho)$, the evolution equation for τ is redundant and therefore not computed.

The contributions for the other dimensions are computed the same way. All contributions, including the source terms, are added together and then used for the time integration via the Method of Lines.

2.2.2 Modification of the standard scheme

The modification I applied to the scheme changes only steps 1 and 4. In this context we can regard piecewise linear reconstruction as a refinement step to an initial model. Instead of using piecewise constant functions as the initial model in step 1, we now use piecewise equilibrium functions, constructed around the cell centres. Step 1 is thus replaced by

$$Y^r_{n-\frac{1}{2}} \equiv \hat{Y}_n(x_{n-\frac{1}{2}}), \quad Y^l_{n+\frac{1}{2}} \equiv \hat{Y}_n(x_{n+\frac{1}{2}}) \tag{2.9}$$

$$\hat{Y}_n(x_n) \overset{!}{=} Y_n \tag{2.10}$$

where the function $\hat{Y}_n$ has to satisfy identities (1.44) to (1.48) with $Q = x_n$.

The second part of the modification changes the computation of the source terms in step 4. Using the new formulation of the evolution equations derived in Sec. 1.2, we obtain the source terms by finite differentiation of the hat terms in Eqs. (1.61) and (1.63) as follows:

$$s^{S_1}_n = \frac{1}{\Delta x}\left[\sqrt{g}\hat{P}_n\right]^{x_{n+\frac{1}{2}}}_{x_{n-\frac{1}{2}}} \tag{2.11}$$

$$\begin{aligned} s^{\tau}_n &= \frac{1}{\Delta x}\left[\sqrt{g}\hat{v}^1_n\hat{P}_n\right]^{x_{n+\frac{1}{2}}}_{x_{n-\frac{1}{2}}} + \text{other dimensions} \\ &\quad + r\left(Y_n, \left(D_j g_{ik}\right)_n, \left(D_j\sqrt{d}\right)_n, \left(D_0 g_{ik}\right)_n, \left(D_0\sqrt{d}\right)_n\right) \end{aligned} \tag{2.12}$$

Here, D_j is the 2nd order accurate central finite difference operator. Note that numerical expressions $D_0 g_{ij}$ for the time derivatives of the metric are needed in the computation of the energy equation. I assume they are given, either because the Cowling approximation is used, or because the code is coupled to a metric evolution code which computes the time derivatives.

2.2.3 Benefit of the modification

By design, the modified scheme has the following property: For each isentropic solution that is manifestly stationary in comoving coordinates, there is a corresponding numerical state which is *exactly preserved*, up to truncation errors, *for any resolution.* The corresponding numerical state is the one whose cell averages q_n are set to the values at the cell centres of the physical state.

Proof: As noted in Sec. 1.2.3, the physical state assumed above corresponds to an equilibrium function $\hat{Y}^G$ with $\hat{w}^i \equiv 0$. For the corresponding numerical state defined above, the equilibrium functions constructed by the modified scheme around the different cell centres perfectly agree with $\hat{Y}^G$ and hence with each other. In particular, there are no gaps between them at the cell boundaries, i.e.

$$Y^r_{n\pm\frac{1}{2}} = Y^l_{n\pm\frac{1}{2}} = \hat{Y}^G(x_{n\pm\frac{1}{2}}) \quad \Rightarrow \Delta Y_{n\pm\frac{1}{2}} = 0 \tag{2.13}$$

Therefore step 2 (linear reconstruction) has no effect:

$$\bar{Y}^r_{n-\frac{1}{2}} = Y^r_{n-\frac{1}{2}} - 0 \quad , \quad \bar{Y}^l_{n+\frac{1}{2}} = Y^l_{n+\frac{1}{2}} + 0 \tag{2.14}$$

The Riemann problem hence reduces to the trivial one where both states are the same, and, up to truncation errors, the numerical fluxes are given by

$$\begin{aligned} F_{n\pm\frac{1}{2}} &= R(\hat{Y}_{n\pm\frac{1}{2}}, \hat{Y}_{n\pm\frac{1}{2}}) = f(\hat{Y}_{n\pm\frac{1}{2}}) & (2.15)\\ &= \left(0, \sqrt{g}\hat{v}^1\hat{P}, \sqrt{g}\hat{P}, 0, 0\right)_{n\pm\frac{1}{2}} & (2.16)\end{aligned}$$

where we have used $\hat{w}^i = 0$ and omitted the superscript G. In the computation of $\partial_0 q_n$, the hat terms in Eqs. (2.11), (2.12), and (2.16) do cancel. The same holds for the contributions of the other dimensions which have been omitted again. The remaining term r in Eq. (2.12) also vanishes due to $w^i = 0$ and the assumption of stationarity. Hence we obtain $\partial_0 q_n = 0$ up to truncation errors, finishing the proof.

An illustration of the reconstruction scheme in various situations is given in Fig. 2.1. Note that in flat space and Cartesian coordinates, the equilibrium functions become constant and the method reduces to the original one.

The class of solutions which are preserved exactly is very narrow. However, it is plausible to assume a significant benefit also for systems that are close to this class, in the sense that pressure and gravitational forces are partially balanced and the system is nearly isentropic. It needs to be stressed that in contrast to the possible benefits, the scheme itself is not limited to quasi-stationary scenarios. In the derivation of Eqs. (1.61) and (1.63), no approximations are needed. Regarding the new reconstruction scheme, the only difference is that the states which are modelled most accurately in step 1 are equilibrium states instead of constant states.

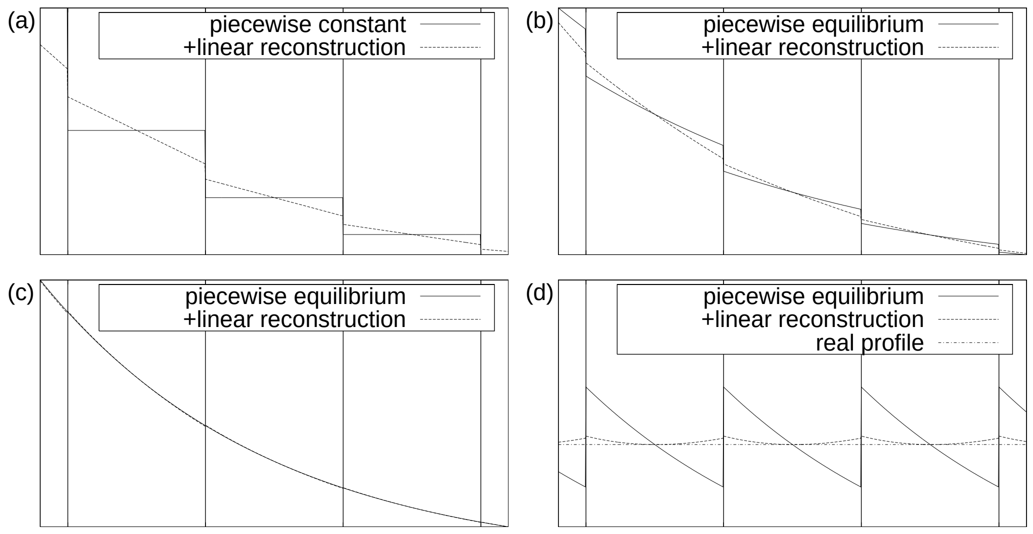


Figure 2.1: Illustration of the modified reconstruction scheme on the example of a polytropic gas at rest in a homogeneous gravitational field. Plotted is the specific energy density. (a) Standard piecewise constant/linear reconstruction. (b) Piecewise equilibrium + linear reconstruction if the true profile is close to equilibrium. (c) Same case if the system is in equilibrium. (d) Same case if the profile is homogeneous, thus far from equilibrium. The fluxes themselves are not captured very well, but the flux differences are still exact (zero) in this particular example. However, the source terms, which are proportional to the height of the sawtooth, are dominant in this case anyway.

Heuristically, one might argue that the new scheme is the natural extension of the HRSC methods developed in flat space to problems involving gravitation, since HRSC methods are based on the solution of Riemann problems, and the key feature of the Riemann problem is not that the left and right states are constant in space, but that they are constant in time. This property is used frequently, for example in the construction of the HLLE approximate Riemann solver. With gravitation, a discontinuity separating two equilibrium states is more similar to the original Riemann problem than two constant states in this respect. However, it is currently unknown wether the new scheme behaves better or even worse than the original one in general scenarios.

2.2.4 Implementation details

So far, a general class of schemes was described. In the following I state the detailed choices used for the actual implementation of the scheme.

As an approximative Riemann solver, the HLLE flux described in [25, 19] is used, given by

$$R = \begin{cases} f\left(q^l\right) & v^- \geq 0 \\ \frac{v^+ f\left(q^l\right) - v^- f(q^r) + v^+ v^- (q^r - q^l)}{v^+ - v^-} & v^- < 0 < v^+ \\ f(q^r) & v^+ \leq 0 \end{cases} \tag{2.17}$$

where v^- and v^+ are the minimum and maximum signal propagation speeds. They are estimated from the extremal eigenvalues of the Jacobian $\partial f / \partial q$ given in [23, 10]. This method is known to be robust, and it is also very fast, since the eigenvectors of the Jacobian are not required, in contrast to other methods. As a downside, the scheme is slightly more diffusive.

The slope limiter employed for most of the simulations is the "monotonized central-difference" method given by

$$L(a,b) = \begin{cases} \min\left(2a, 2b, \frac{1}{2}(a+b)\right) & a,b > 0 \\ \max\left(2a, 2b, \frac{1}{2}(a+b)\right) & a,b < 0 \\ 0 & ab \leq 0 \end{cases} \tag{2.18}$$

Other implemented choices are the "MinMod" limiter, defined by

$$L(a,b) = \begin{cases} \min(a,b) & a,b > 0 \\ \max(a,b) & a,b < 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.19}$$

and the geometric mean

$$L(a,b) = \begin{cases} \frac{ab}{a+b} & ab > 0 \\ 0 & ab \leq 0 \end{cases} \tag{2.20}$$

2.2.5 Stellar surfaces

In the preceding discussion I have not taken into account the possibility of fluid-vacuum boundaries, e.g. stellar surfaces, which are a frequent source of problems in numerical hydrodynamics. The main difficulties are the following: First, the computation of the primitive variables from the conserved ones degenerates for $\rho = 0$. Second, the change of density inside a numerical cell is comparable to the density itself at the surface. If the change was due to a discontinuity at a shockwave, a HRSC scheme would yield good results because it has built-in analytical knowledge about shocks. But since the change is due to gravitational forces the numerical scheme is ignorant about, the results will be quite wrong.

The usual way of treating the surface is to use an artificial atmosphere, which means to enforce a minimum density and set the velocities to prescribed values in low density regions. Unfortunately, this approach would break the exact preservation of stationary solutions that would otherwise be achieved by the modified scheme. As a remedy, the new scheme also allows for a new treatment of surfaces, as described in the following.

First, a cutoff density ρ_v is introduced. If the density drops below ρ_v in a cell after one evolution step, the advection speed w^i is set to zero. The density is only forced to be nonnegative. In addition, a polytropic EOS is used if $\rho < \rho_{\mathrm{cold}}$, where ρ_{cold} is another parameter. This allows one to compute the primitive variables without encountering numerical problems in low density regions.

Second, fluxes between two cells with densities above and below ρ_v are computed by assuming that the state at the boundary is given by the equilibrium function of the cell with higher density. If the denser cell has zero advection speed, the flux and source contributions belonging to this boundary cancel (see Sec. 2.2.3). The cell at the surface is therefore kept in a one-sided equilibrium. Only the interactions with interior cells can cause any acceleration. States which are exactly preserved inside the star are not disturbed by the surface treatment. Fluxes between cells with densities below ρ_v are not computed at all. This also saves computation time compared to artificial atmosphere methods, since vacuum regions are not evolved.

In general scenarios this method leads to results which are plain wrong at the surface, like it is the case with artificial atmospheres. It is difficult to estimate theoretically how this crude treatment of the surface affects the simulation as a whole; this question will be tackled in the section on numerical results.

It might seem as if a vacuum-aware piecewise equilibrium reconstruction procedure would be the straightforward solution to the aforementioned problems. The equilibrium functions can easily and correctly be extended into vacuum regions. However, in more than one dimension the task of finding the equilibrium function corresponding to a given cell averaged density would become quite complex.

For the simulations shown in this work, yet another tweak was applied to the low density regions: The mass flux out of cells with $\rho < \rho_v$ is limited in a way such that the density cannot drop below zero after one timestep. The correction is not important for the evolution; it just removes a small drift of the conserved mass which would be present otherwise. For the interpretation of other errors, it is advantageous if the mass is conserved exactly.

2.3 Other ingredients

Although the numerical scheme is the most important part of any hydrodynamic evolution code, many other ingredients are needed to actually perform a simulation. In the following, some of them are described.

2.3.1 General design

The largest part of an evolution code usually consists of routines providing the infrastructure, like memory allocation or disk input/output. The `Pizza` code is build inside the framework of the `Cactus` computational toolkit [1], which provides some core functionality such as memory management for grid variables, a sophisticated scheduler, handling of parameter files, and, most importantly, parallelisation support. One goal of `Cactus` is to ease modularisation. Different tasks are implemented as modules (called thorns in the `Cactus` terminology), which can interact with each other via standard interfaces.

There already exist a number of thorns, implementing for example the saving of data, the Method of Lines with different time integrators, fixed mesh refinement, or rotating star initial data. Also the `Pizza` code, which implements the hydrodynamic evolution scheme described in Sec. 2.2, is a `Cactus` thorn. Other thorns written for this work are `PizzaTOV`, which computes initial data for spherical neutron stars, `PizzaTOY`, which sets up initial data of the toy model described in Sec. 3.3.1, `PizzaRecycle`, which extracts oscillation mode eigenfunctions (see Sec. 2.3.4), `PizzaStar`, which provides boundary conditions and runtime analysis for single star simulations, and `PizzaShock` which does the same for shock tube tests. All these thorns use a thorn `PizzaBase`, which provides the EOS as well as utility routines like conversion between primitive and conserved hydrodynamic variables.

The code is written in the C++ programming language described in [56]. Special care was taken to achieve a sensible degree of internal modularisation and abstraction. For this, the language features of C++, in particular the object-oriented approach, turned out to be very useful.

An example for the benefits of C++ is the treatment of vector and matrix operations, which are frequently needed in the `Pizza` code. For this task, a small library implementing fixed-size vectors and matrices was written. It also defines a notation of co- or contravariant vectors and metric tensors as well as useful arithmetic operators. Using this library, the implementation of a term $z = b^i v^j g_{ij}$ can be written as

```
  z = b * (v * g);
```

It is not possible to perform illegal operations this way, like multiplying two covariant vectors or to add a covariant to a contravariant vector.

A (trivial) example for abstraction and modularisation is the treatment of the EOS. It is accessed exclusively via a general set of routines, which are not specific to a particular EOS. To change the EOS, one only needs to modify those routines and nothing else.

The usage of C++ language features also proved extremely useful to detect coding errors already during the compilation stage. Although it is sometimes claimed that C++ results in slower code than Fortran, one has to take into account how often a simulation has to be performed until all errors are eliminated. In the author's opinion, a well structured coding style is more important than a minor performance increase.

For simulations with coupled spacetimes evolution, I use `Pizza` in conjunction with the `BSSN_MoL` thorn developed at the Albert Einstein Institute Potsdam, which integrates the BSSN equations in time. To compute the constraint violation in this case, I also use a thorn `ADM_Constraints`. Both thorns were usable with the `Pizza` code without the need for any modifications, which demonstrates the usefulness of the modularisation philosophy behind `Cactus`.

As important as the simulation itself is the postprocessing of the data. Because this is a very tedious task for large scale simulations, I automated the process. After each simulation, a single command is used which produces a set of HTML pages viewable with any internet browser. For star simulations, these pages show the time evolution and corresponding Fourier spectra of various quantities at a given sample point, frequency peaks detected in the spectra, 1D-cuts along the coordinate axes, 2D cuts in the coordinate planes, the evolution of global quantities like conserved mass, a log file of the simulation, and the simulation parameters. This is achieved by combining Linux shell scripts, `gnuplot` scripts to produce 1D plots, scripts for the `OpenDX` visualisation package to produce 2D cuts, a C++ program for Fourier analysis and peak search, and `PHP` programs to produce the web pages. The time needed for the implementation of this framework was more than balanced by the time savings during postprocessing. More importantly, this approach raises the probability that numerical artifacts are detected.

2.3.2 Numerical evolution schedule

In the following, an overview on the numerical evolution schedule is given. The main loop performs the time integration, using the Method of Lines described in Sec. 2.1. For this, it is convenient to define two groups of variables. Evolved variables, `Ev` thereinafter, are the ones directly evolved in time by computing their time derivatives, whereas derived variables (`Dv`) are variables which are recomputed from the evolved ones after each evolution substep.

The evolution thorns `Pizza` and `BSSN_MoL` together provide the time derivative for all evolved variables. The time derivatives are then used by another module, `MoL`, to perform the time integration. The process is outlined by the pseudo code below. The actual implementation inside the `Cactus` framework looks very different, although it does the same.

```
InitialData(Ev,Dv)          \\initialise ALL variables
while (t<Tfinal)
{
  RuntimeAnalysis(t,Ev,Dv)  \\save interesting quantities
  dt = Timestep(Dv,Ev)      \\compute timestep
  MoL_Update(dt,Ev,Dv)      \\Perform evolution step
  t = t + dt
}
```

The `Timestep` function computes the timestep as described in Sec. 2.1. In `MoL_Update(Dv,Ev)` the state is evolved over one timestep `dt` using the time integrator of choice. For the example of the 2nd order Runga-Kutta scheme described in Sec. 2.1, the routine does the following:

```
MoL_Update(dt, Ev, Dv)
{
  MoL_PreStep(Ev,Dv)

  MoL_Substep(dt,Ev,Dv,Ev1,Dv1)
  MoL_Substep(dt,Dv1,Ev1,Ev2,Dv2)
  Ev = (Ev + Ev2) / 2

  MoL_PostStep(Ev,Dv)
}
```

In `MoL_PostStep`, the derived variables are recomputed and boundary conditions are applied. `MoL_PreStep` will be discussed later. The function `MoL_Substep` performs a forward Euler step with stepsize `dt`, that is

```
MoL_Substep(dt, Ev, Dv, Ev1, Dv1)
{
  dEv = MoL_RHS(Ev,Dv)
  Ev1 = Ev + dt*dEv
  MoL_PostStep(Ev1,Dv1)
}
```

In MoL_RHS, numerical estimates for the time derivatives of the evolved variables are computed. MoL_RHS and MoL_PostStep call the following functions, which do the main work:

```
MoL_RHS (Dv,Ev)
{
  Pizza_RHS(Dv,dEv)
  BSSN_RHS(Dv,Ev,dEv)
  return dEv
}

MoL_PostStep(Dv,Ev)
{
  BSSN_Symmetry_BC(Ev)
  BSSN_Compute_ADM(Ev,Dv)

  Pizza_BC(Ev)
  Pizza_Compute_Upper_Metric(Dv)
  Pizza_Recover_Primitives(Ev,Dv)
  Pizza_Compute_T_ab(Dv)
}
```

Here, Pizza_RHS computes the time derivatives of the conserved hydrodynamic variables as described in Sec. 2.2. For this, it only needs the primitive variables ρ, ϵ, w^i and the spacetime variables α, β^i, g_{ij}, and $\sqrt{d}$. When evolving with a two-parametric EOS *and* coupled spacetime evolution, it would also require the time derivative of the 3-metric to compute the corresponding terms in Eq. (1.64). However, the terms containing the time derivatives are not implemented yet, restricting the coupled spacetime simulations to one-parametric, e.g. polytropic, EOSs.

In BSSN_RHS, the time derivatives of the BSSN variables are computed. For this, the stress energy tensor T_{ab} is required in addition to the spacetime variables. This routine also applies outer boundary conditions to the time derivatives of the BSSN variables.

In `Pizza_BC`, boundary conditions are applied to the conserved hydrodynamic variables. The BC are problem-specific and will be described in the corresponding sections. In `BSSN_Symmetry_BC`, symmetry boundary conditions, like equatorial or octant symmetry, are applied to the BSSN variables.

The function `BSSN_Compute_ADM` computes the standard ADM variables g_{ij}, K_{ij}. In `Pizza_Compute_Upper_Metric` $\sqrt{d}$ and g^{ij} are computed. The latter is only needed in `Pizza_Recover_Primitives`, where the primitive variables are computed from the evolved ones as described in Sec. 2.3.3. Additionally, the routine applies corrections in low density regions, e.g. stellar surfaces, as described in Sec. 2.2.5. If corrections are applied, the conserved quantities are recomputed from the primitives, ensuring that conserved and primitive variables are in a consistent state everywhere after `Pizza_Recover_Primitives`.

The stress energy tensor T_{ab} is computed in `Pizza_Compute_T_ab`. Note that `Pizza` and `BSSN_MoL` communicate only via the stress energy tensor provided by `Pizza` and the metric variables provided by `BSSN_MoL`. This approach makes it easy to replace either module by a different one, if available.

For clarity, lapse and shift were not explicitly mentioned in the pseudocode so far. They are neither evolved nor derived variables, but gauge variables which can be arbitrarily chosen. During the `MoL`-substeps, they stay fixed. Lapse and shift are set by `BSSN_MoL` according to the chosen gauge condition. This happens in `MoL_PreStep`, i.e.

```
MoL_PreStep(Dv,Ev)
{
  Lapse = BSSN_Compute_Lapse(Ev,Dv)
  Shift = BSSN_Compute_Shift(Ev,Dv)
}
```

For simulations in Cowling approximation, the `BSSN` routines are deactivated and the 3-metric, lapse, and shift are constant in time.

2.3.3 Reconstruction of primitive variables

One of the most time consuming parts of the `Pizza` code is the computation of the primitive variables. Since the variables ρ, ϵ, v^i are needed during the evolution, they have to be computed from the numerically evolved variables D, τ, S. Unfortunately, there is no analytic expression of the primitive variables in terms of the evolved ones. They have to be obtained by numerically solving the nonlinear equations (1.25), (1.27), and (1.28). This is usually done by casting the equations into a root finding problem in terms of one of the primitive variables. In [35], the pressure is used as the independent variable. I use the specific enthalpy h instead, which has some minor technical advantages. For example the initial bracketing of the root takes a simple form. In detail, the primitive variable reconstruction used in the `Pizza` code works as follows. First we define some useful quantities by

$$
\begin{aligned}
z &\equiv \gamma v\,, \quad v \equiv \sqrt{v^i v_i} && (2.21)\\
r^i &\equiv \frac{S^i}{D}\,, \quad r \equiv \sqrt{r^i r_i} && (2.22)\\
s &\equiv \frac{\tau}{D}\,, \quad d \equiv \frac{D}{\sqrt{d}} = \gamma\rho && (2.23)
\end{aligned}
$$

It is trivial to show the identities

$$
r = zh, \quad \gamma^2 = 1 + z^2, \quad \gamma - 1 = \frac{z^2}{\gamma + 1}
$$

We have

$$
\begin{aligned}
s &= \gamma h - 1 - \frac{P}{\gamma\rho} && (2.24)\\
&= \gamma h - 1 - \frac{1}{\gamma}\left(h - 1 - \epsilon\right) && (2.25)\\
&= \frac{1}{\gamma}\left(\left(\gamma^2 - 1\right) h - \gamma + 1 + \epsilon\right) && (2.26)\\
&= \frac{1}{\gamma}\left(z^2 h - \frac{z^2}{\gamma + 1} + \epsilon\right) && (2.27)
\end{aligned}
$$

For given evolved variables, we can now write the primitive variables as functions of h alone, i.e.

$$
\begin{aligned}
\gamma(\tilde{h}) &= \sqrt{1 + \frac{r^2}{\tilde{h}^2}} && (2.28)\\
\epsilon(\tilde{h}) &= \gamma(\tilde{h}) s - \frac{r^2}{\tilde{h}^2}\left(\tilde{h} - \frac{1}{\gamma(\tilde{h}) + 1}\right) && (2.29)\\
\rho(\tilde{h}) &= \frac{d}{\gamma(\tilde{h})} && (2.30)\\
v^i(\tilde{h}) &= \frac{r^i}{\tilde{h}\gamma(\tilde{h})} && (2.31)
\end{aligned}
$$

where the tilde is used to distinguish between the function argument $\tilde{h}$ and the actual enthalpy h. Inserting $\rho(h)$ and $\epsilon(h)$ into the equation of state results in

$$
0 = f(h), \quad \text{where} \quad f(\tilde{h}) \equiv \tilde{h} - h(\rho(\tilde{h}), \epsilon(\tilde{h})) \qquad (2.32)
$$

This is a root finding problem in $\tilde{h}$ which can be solved numerically. However, there is not always a physical solution. The physical regions in the space of evolved variables are bounded by the requirements $\rho > 0$ and $\epsilon > 0$. The first is trivially satisfied if $D > 0$, which is silently assumed in the following. For the latter, it is instructive to look at the case of dust, that is $P = \epsilon = 0, h = 1$. From Eq. (2.24), we then obtain

$$s = \gamma - 1 = \sqrt{1 + r^2} - 1 \qquad \Leftrightarrow \qquad r^2 = s(s + 2) \tag{2.33}$$

A lower value of s would mean $\epsilon < 0$, hence we have to demand

$$r^2 \leq s(s + 2) \tag{2.34}$$

The other way around, it is easy to show that $h > 1$ if Eq. (2.34) holds. Another special case is given for zero velocities, $v^i = 0$. We than have $\gamma = 1, r = z = 0$ and hence

$$\rho = d, \quad \epsilon = s, \quad h = h(d, s) \tag{2.35}$$

Using only $\gamma \geq 1$, it is easy to show that $\epsilon \leq s$ and $\rho \leq d$. Assuming $\partial_\epsilon h > 0, \quad \partial_\rho h > 0$, it follows that $h < h_m \equiv h(d, s)$. Therefore, the solution of Eq. (2.32) is located in the interval $[1, h_m]$ if condition (2.34) is satisfied.

For a one-parametric equation of state, i.e. $h = h(\rho)$, the reconstruction procedure is almost the same. We only need to replace Eq. (2.32) by

$$f(\tilde{h}) \equiv \tilde{h} - h(\rho(\tilde{h})) \tag{2.36}$$

This means that τ and hence the energy equation is not used anymore. Also, condition (2.34) is not needed in this case, since ϵ is now determined by ρ via the EOS, and $\epsilon \geq 0$ is always satisfied.

Using these prerequisites, the root is computed using the Regula Falsi iterative method. The iteration is stopped if $|f(\tilde{h})| \leq a\tilde{h}$, where a is the desired accuracy in $\tilde{h}$. For the simulations shown here, a value of $a = 10^{-10}$ is used.

I choose the Regula Falsi method instead of a derivative root solving algorithm like Newton-Raphson because when using a tabulated EOS (which is not implemented yet), the computational benefits are questionable while the code becomes more complex. In practice, the Regula Falsi method converges very fast, for the stellar simulations typically after 3 iterations.

From Eqs. (2.30) and (2.31), it is obvious that the accuracy of ρ and v^i is comparable to the accuracy of $\tilde{h}$. For low velocities the accuracy of ρ will be even better. In the case of a one-parametric EOS the accuracy of ϵ is comparable to a, since ϵ is then determined from ρ using the EOS.

For a two-parametric EOS on the other hand, the accuracy of ϵ can be very bad, namely if the first and the second term in Eq. (2.29) nearly cancel. This is the case if $z^2 \gg \epsilon$, which means that the kinetic energy dominates the internal one. To require a better accuracy in $\tilde{h}$ would not cure the problem, because the evolved variables have errors too, which are amplified by the cancellation of terms. For star oscillations, this problem luckily manifests itself only near the star surface, where ϵ goes to zero. Since ϵ cannot be determined accurately in low density, medium velocity regions anyway, a parameter ρ_{cold} was introduced, and a polytropic EOS is used if $d < \rho_{\mathrm{cold}}$. This prevents low density regions to produce excessive errors in the temperature which would in turn affect denser neighbouring regions.

2.3.4 Mode recycling

To extract eigenfunctions of stellar oscillation modes, I use a technique called mode recycling, which is also used in [17]. In a first run, several modes are excited at once by applying a trial perturbation. A first guess of the eigenfunction for some variable z is obtained by numerically computing the integral

$$\frac{1}{T}\int_0^T (z(t) - z_0) e^{i\omega_m t} dt \tag{2.37}$$

where ω_m is the mode frequency extracted from an earlier run, z_0 is the ground state. This way the Fourier transform for a single frequency is obtained at each point. The integration is performed during the evolution using the midpoint approximation, that is, we simply add up the integrand for every timestep. Since one oscillation period is typically resolved by more than thousand timesteps, there is no need to use higher order integration methods.

The eigenfunction is used in a new simulation to perturb the initial data, which yields a better guess due to lesser interferences of other modes. For infinite evolution times, the presence of other modes should have no influence on the results of the Fourier transform. In practice, the evolution times are finite and other modes are not filtered out completely during the Fourier transformation. This is particularly disturbing for the computation of node locations, because near the node, the mode of interest is easily dominated by such errors.

For the excitation of oscillations using an extracted eigenfunction, it is sufficient to perturb the velocity, since the oscillations of density and velocity are typically phase shifted by $\pi/2$. For typical simulations, the eigenfunctions converge after two of the iterations described above.

Chapter 3

Numerical tests

In this chapter, I present numerical tests which demonstrate that the `Pizza` code is working as intended and well suited for single star simulations. In particular, convergence tests for three different test cases are performed, which also yield error estimates for the computation of stellar oscillation frequencies shown in Sec. 4.

3.1 Shocktube tests

A standard test for HRSC methods is the so called shock tube, which is nothing but a Riemann problem, i.e. the initial data consists of two constant states separated by a discontinuity plane. The spacetime is set to flat Minkowski space, and standard coordinates are used, i.e. $g_{ij} = \delta_{ij}$, $\alpha = 1$, $\beta^i = 0$. The exact solution of this problem is analytically known, see [34, 41].

Since the spacetime is flat, this test only validates the correct implementation of special relativistic terms, everything related to gravitation and gauge conditions remains unchecked. For example, the remaining source terms (1.64) in the energy equation vanish. Also the offdiagonal terms in g_{ij} are zero in Minkowski space.

In flat space, the modified scheme used by the `Pizza` code is identical to the standard scheme it is based on. Therefore, I only studied one particular shock problem to validate the correct implementation of the scheme; it is not the aim of this section to investigate the behaviour of the standard scheme in general.

3.1.1 Problem setup

The specific shock tube problem used for the numerical tests is given by:

$$\rho_L = 10 \qquad \rho_R = 1 \tag{3.1}$$
$$P_L = 13.3 \qquad P_R = 10^{-10} \tag{3.2}$$
$$v_L = 0 \qquad v_R = 0 \tag{3.3}$$

where the subscripts L and R refer to the left and right states, and geometric units are used. The EOS for the test is the ideal gas EOS given by $P = (\Gamma - 1)\rho\epsilon$, with $\Gamma = 5/3$. The solution of this problem contains a shockfront, a contact discontinuity, and a rarefaction wave. The velocities are mildly relativistic ($v < 0.8c$). The exact solution, needed for comparison, is computed using a code provided by [28].

To perform tests in 3D and high resolution without using excessive computation time, shear periodic boundary conditions are used together with a numerical grid which is small in all but one direction. In detail, if the grid size is given by the index range $[0..N-1] \times [0..n-1] \times [0..n-1]$ (not including ghost zones), the boundary conditions are:

$$q_{j,(n-1+l),i} = q_{(j-k_x),(l-1),i}, \qquad q_{j,-l,i} = q_{(j+k_x),(n-l),i} \tag{3.4}$$
$$q_{j,i,(n-1+l)} = q_{(j-k_y),i,(l-1)}, \qquad q_{j,i,-l} = q_{(j+k_y),i,(n-l)} \tag{3.5}$$

where $l \in 1..g$ and g is the number of ghost zones. In x-direction, flat boundary conditions are assumed, i.e.

$$q_{N-1+l,i,j} = q_{N-1,i,j} \tag{3.6}$$
$$q_{-l,i,j} = q_{0,i,j} \tag{3.7}$$

The grid spacing is uniform and equal for all dimensions.

The physical problem setup should have the same shear periodic symmetries. This determines the angle of the shockfront, which is therefore parametrised by

$$z = x\frac{k_x}{n} + y\frac{k_y}{n} \tag{3.8}$$

3.1.2 Convergence tests

To test the basic functionality of the code, I evolve the shock tube problem described in Sec. 3.1.1 in 1D for various resolutions. For this, the 3D code is used, which is done by simply not computing derivatives in the unused directions. As a slope limiter, the MC-method described in Sec. 2.2 is used. As

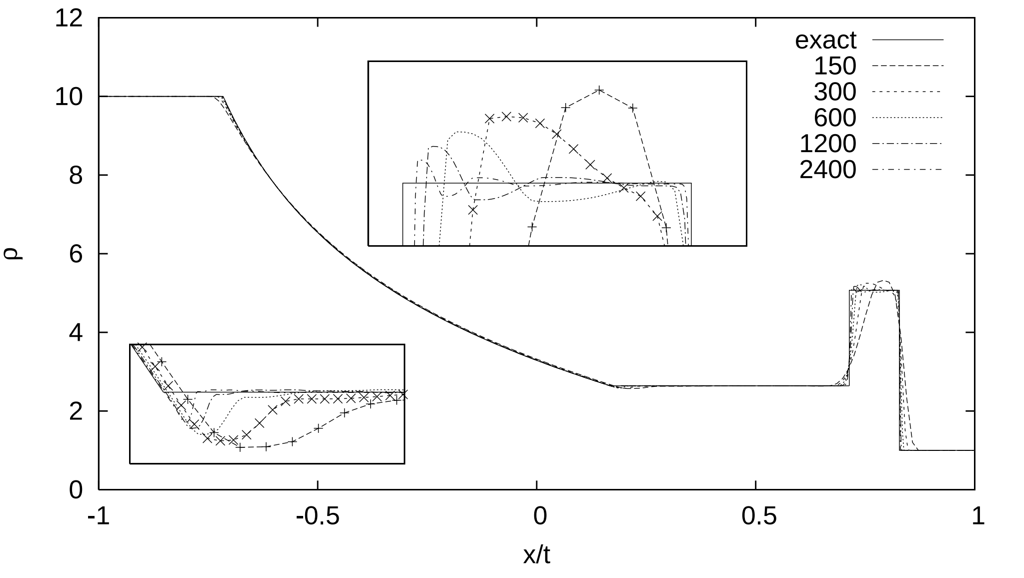


Figure 3.1: Convergence for the shock tube problem in 1D, when using the MC slope limiter. Shown is the rest mass density for various resolutions, after evolution time $t = 0.36$.

time integrator, a 3rd order Runge-Kutta method is applied, with a Courant-factor of 0.35. The results are shown in Figures 3.1 to 3.3. One observes a tendency to smear out the contact discontinuity. This is a known feature of the HLLE approximative flux, caused by the underlying assumptions (see [57]).

However, the main result of this test is that the numerical solution is converging to the exact one. To investigate the convergence quantitatively, the following measure on the final state is used:

$$\delta\rho = \frac{\sum_i |\rho_i - \rho_i^e|}{\sum_i \rho_i^e} \tag{3.9}$$

where ρ_i^e is the exact solution (which is given in tabulated form) interpolated to x_i. In the same way, errors for P and v are defined. Fig. 3.4 shows the errors versus the resolution. The data does roughly follow a convergence law, with a convergence order in the range 0.9..1. This is nearly the 1st order convergence behaviour expected for problems with discontinuities.

To validate the correct behaviour of the code in more than one dimension, the test above is repeated in 3D with a shock that runs along the space diagonal. In detail, $n = k_x = k_y = 2$. Figures 3.5 to 3.7 show the results, which are very similar to the 1D case. Also the quantitative convergence

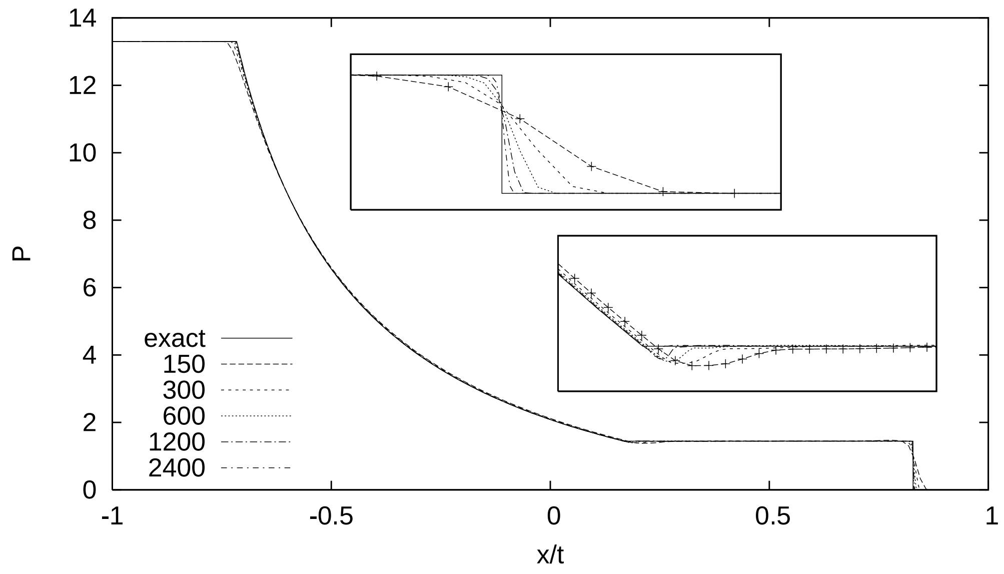


Figure 3.2: Like Fig. 3.1, but displaying the pressure.

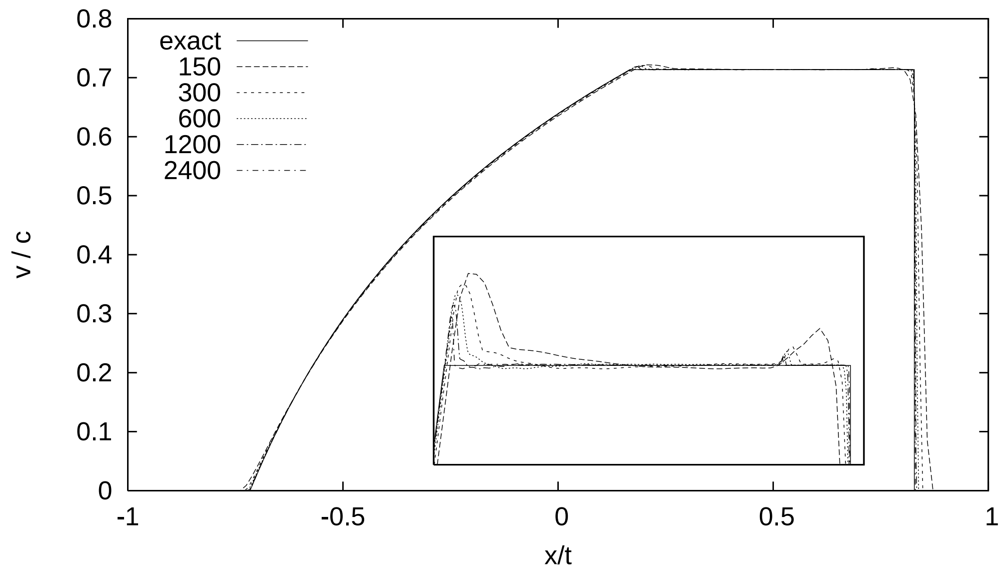


Figure 3.3: Like Fig. 3.1, but showing the velocity.

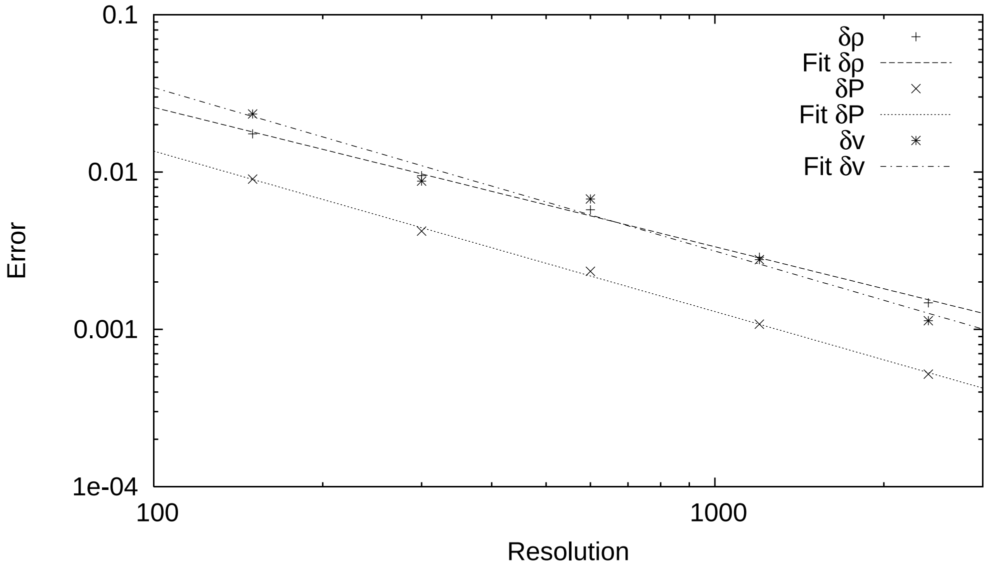


Figure 3.4: Numerical errors as defined in Eq. (3.9), for the 1D shock tube with the MC slope limiter. The fitted convergence orders are 0.89 (error of ρ), 1.0 (error of P), and 1.0 (error of v).

behaviour shown in Fig. 3.8 is almost the same as for the 1D case: Again the convergence order is in the range 0.9..1. A direct comparison between 1D and 2D is given in Fig. 3.9.

So far, only the MC slope limiter was used. The results for the MinMod and geometric mean slope limiters are comparable, as shown in Fig. 3.10. Regarding the resolution of contact discontinuity and shock front, the MinMod limiter is slightly inferior to the other two for the given setup.

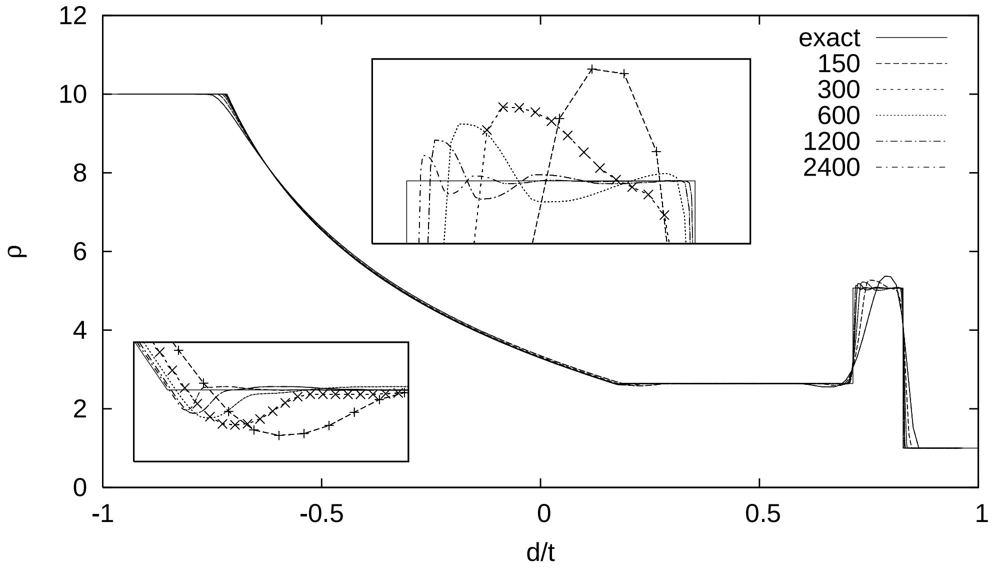


Figure 3.5: Convergence for the shock tube problem in 3D, when using the MC slope limiter. Shown is a cut along the x-axis of the rest mass density for various resolutions, after an evolution time $t = 0.36$. d is the distance to the initial discontinuity.

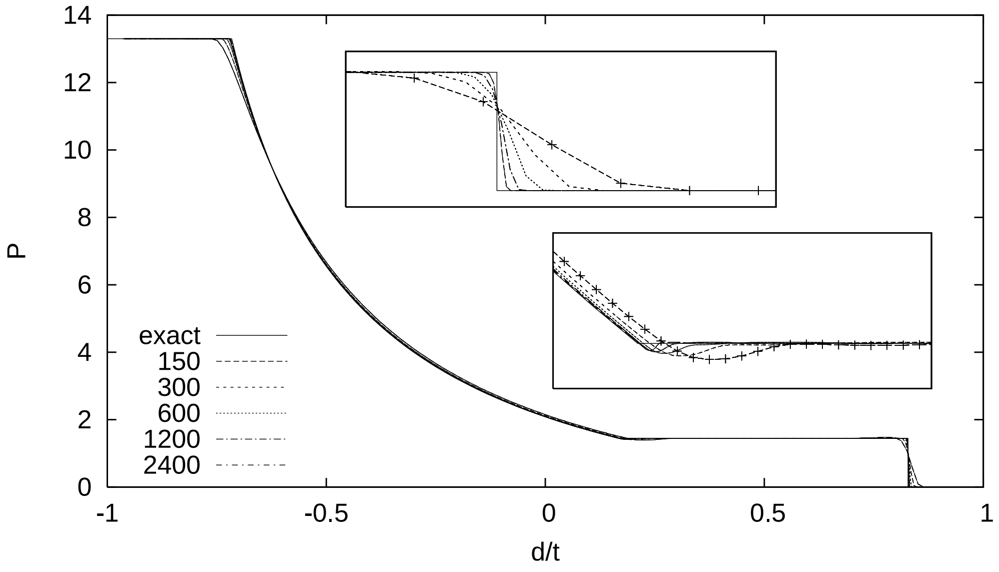


Figure 3.6: Like Fig. 3.5, but showing the pressure.

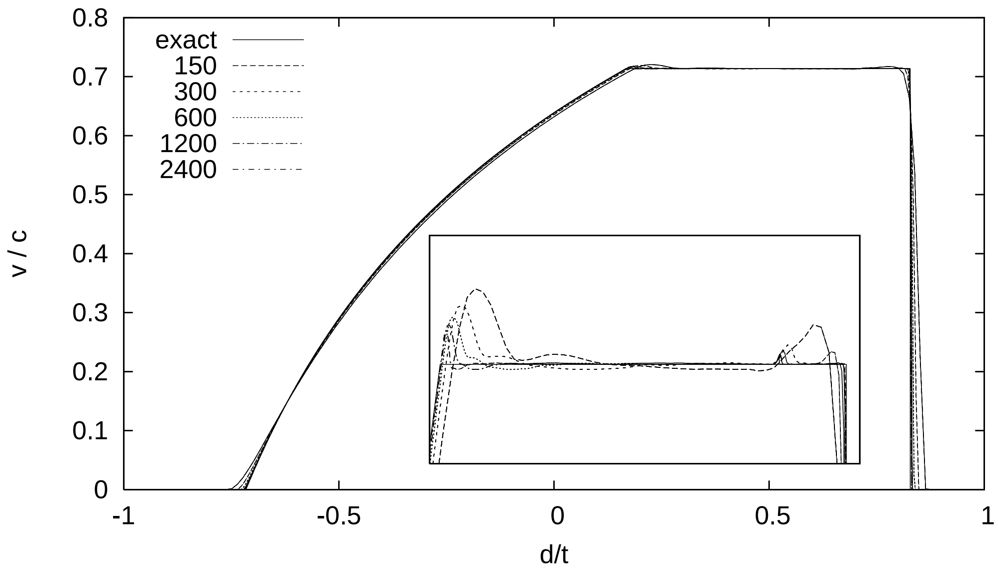


Figure 3.7: Like Fig. 3.5, but displaying the velocity.

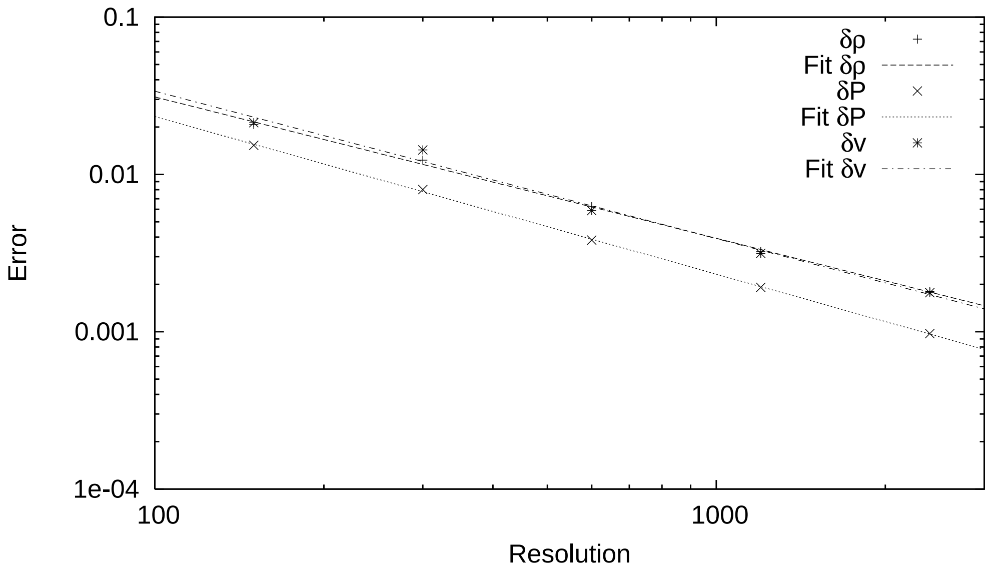


Figure 3.8: Numerical errors as defined in Eq. (3.9), for the 3D shock tube with the MC slope limiter. The fitted convergence orders are 0.90 (error of ρ), 1.0 (error of P), and 0.94 (error of v).

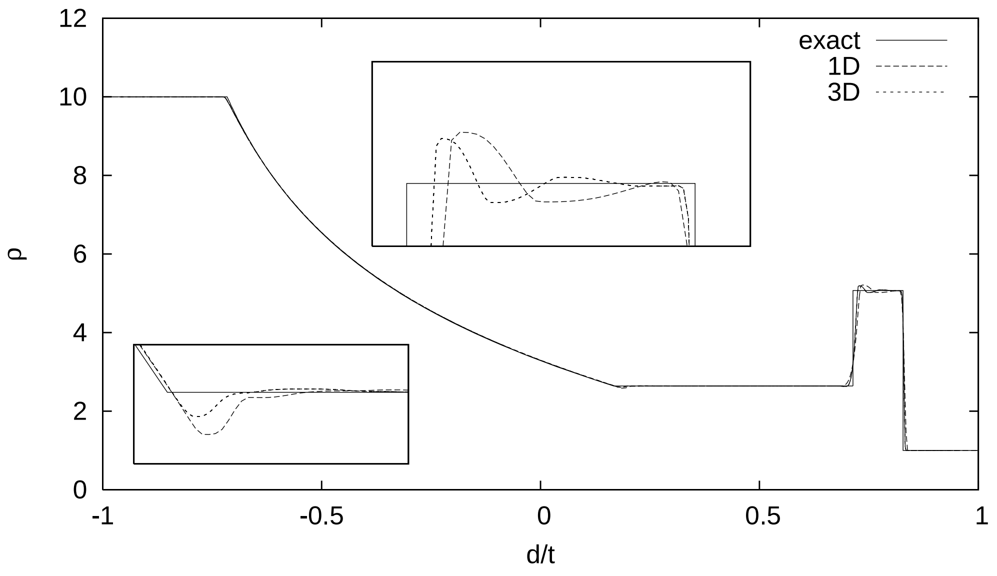


Figure 3.9: Comparison between 1D and 3D shock tube results. For the 3D case, a cut along the x-axis is shown. d is the distance to the initial discontinuity. The resolution is 600 points in 1D and the grid spacing in 3D is the same. Note: since the 3D-shock runs along the space diagonal, it is resolved by a factor of $\sqrt{3}$ more points than the 1D-shock in the plot.

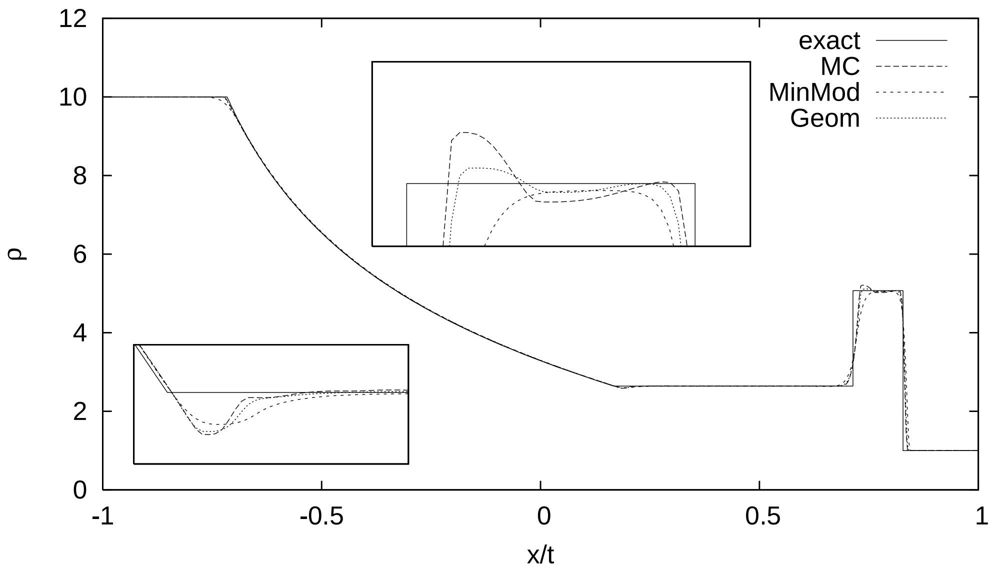


Figure 3.10: Comparison of the MC, MinMod and geometric mean slope limiters for the 1D shock tube. The resolution is 600 points.

3.2 Polytropic neutron star tests

The main test problems for the `Pizza`-code are single neutron stars, in particular a nonrotating and a rigidly rotating stellar model, whose oscillations are also investigated in [21, 54]. For both of them, the initial data obeys a polytropic EOS given by

$$\frac{P}{\rho_p} = \left(\frac{\rho}{\rho_p}\right)^{\Gamma} \tag{3.10}$$

where Γ and ρ_p are constants. This EOS is also used during the time evolution. Since it is a one-parametric EOS, the energy equation is redundant and therefore not used in the time integration. The case of a two-parametric EOS will be discussed Sec. 3.4.

For the tests shown in this section the Cowling approximation is used, i.e. the metric is kept fixed. This is not a physical approximation, but useful to study the hydrodynamic evolution code alone, without interferences from the metric evolution code.

3.2.1 Spherical star model

The first stellar testbed is a nonrotating static neutron star, called model N in the following, whose parameters are summarised in Table 3.1. The structure of such stars was already discussed in Sec. 1.4. Oscillation frequencies for model N can be found in [21, 54].

To construct initial data, the TOV equations (1.100) to (1.102) are integrated numerically using a 4th order Runge-Kutta-Fehlberg scheme (see [45]) with adaptive step size control. Since Eq. (1.102) degenerates at the surface, the integration is stopped shortly below, and the location of the surface is computed using approximation (1.107). After solving the TOV equations, the initial data is transformed to the coordinate system of choice as shown in Sec. 1.4.

Fig. 3.11 shows the density profile of the star. Another important quantity is the sound speed, which is shown in Fig. 3.12.

Due to its spherical symmetry, model N can be considered as a 1-, 2- (axisymmetric) or 3-dimensional problem. Numerically this is realised with one and the same evolution code by just not computing derivatives along one (φ) or two (ϑ, φ) directions. Vectors and tensors remain 3-dimensional.

As coordinate systems, I choose spherical coordinates for the 1D case, cylindrical coordinates for the 2D axisymmetric case, and Cartesian coordinates for the full 3D problem. I use coordinates which are *not* adapted to the shape of the star since the code is designed for general setups. As a

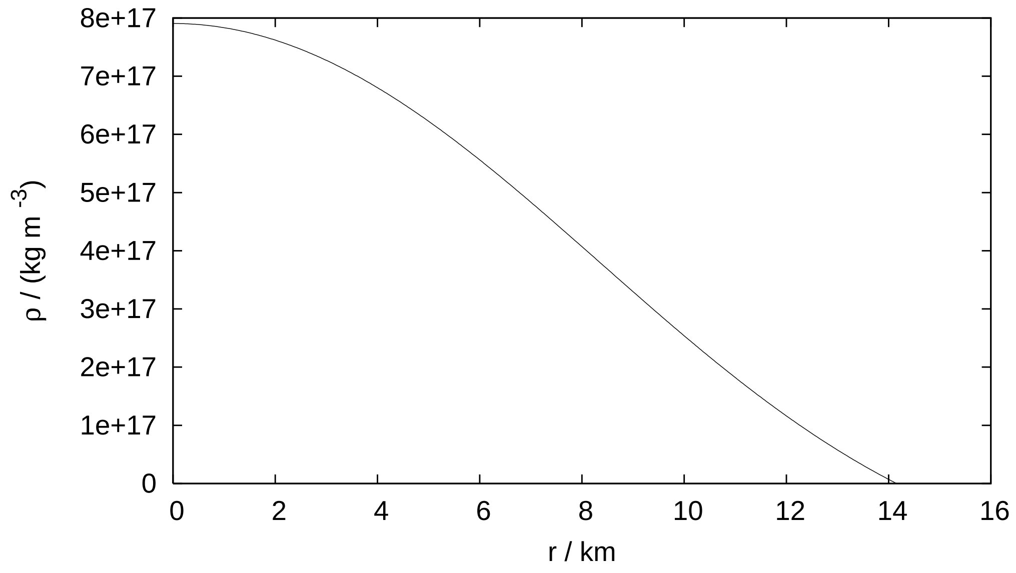


Figure 3.11: Density profile of star model N. r is the circumferential radius.

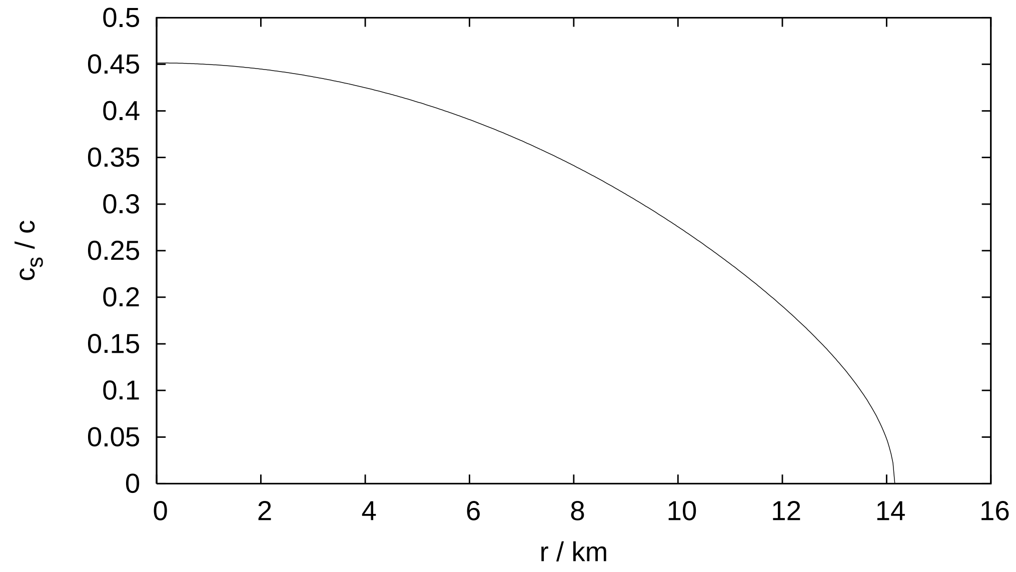


Figure 3.12: Profile of the sound speed for star model N. r is the circumferential radius.

	Model N	Model R
Name in [54]	BU0	BU3
Rotational frequency	0	590 Hz
Rotational period	∞	1.69 ms
Polar/equatorial radius	1	0.85
Equatorial circumferential radius	14.16 km	15.38 km
Gravitational mass	1.400 $M_\odot$	1.503 $M_\odot$
Central restmass density ρ_c	$7.9056 \cdot 10^{17}$ kg/m^3	
Central soundspeed	0.45 c	
Γ	2	
ρ_p	$6.1760 \cdot 10^{18}$ kg/m^3	

Table 3.1: Summary of stellar models N and R. The EOS is a polytrope given by Eq. (3.10). $M_\odot$ is the solar mass.

bonus, cylindrical coordinates allow for bigger timesteps and shorter computation times in 2D axisymmetric simulations than spherical ones. To save further computation time, octant symmetry is applied in 3D and equatorial symmetry in 2D.

In the 2D axisymmetric case, reflective boundary conditions are applied at the symmetry axis. Severe numerical problems could arise from the fact that the volume element $\sqrt{d}$ gets zero at the axis. Therefore, cells located on the axis have to be avoided. Otherwise, divisions by zero would occur during the computation of the primitive variables from the conserved ones. For the 2D simulations, I use a grid for which the axis is aligned with the cell boundaries, not the cell centres. For the space discretisation scheme, the axis is not a problem since it uses primitive variables, which do not degenerate at the axis. Only when computing the final fluxes, the volume element $\sqrt{d}$ is used, but in a multiplication, not a division. Therefore, $\sqrt{d} = 0$ on the axis only results in vanishing fluxes across the axis, which is the correct result. However, since $\sqrt{d}$ on the cell boundaries is obtained by interpolation, the value used in the flux computation is not zero. But due to the reflective boundary condition, the fluxes across the axis will be zero anyhow.

3.2.2 Rotating star model

The second testbed is a rigidly rotating neutron star, called model R, as summarised in Table 3.1. Rigid rotation means that the physical distance between any two fluid elements does not change in time; there is no shear or meridional motion. The rotation frequency (defined with respect to an observer at infinity) is far from the mass shedding limit, but already in the upper range of observed pulsar frequencies. The fastest rotating known pulsar, discovered by [26], spins at 716 Hz. As shown in Fig. 3.13, the profile of the star is significantly oblate. The surface velocity at the equator is around 0.24 c, corresponding to a Lorentz factor of 1.03.

For the evolution, I use a shift vector which is constant in time and corresponds to coordinates corotating with the initial data. Note this is only possible for rigid rotation, if the metric is supposed to stay fixed. I choose corotating coordinates because in that case there is no advection, i.e. $w^i = 0$. Since the accuracy of the advection terms is not improved by the new scheme in any way, there would otherwise be no exact preservation of the equilibrium state, at least when using Cartesian coordinates. In cylindrical or spherical coordinates, the evolution scheme would still be exact in nonrotating coordinates. However, the method of treating the surface yields exact results only for corotating coordinates.

To compute the initial data, a customised version of the `RNS` code described in [55] is used. The main modification, besides technical adaptions to the `Pizza` code, was the possibility to transform to cylindrical coordinates, thus allowing for 2D axisymmetric simulations in the same manner as for model N. I also added the option to enforce Eq. (1.47) by recomputing the density from the lapse function and the velocity.

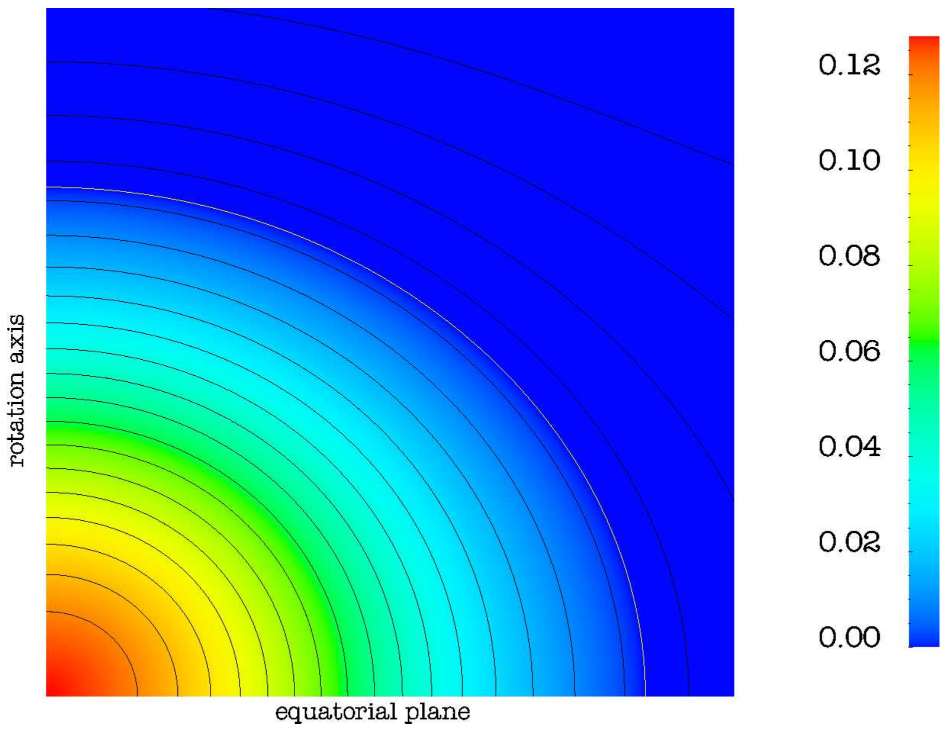


Figure 3.13: Profile of star model R. Colour coded is the specific energy density. The yellow line marks the surface. The black lines are isolines of the effective potential $\ln(\alpha/\gamma)$, which are also lines of constant specific energy inside the star. Shown is a quadratic region of 15×15 km in coordinate space.

3.2.3 Equilibrium preservation

As shown in Sec. 2.2.3, the scheme should have no discretisation errors if applied to the unperturbed star models N or R, independent of the resolution. To test this, I evolve model N with the extremely low resolution (defined as $R/\Delta x$, where R is the stellar radius and Δx the grid spacing) of 10 cells over a time of 100 ms.

It turns out that the state is indeed preserved to an accuracy near machine precision. The maximum change in rest mass density is generally below $10^{-14}\rho_c$, and the maximum velocity below $10^{-15}c$. This is the case in one, two and three dimensions (and also for the ideal gas case, see Sec. 3.4). Fig. 3.14 shows the time evolution of the radial velocity and density at some point inside the star.

When increasing resolution, the tiny amplitude of the noise remains nearly unchanged, which is a strong hint that it is caused in fact by truncation errors and not discretisation errors.

For the rotating model R, small oscillations with $\delta\rho/\rho \approx 10^{-5}$ occur. This is because the initial data obtained with the `RNS` code satisfies Eq. (1.47) only to an accuracy around 10^{-5}. After enforcing Eq. (1.47) in the initial data, the rotating model is preserved as accurately as the nonrotating one by the evolution scheme.

The errors present in simulations with the unmodified standard HRSC scheme behave qualitatively different, as shown in Fig. 3.15. After some time, the fluid flow develops unphysical, vortex-like structures, as shown in Fig. 3.16, with typical velocities in the range $0.001..0.01c$. Additionally, the star immediately starts to oscillate with a small amplitude. The results for the unmodified scheme were obtained at an early stage of the code development and might contain bugs. However, the perfect preservation of equilibrium is clearly impossible with standard schemes.

Taken by itself, the perfect preservation of equilibrium is not very useful. However, assuming a smooth transition for the magnitude of numerical errors when departing from equilibrium, there will exist a region where the benefits are still substantial. As will be shown in the next sections, this region does include star oscillations.

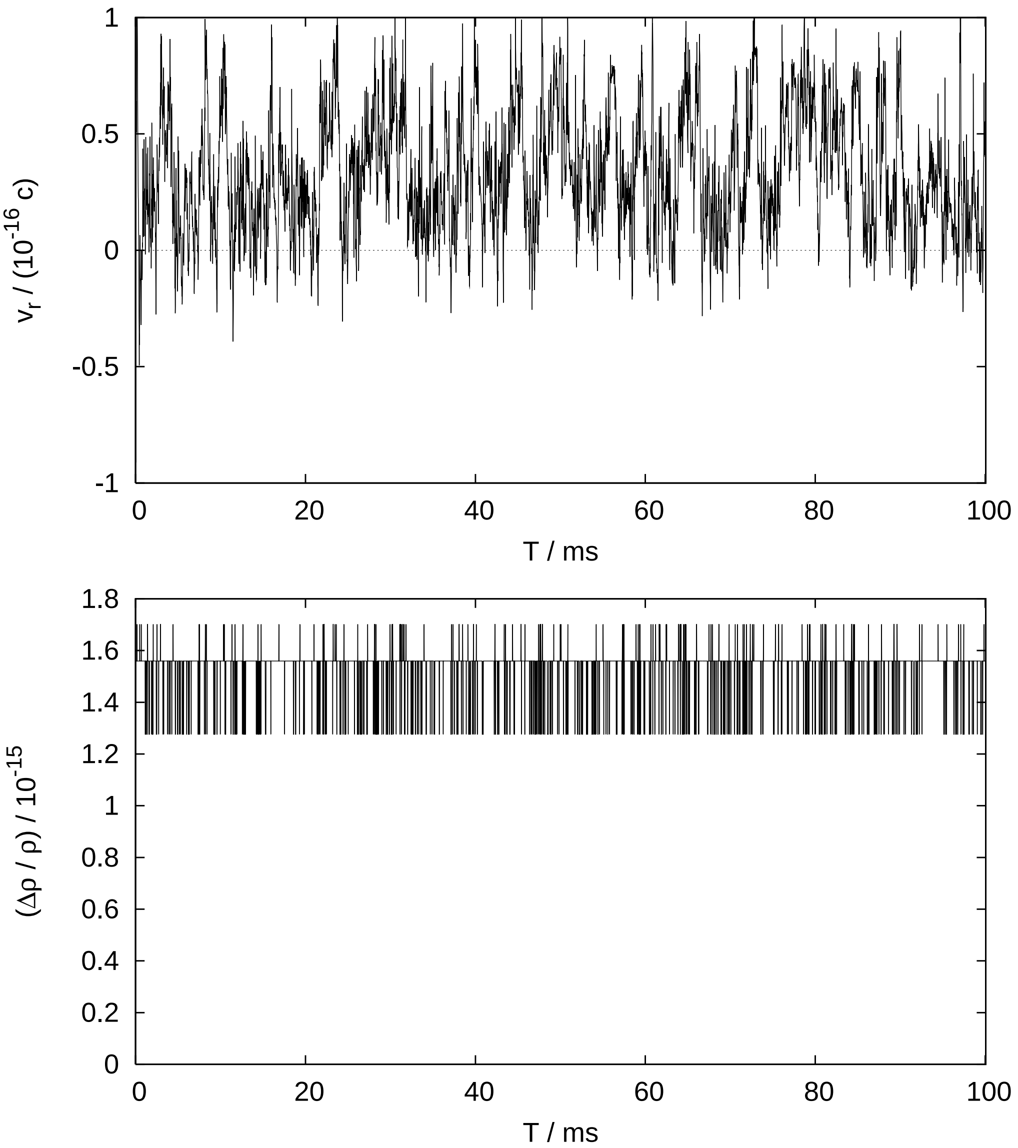


Figure 3.14: Evolution of the radial velocity (Top) and density (Bottom) at $r = 7$ km, $\vartheta = \pi/4$, for a polytropic, axisymmetric 2D-simulation of model N with a resolution of 10^2 cells. The relative density change is comparable to the machine precision $\epsilon_n \approx 2.22 \cdot 10^{-16}$, which is the smallest number where $1 + \epsilon > 1$ holds numerically. The corresponding plots for 1D and 3D simulations are qualitatively the same.

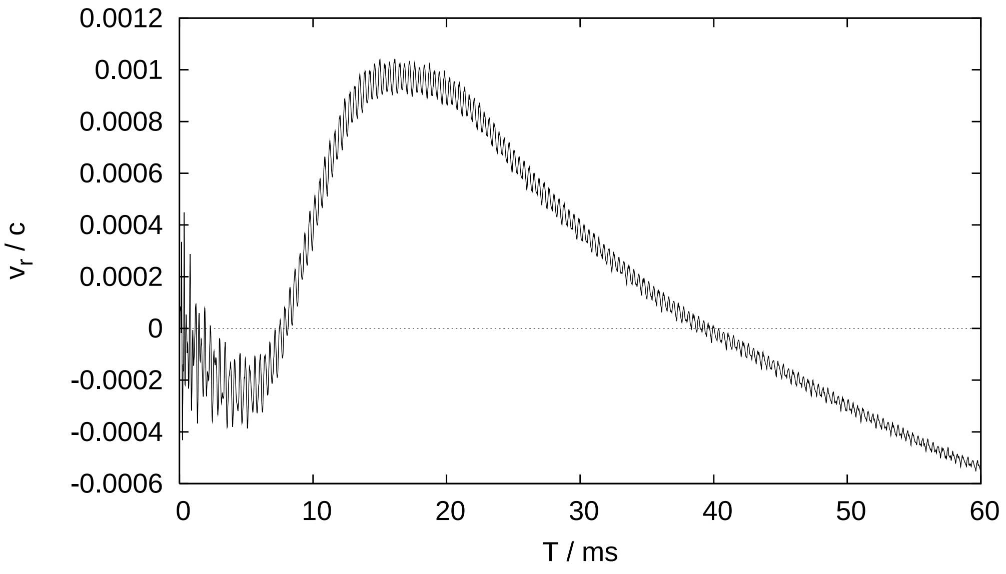


Figure 3.15: Results obtained with the unmodified HRSC scheme described in Sec. 2.2.1. Shown is the evolution of the radial velocity for a polytropic simulation of Model N with a resolution of 50^2 points.

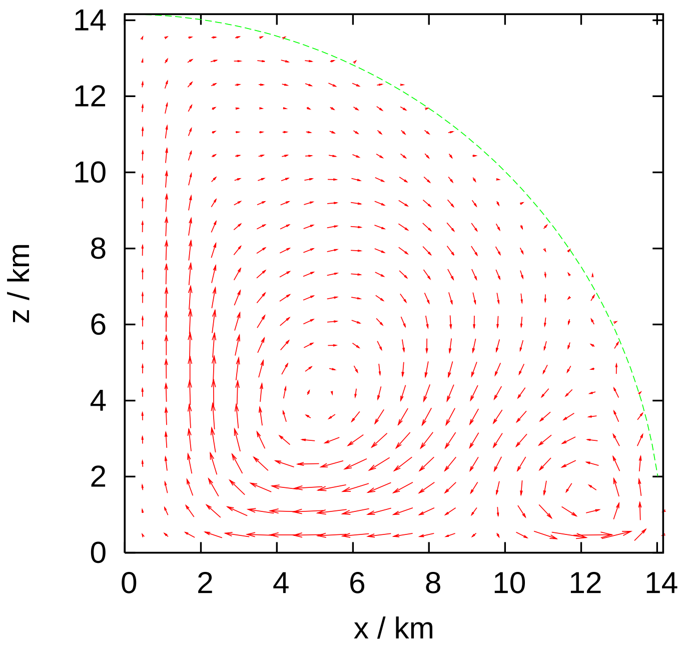


Figure 3.16: Conserved momentum density S_i after 60 ms, for the same simulation as in Fig. 3.15.

3.2.4 Oscillations

To excite oscillations, I apply the following perturbation to the initial data:

$$\Delta\rho = \rho A \, \frac{r}{R} \, Y_l^0(\vartheta) \tag{3.11}$$

where A is a constant, r is the coordinate radius and R the stellar radius. In addition, the density is globally rescaled such that the total conserved mass remains unchanged. Finally, the other fluid variables are recomputed from the density, using the EOS.

For nonrotating stars, the eigenfunctions have the same angular dependency as the perturbations, i.e. spherical harmonics, whereas the radial part of the perturbation does not correspond to any particular eigenfunction. Therefore, several modes are excited at once. Since I use boundary conditions which enforce equatorial symmetry, only modes with even l can be excited.

Throughout this work, the radial modes will be denoted (in order of increasing frequency) by F, H_1, H_2, etc. The nonradial axisymmetric modes will be called ${}^lf, {}^lp_1, {}^lp_2$, etc., where l indicates the angular dependency.

To extract frequencies, I use the time evolution of the radial velocity v_r for radial modes and the velocity in ϑ-direction, v_ϑ, for l=2 modes, on a suitable sample point inside the star. Fig. 3.17 shows a typical result for a perturbed star. Several modes are excited, and the higher modes are damped away more quickly. Since analytically there exists no damping within the Cowling approximation, it has to be purely numerical. As will be shown in Sec. 3.3.2 and Sec. 3.2.5, the damping is caused mainly by the treatment of the stellar surface.

Using the mode recycling technique described in Sec. 2.3.4, it is possible to excite single modes selectively. Fig. 3.18 shows the ${}^2p_1$-oscillation. A prominent feature, which is generally found for small oscillation amplitudes ($A < 0.001$), is that the decay of the amplitude is almost perfectly exponential. This indicates that the numerical errors responsible for the decay are proportional to the perturbation from equilibrium.

As shown in Fig. 3.19, other modes can be suppressed almost completely, at least for good resolutions above 150. For lower resolutions, this is not always the case, especially for the rotating model.

Since many numerical artifacts cannot be detected by looking at time-series at a sample point, 2D cuts of the final state are plotted automatically for every simulation. Fig. 3.20 shows the velocity field at the end of a typical simulation. The only visible artifacts are small spots of slightly increased velocity at the surface.

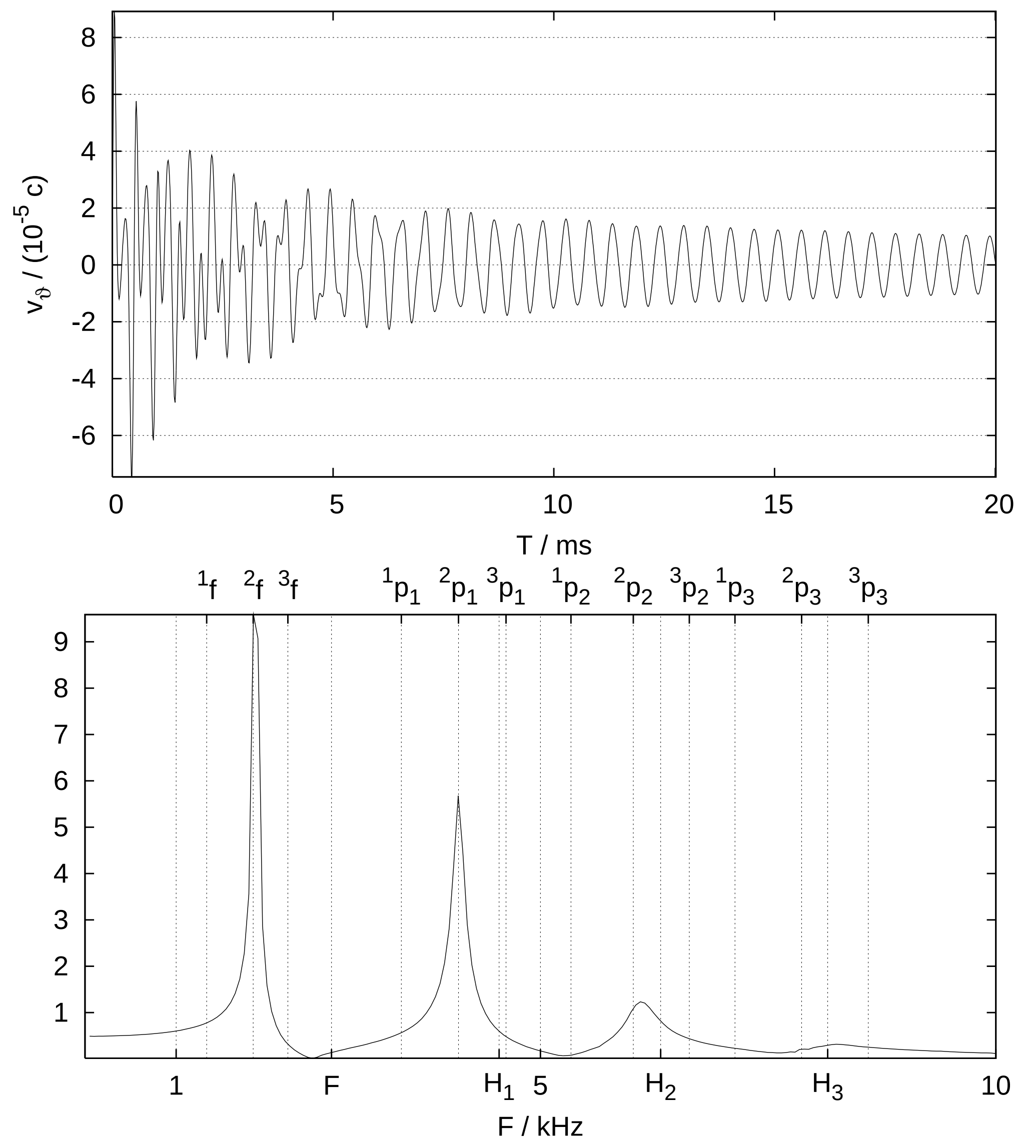


Figure 3.17: Results of a 3D simulation of star model N, with a resolution of 47^3 points. The initial data was perturbed with $l = 2, A = 0.001$. Shown are time evolution and corresponding Fourier spectrum of the velocity in ϑ-direction on the space diagonal at $r = 7$ km. The vertical lines mark mode frequencies for the same star model as reported by [21].

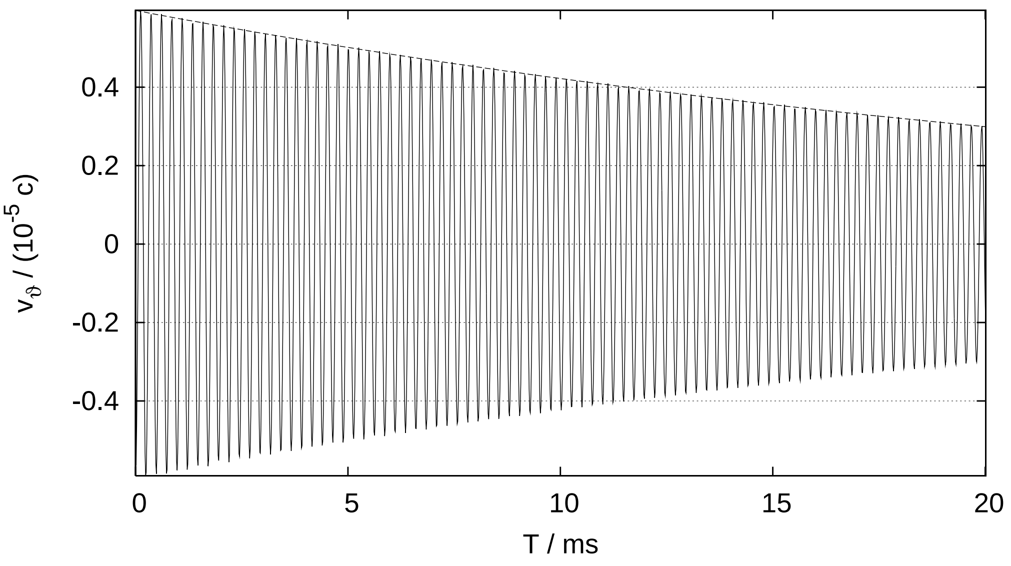


Figure 3.18: Results of a 2D simulation of model N, with a resolution of 189^2 points. The initial data was perturbed using the eigenfunction of the ${}^2p_1$-mode which was obtained via mode recycling. Shown is the time evolution of the velocity in ϑ-direction at $r = 7$ km, $\vartheta = \pi/4$. The dashed line is an exponential fitted to the maxima of the oscillations.

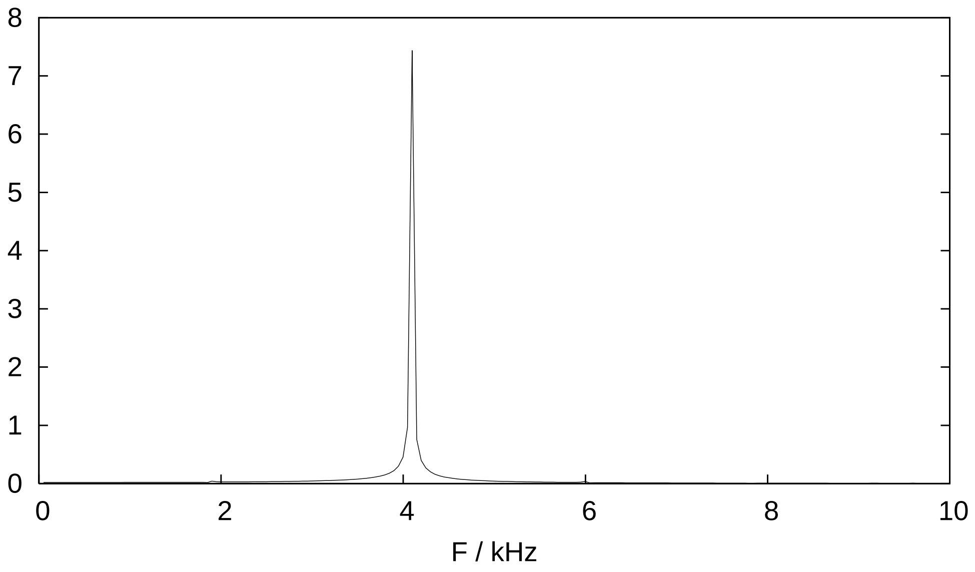


Figure 3.19: Fourier spectrum corresponding to Fig. 3.18. Before applying the Fourier transformation, the exponential decay was divided out, resulting in a slightly sharper peak.

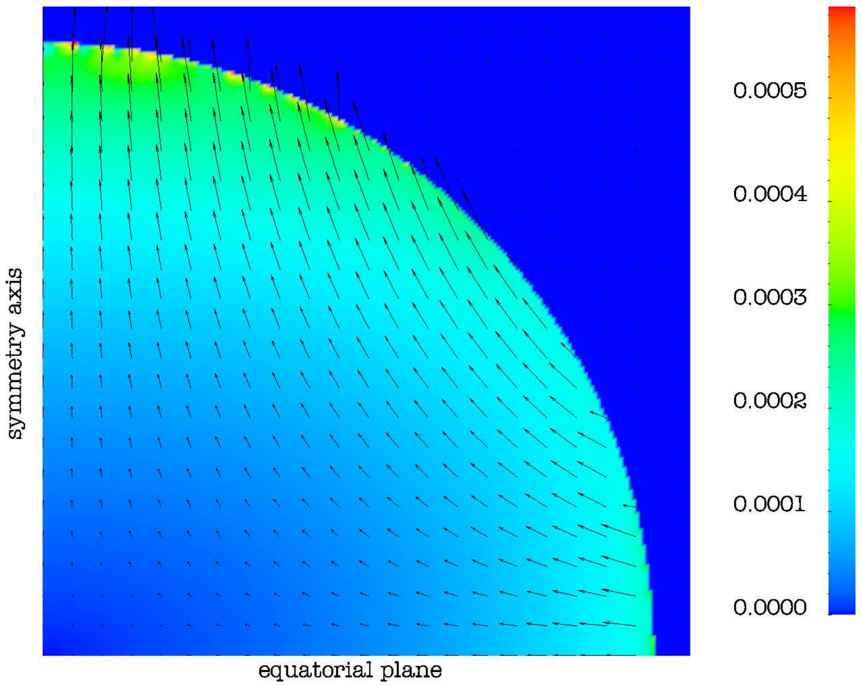


Figure 3.20: Velocity field after an evolution time of 10 ms, for a 2D simulation of model N. The initial data was perturbed with the ${}^2f$-eigenfunction. The resolution is 189^2 points. Colour coded is the physical velocity in units of c, the arrows represent the coordinate velocity.

Considering the crude treatment of the stellar surface, such artifacts are to be expected. However these artifacts are not a problem since their magnitude is completely tolerable. In general, the maximum velocities at the surface are bounded by the oscillation amplitude. This is different for codes using artificial atmospheres. In that case, regardless of the oscillation amplitude, there is typically a minimum amount of noise at the surface.

To study the robustness of the numerical scheme, I carried out a simulation with an extremely low resolution of 10^3 points. The original scheme would in that case be unable to obtain meaningful results at all. With the new scheme, this is different, as shown in Fig. 3.21. The numerical damping is of course much higher now, which allows one to observe how the system neatly settles down to its equilibrium state. As shown in Fig. 3.21, the oscillation amplitude spans eight orders of magnitude without any visible changes. In fact, the numerical noise becomes important for oscillations with velocities as low as $10^{-14}c$. This behaviour is a design feature of the scheme and not caused by the high damping at low resolution; it can be observed at higher resolutions as well.

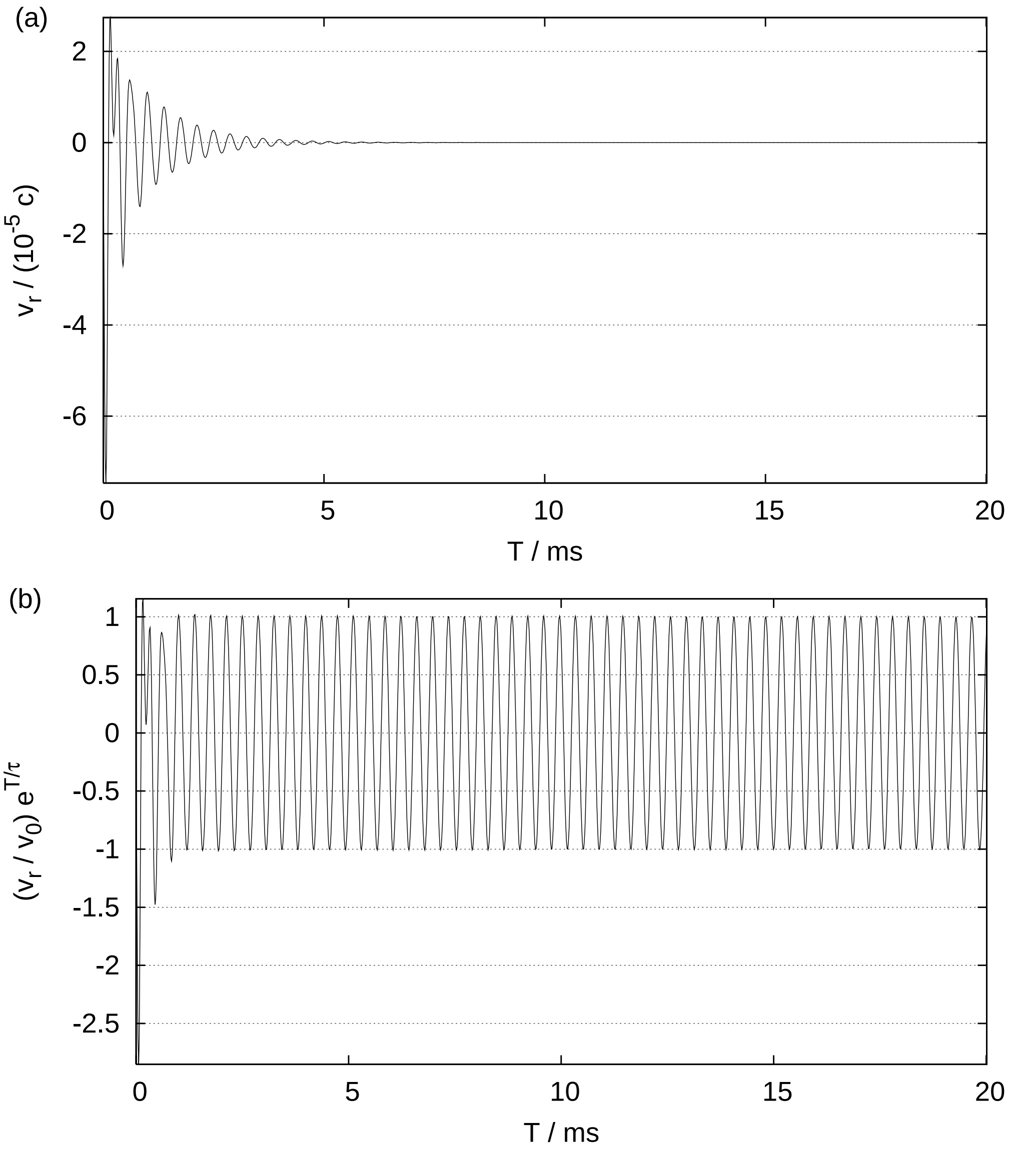


Figure 3.21: Results obtained with a resolution of only 10^3 points in 3D, for star model N and a polytropic EOS. The initial data was perturbed with $l = 0, A = 0.001$. (a) Time evolution of the radial velocity v_r at $r = 7$ km on the space diagonal. (b) Radial velocity divided by an exponential decay ($\tau = 1.07292$ ms) fitted to its maxima. Note the higher modes are damped away almost instantaneously.

The ability to evolve very small oscillations as well as large ones might be useful for studies of nonlinear phenomena, where ideally the transition from a clearly linear to the nonlinear regime should be investigated, with a good signal-to-noise ratio for the whole range of amplitudes.

3.2.5 Convergence tests

In this section, I study the convergence behaviour of the new scheme, for oscillations of the nonrotating neutron star model N. This is particularly important since the convergence properties have not been investigated analytically.

For this task, I performed simulations of single oscillation modes, excited via mode recycling, using various resolutions. I choose the radial F- and H_1-modes, which allows for a direct comparison between 1D,2D, and 3D simulations. Using a single mode also makes it possible to fit a damped oscillation of the form

$$y(t) = y_0 e^{-t/\tau} \cos(2\pi f t) \tag{3.12}$$

to the evolution of the velocity components at some sample point. This yields a more precise measure of the frequency than using a Fourier spectrum. Also the exponential decay time is determined this way, providing a quantitative measure of the numerical damping.

As it turns out, the convergence behaviour for the star model is quite complex. The main difficulty is that the numerical damping strongly depends on the threshold density parameter ρ_v of the surface treatment scheme: At a resolution of $N = 142$ points in 2D, the damping speed roughly doubles when increasing ρ_v by a factor of six, starting from $4.2 \cdot 10^{-4} \rho_c$. Lowering ρ_v however does not decrease the damping further, but the scheme gets unstable at some point. This proves that the surface treatment causes a significant part of the damping. Before doing convergence tests, it is necessary to set up a rule how to choose ρ_v for a given resolution. The following rule, which has been found empirically and which is used for all simulations shown here, guarantees stability and keeps the numerical damping near its minimum: For a resolution N, set $\rho_v = 6 \cdot 10^{-2} \rho_c / N$.

The second complication only affects the 1D case: The frequency error and the damping timescale strongly depend on the location of the star surface inside the surrounding cell. This was found by slightly changing the grid spacing such that the surface is located at different locations inside the same cell. The results are shown in Fig. 3.22. At a resolution of 142, the frequency error varies between -2% and 0.5%, and the damping timescale τ between

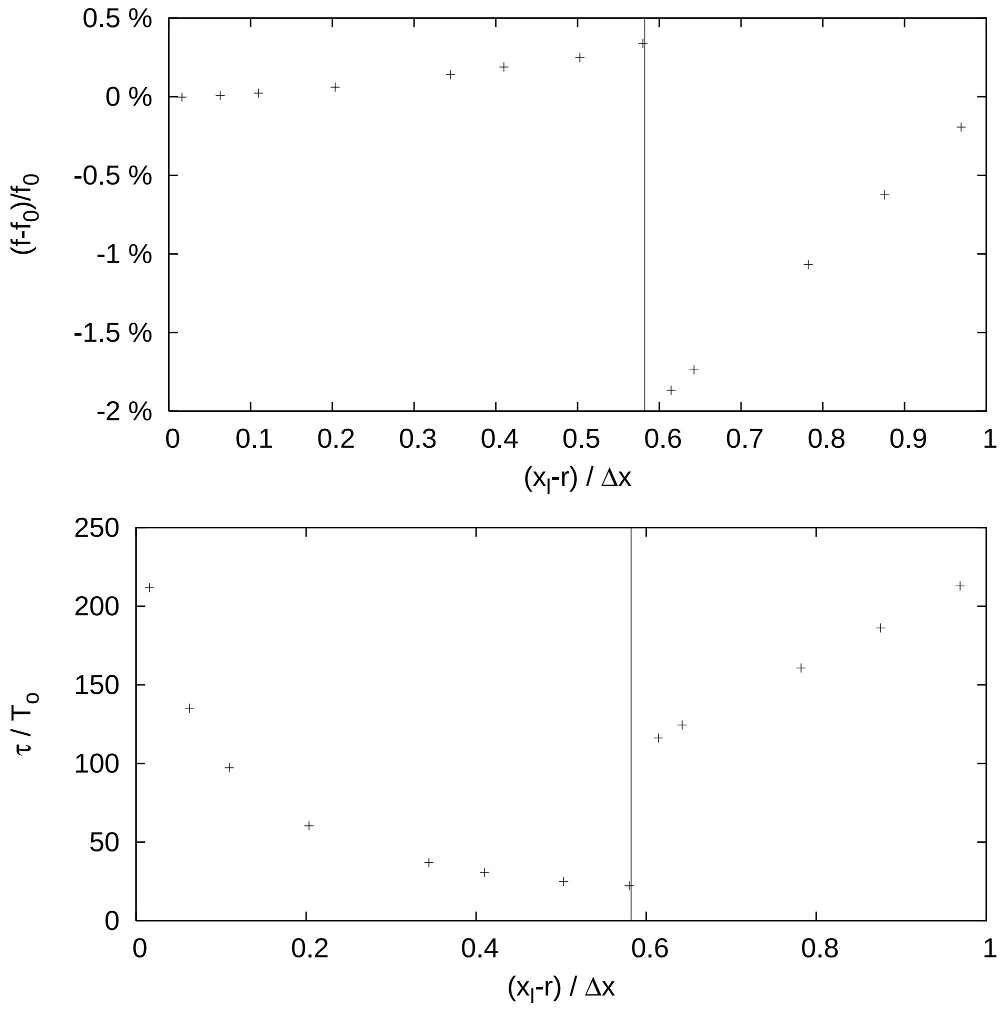


Figure 3.22: Dependency of the frequency error and numerical damping on the location of the star surface relative to the numerical grid, for the H_1-mode of model N, evolved in 1D. For all simulations, the surface was located in the same cell, and the resolution was $r/\Delta x \approx 142$, where Δx is the grid spacing and r the star radius. The x-axis gives the distance from the star surface (in equilibrium) to the position x_l of the left face of the grid cell containing the surface. The vertical line marks the case where the density at the cell centre equals the density threshold parameter ρ_v. Top: Frequency error, estimated using the frequency f_0 computed in high resolution (1133 points). Bottom: Numerical damping time in units of the oscillation period T_o.

22 and 220 oscillation periods. The worst case occurs when the density at the centre of the cell containing the surface just reaches the threshold parameter, and the best case when the surface coincides with the cell boundary.

Due to this issue, no satisfying convergence could be observed in 1D at first; the location of the surface relative to the grid is effectively random when changing the resolution while keeping the computational domain fixed, and the influence of the location is comparable to the one of the resolution. To overcome this problem and compare the same case in different resolutions, the grid spacing for the 1D convergence test shown in the following was chosen in a way that the star surface is always located at a position near the left cell boundary, $0.05\Delta x$ away from it. I have to stress that without this adjustment, the 1D simulations expose bigger errors which are quite irregular as a function of resolution. In 2D and 3D, no such adjustment was necessary, because in that case the surface crosses many cells, and since cylindrical or Cartesian coordinates are used, it does so at different locations inside the cells. Hence the irregularities are averaged away to some extend.

Fig. 3.23 shows the variation of the mode frequencies for different resolutions. The frequencies seem to converge, but clearly not following a power law. This might indicate that the regime where only the errors with the lowest convergence order are dominant starts at higher resolutions than the ones investigated, or that the errors induced by the surface treatment simply do not follow a power law.

Nevertheless, I estimate the errors $\delta f/f$ due to the finite resolution below 1% for $N > 50$, and below 0.5% for $N > 1000$. The real error might be smaller, but without a recognisable convergence law, one cannot tell. This estimate will also be used for the low order nonradial modes.

As shown in Fig. 3.24, the damping times do roughly follow a power law with a convergence order in the range $1.6 - 1.8$ (2D/3D) and 2.3 in 1D. The big difference between 1D and 2D/3D may be due to the aforementioned special adjustment of the grid spacing applied in 1D. For infinite resolutions, the convergence order should not depend on the mode, as it seems to be the case for the F- and H_1-modes in 2D. However, the resolutions are finite and ideal convergence behaviour cannot be expected. Also the differences between the F- and H_1-mode is small.

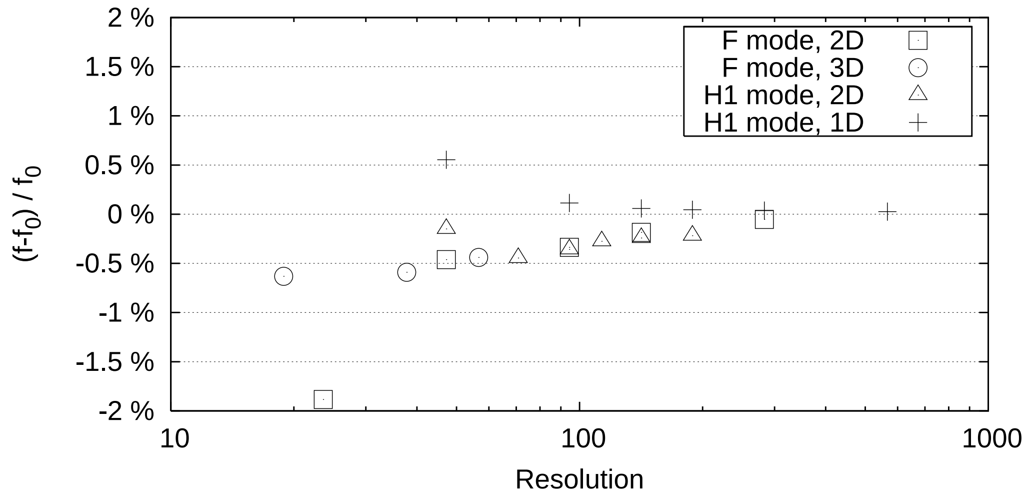


Figure 3.23: Convergence behaviour of the frequencies for the F- and H_1-modes of model N. f_0 is the frequency obtained from a 1D simulation with a resolution of 1133. In detail $f_0 = 2.6845679$ kHz for the F-mode and $f_0 = 4.5456764$ kHz for the H_1-mode.

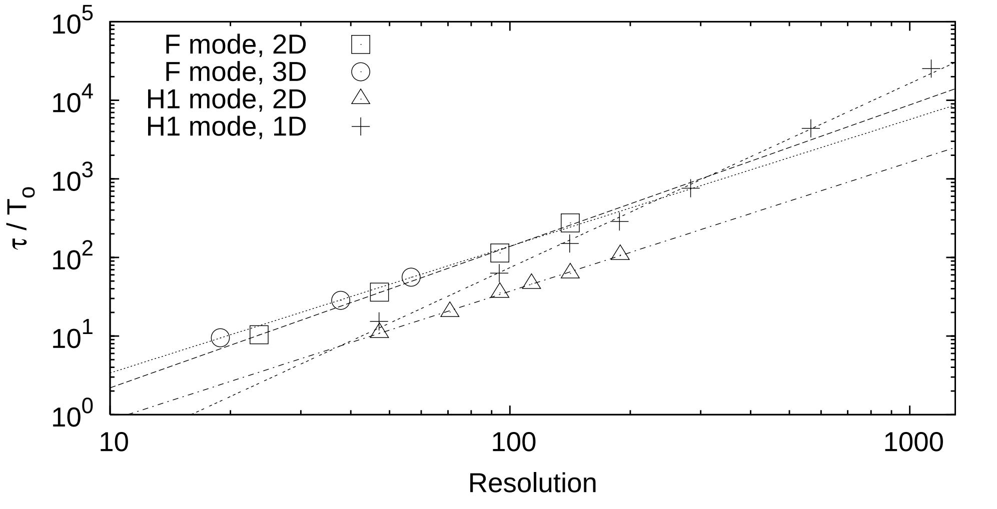


Figure 3.24: Convergence of the numerical damping timescale τ in units of the respective oscillation period T_o, for the F- and H_1-modes of model N. The straight lines are fits of the convergence law $\tau = a\Delta x^p$ to the data. The convergence order is $p = 1.62$ (H_1-mode, 2D), $p = 1.80$ (F-mode, 2D), $p = 1.61$ (F-mode, 3D), and $p = 2.34$ (H_1-mode, 1D). Note: the damping time for the highest resolution 2D F-mode run could not be determined due to insufficient evolution time (10 ms).

3.3 Toy model tests

The convergence tests for star model N clearly show that the treatment of the star surface has a significant influence on the numerical errors. To separately investigate the numerical errors caused by the evolution scheme itself and the ones caused by the crude treatment of the star surface, I introduce another testbed without a fluid-vacuum boundary.

3.3.1 Toy model

The toy model testbed consists of an artificial, unphysical metric (i.e. not a solution of the Einstein equations) which is periodic in all directions, and a fluid which is at equilibrium in the artificial gravitational potential. In detail, the artificial metric is given by:

$$\alpha = 1 - (1-\alpha_c)\prod_{i=1}^{d}\frac{1}{2}(1+\cos(\pi x^i/L)) \tag{3.13}$$

$$g_{ij} = \delta_{ij} \quad , \quad \beta^i = 0 \tag{3.14}$$

where d is the dimensionality of the problem. Note that in stationary spacetimes, the lapse function (shown in Fig. 3.25) takes over the role of the Newtonian gravitational potential. The computational domain is $[0..L]^d$ and reflecting boundary conditions are applied, which are compatible with the symmetries of the problem. The density profile is obtained from Eq. (1.47) by setting $w^i = 0$ and prescribing a central density ρ_c such that $\rho = \rho_b > 0$ at the outer boundaries. The EOS is the same polytrope as for the star testbeds. I choose the parameters in a way that the system mimics a neutron star with respect to the magnitude of "gravitational" and pressure forces. In detail, $L = 15$ km, $\alpha_c = 0.82$, and $\rho_c = 7\cdot 10^{17}$ kg/m^3. It follows that $\rho_b = 0.026\ \rho_c$. Fig. 3.26 shows a comparison of the profile to the one of model N.

For the numerical tests $d = 1$ and $d = 2$ are used with the same 3D-code as for the star testbeds, by assuming translational symmetry in the unused direction(s). Note that in contrast to the spherical star test, $d = 1$ and $d = 2$ yield physically different systems.

3.3.2 Convergence

Using the toy model described in the previous section, I study the convergence behaviour of the evolution scheme itself, without interference from the surface treatment.

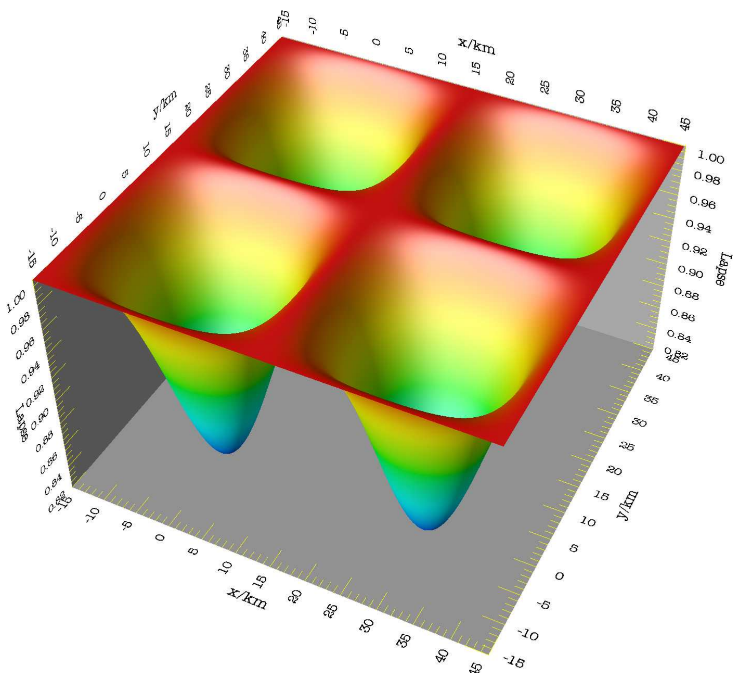


Figure 3.25: Lapse function of the toy model, plotted over a wide range to show its symmetries. The computational domain covers only one quadrant of the cell at the origin.

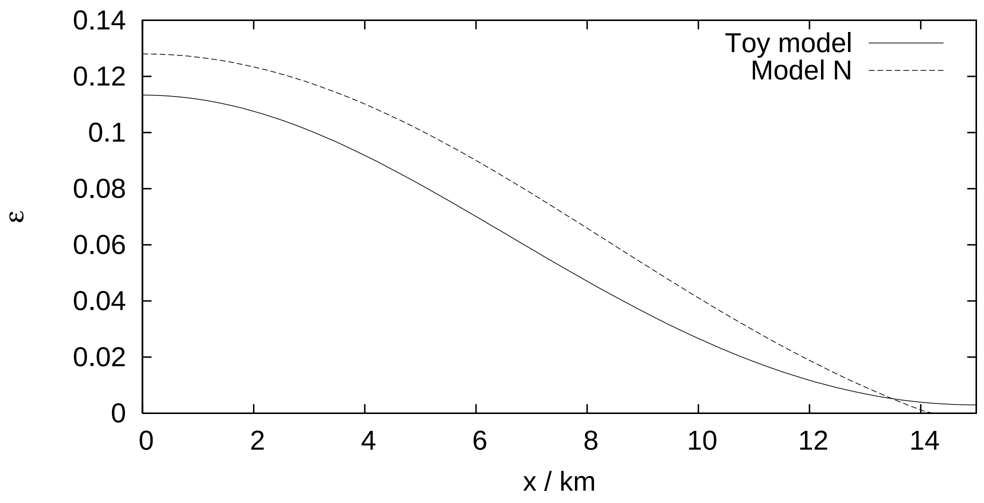


Figure 3.26: Profile of specific energy density ϵ for the toy model in comparison to the one of star model N. Both are plotted along the x-axis. The profile of the toy model is mirror symmetric not only at the origin but also around $x = 15$ km, and contains no vacuum.

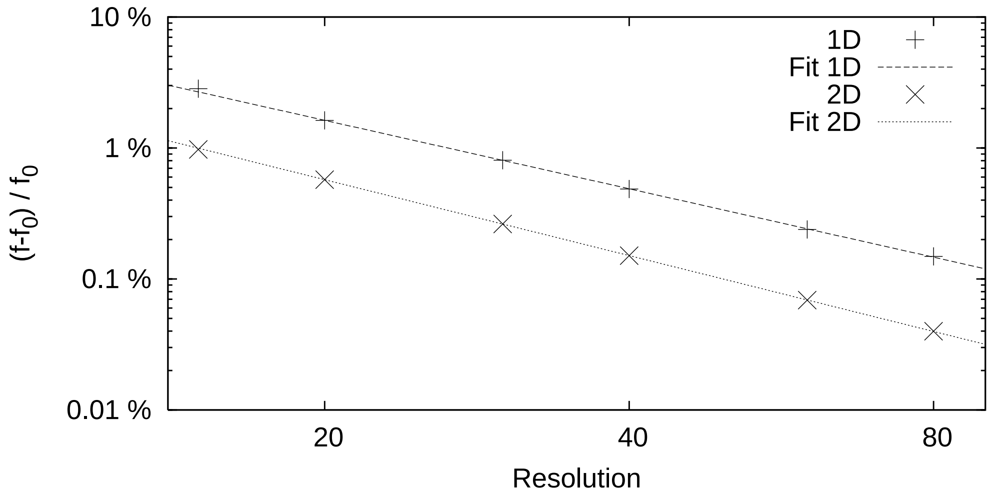


Figure 3.27: Convergence of the frequency for the lowest (known) mode of the 1D and 2D toymodels. Shown is the relative difference to the "true" frequency f_0 obtained by fitting convergence law Eq. (3.15) to the data. The fit results are $f_0 = 2.181$ kHz, $p = 1.74$ in 1D and $f_0 = 1.257$ kHz, $p = 1.93$ in 2D. The lowest resolution ($N = 15$) was excluded for the fitting.

As for the star model tests, single modes are excited via mode recycling. I use the lowest mode excited by some trial perturbation, which has a frequency of 2.181 kHz for the 1D model, and 1.257 kHz for the 2D model. The excited oscillation amplitudes are small, with typical velocities of $10^{-4} - 10^{-5}c$.

The frequencies obtained for the different resolutions follow the convergence law

$$f = f_0 + \frac{a}{N^p} \tag{3.15}$$

where N is the resolution and $p = 1.74$ in 1D, or $p = 1.93$ in 2D respectively. Fig. 3.27 shows the fit of Eq. (3.15) to the numerical results. In comparison with the star model, the frequency errors not only follow a recognisable convergence law, but they are also smaller. The damping times shown in Fig. 3.28 also follow a power law, with a convergence order of $p = 3.24$ in 1D and $p = 3.12$ in 2D. Again, this is much better than for the star simulations.

The comparison between the toy and the star models, and the problems in 1D for the latter, prove that the numerical error of the star simulations is mainly caused at the star surface. Any effort to improve the effective accuracy has to start with a more sophisticated way of treating the surface.

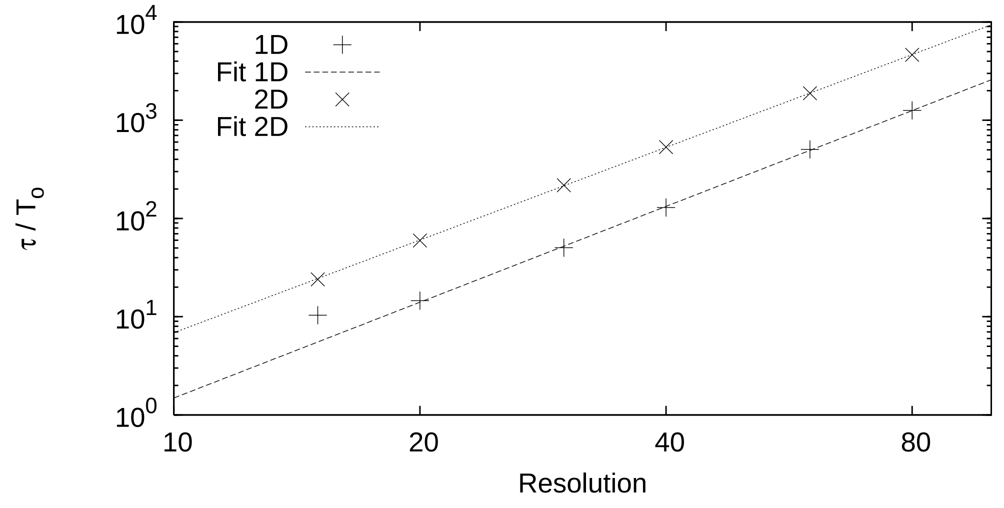


Figure 3.28: Convergence of the numerical damping timescale τ (in units of the oscillation period T_o) for the lowest (known) mode of the 1D and 2D toymodels. The convergence orders obtained from the fits (straight lines) to the convergence law $\tau = bN^{-p}$ are $p = 3.24$ (1D) and $p = 3.14$ (2D). The lowest resolution ($N = 15$) was excluded for the fitting.

3.4 Evolution with the energy equation

In this section, I investigate the behaviour of the new scheme in simulations of stars with the 2-parametric ideal gas EOS, which is given by

$$P(\rho, \epsilon) = (\Gamma - 1)\rho\epsilon \tag{3.16}$$

For the ideal gas case, there is an additional degree of freedom in the evolution system compared to the polytropic case. Nevertheless the evolution has to evolve adiabatically unless shocks are present, i.e. the specific entropy along fluid worldlines cannot change. If the system is isentropic in the beginning, it has to stay isentropic. Assuming that for small amplitude star oscillations, no shocks will form, the deviation from isentropy is therefore an additional measure for the numerical accuracy.

An important difference to the polytropic case is that the presence of entropy gradients due to numerical errors can lead to convective movements. This will be most visible in the numerical results for nonrotating stars, for which stationary solutions with mass flows in the meridional plane exist. These can be easily driven by convection since only the kinetic energy has to be provided; there is no restoring force. In the rotating case on the other hand, the Coriolis force counteracts movements in the meridional plane.

The actual behaviour of the numerical scheme with the ideal gas EOS was studied mainly for simulations of model N. Although this model is constructed using a polytropic EOS, it is also an isentropic equilibrium state when using the ideal gas EOS. This is because polytropes are the curves of constant specific entropy for the ideal gas EOS with the same constant Γ. Therefore, the ideal gas EOS with $\Gamma = 2$ is used to evolve model N.

To measure the constancy of specific entropy, one can use the constant ρ_p characterising the polytropic EOS as a variable to parametrise the fluid state, defining

$$\rho_p = \left(\frac{P}{\rho^\Gamma} \right)^{\frac{1}{\Gamma - 1}} \tag{3.17}$$

For adiabatic evolution, ρ_p has to stay constant along fluid worldlines, and globally constant in the isentropic case of model N.

As shown in Sec. 2.2.3, isentropic stationary equilibrium states, like model N, should be preserved exactly even for the case of a 2-parametric EOS like the ideal gas. Various simulations for models N and R in 1D (possible only for model N), 2D, and 3D proved that the `Pizza` code indeed preserves the initial data as well as for the polytropic case. Results for a very low resolution are shown in Fig. 3.29.

To produce some errors, I excite oscillations with an amplitude of $\delta\rho/\rho \approx 10^{-4}$, assuming that oscillations of this magnitude do not lead to the formation of shocks. Fig. 3.30 shows the violation of isentropy for different resolutions in 2D, after an evolution time of 20 ms. Obviously, the change of ρ_p is most prominent at the surface and near the axis of axisymmetry. With increasing resolution, the errors are concentrated more sharply at the surface, but the magnitude decreases only slowly.

Another significant feature shown in Fig. 3.30 are the large scale vortices in the velocity field, which start to develop after a typical timescale of 5 ms. The velocity profile corresponds to a nearly stationary flow; the density profile stays constant up to 0.5% for $N = 50$, and up to 0.1% for $N = 200$.

I suppose the vortices are caused by small entropy gradients inside the star (not the ones at the surface), which are induced by the numerical errors. However, this issue has to be investigated further before drawing a conclusion. In a simulation of the rotating model (with a resolution of $N = 200$), no vortices formed, as predicted.

For model N and a resolution of $N = 200$, the oscillation frequencies extracted from the Fourier spectrum agree very well with the results obtained with the polytropic EOS shown in Sec. 4.1.1. The differences are $\delta f/f = 0.5\%$ for the F-mode, 0.1% for the H_1-mode, and 0.2% for the H_2-mode.

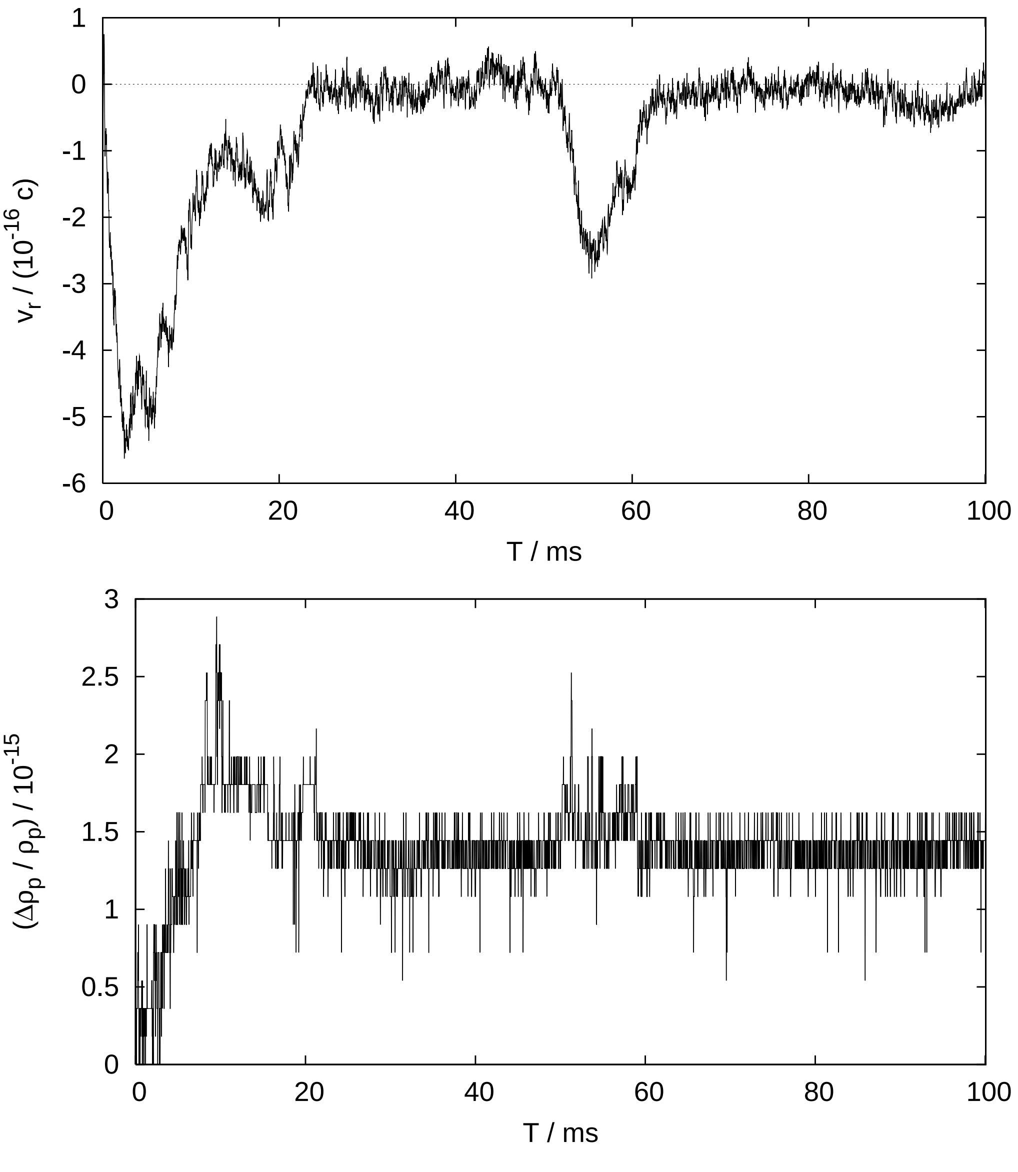


Figure 3.29: Equilibrium preservation for model N when using the ideal gas EOS for the evolution. The resolution is 10^2 cells in 2D. Top: radial velocity at $r = 7$ km, $\vartheta = \pi/4$. Bottom: Relative change of ρ_p defined by Eq. (3.17). For adiabatic evolution, ρ_p has to stay fixed.

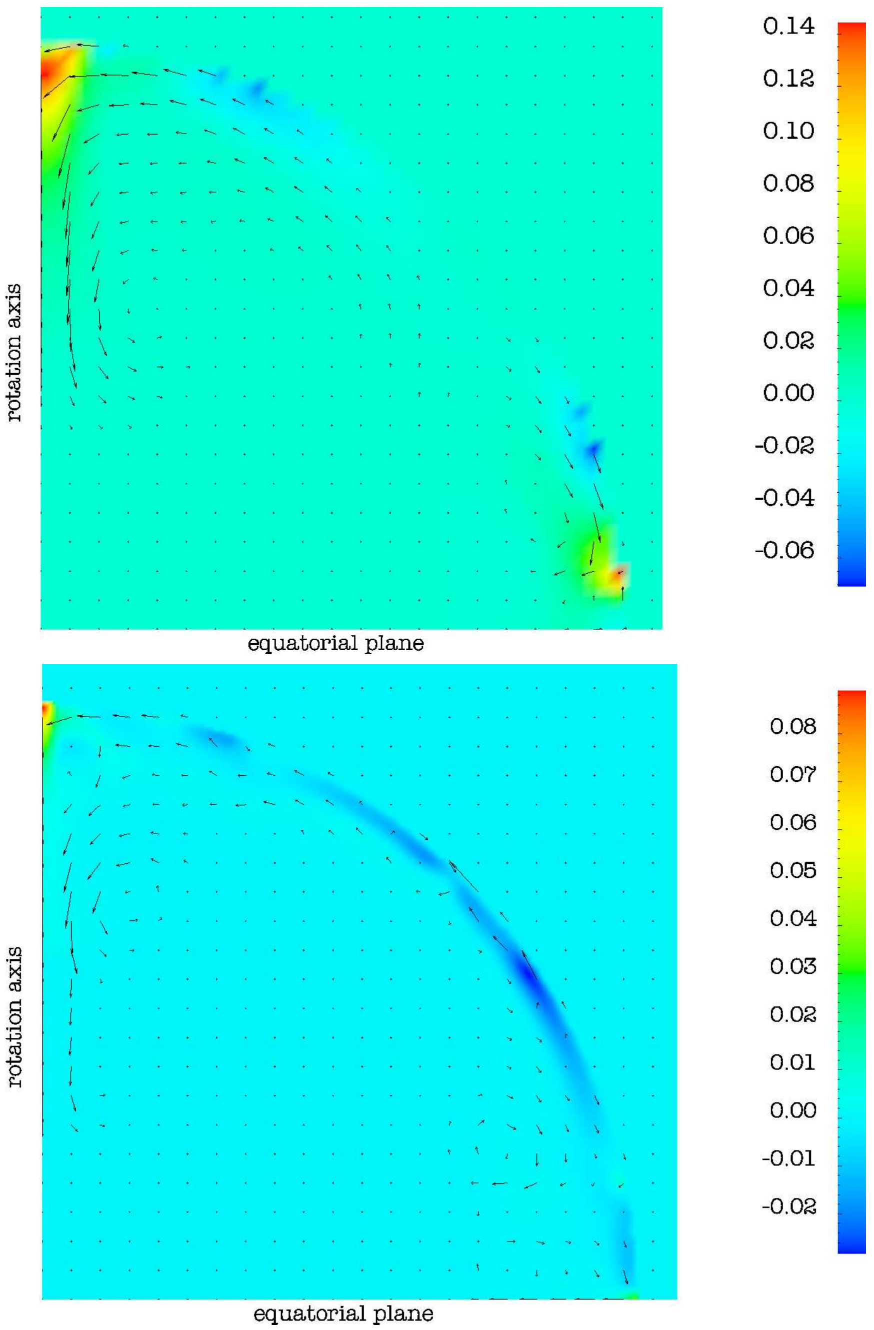


Figure 3.30: Relative change of ρ_p after 20 ms evolution of model N with the ideal gas EOS. Resolution is 50^2 (Top) or 200^2 (Bottom). The arrows represent the coordinate velocity.

If not at the surface, there should be at least a reasonable convergence behaviour in the interior of the star. To investigate the convergence, I study the violation of adiabaticity at a sample point inside the star for three resolutions. It turns out that with increasing resolution, the isentropy violation follows a convergence law for approximately 1 ms, and behaves more complex for longer evolution times. Fig. 3.31 shows the deviation from adiabatic evolution. The error converges roughly with second order accuracy during the first millisecond. Afterwards, the convergence does not follow a power law, but the error converges faster than second order, at least for the investigated resolutions.

Although ρ_p is preserved to good accuracy inside the star, one has to relate its change to the oscillation amplitude, i.e.

$$a \equiv \frac{\delta\rho_p}{\rho_p}\left(\frac{\delta\rho}{\rho}\right)^{-1} \tag{3.18}$$

has to be small for adiabatic oscillations. For the simulations shown in Fig. 3.31, the relative change of restmass density is around 10^{-4}. From this, we obtain $a < 1.4 \cdot 10^{-2}$ for $N = 50$, $a < 2 \cdot 10^{-3}$ for $N = 100$, and $a < 6 \cdot 10^{-5}$ for $N = 200$. Hence the oscillation is adiabatic to good accuracy inside the star. Note however that near the surface, the oscillation is not adiabatic at all, i.e. $a > 1$.

Results for the 1D case are shown in Fig. 3.32. The error is comparable with the 2D case. Also for 3D simulations the results are very similar, but the errors near the axis of axisymmetry visible in Fig. 3.30 disappear. This indicates that those errors are caused by the degeneration of the cylindrical coordinates near the axis, which is not present in Cartesian coordinates.

A possible explanation for the more irregular long term behaviour of the adiabatic deviations would be that different types of errors interact and form a nonlinear system. The deviation from adiabatic evolution depends on the deviation of the system from stationarity, for which the the numerical scheme gives exact results. For the neutron star tests, deviations from stationarity are given by oscillations, but also by convective movements induced by entropy gradients, which in turn are caused by the deviation from adiabatic evolution. With increasing resolution, the system will settle to its stationary ground state more slowly, which by itself leads to a *greater* deviation from adiabaticity after a given evolution time. On the other hand, the errors in adiabaticity for a given deviation from stationarity will get smaller with increasing resolution. The net error after long evolution times is therefore the result of a complex interplay of several effects. However, this model still has to be substantiated quantitatively.

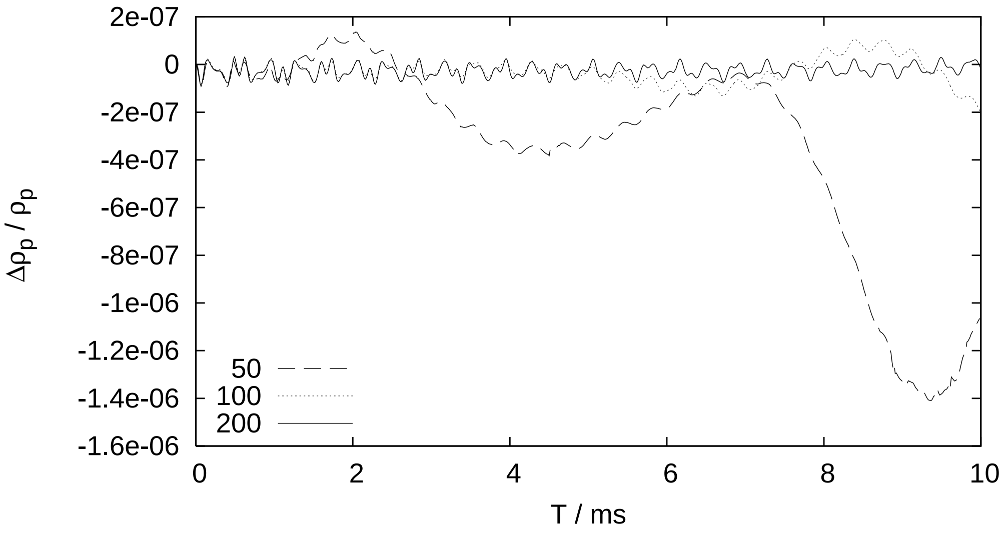


Figure 3.31: Change of the polytropic constant ρ_p at $r = 7$ km, $\vartheta = \pi/4$ during evolution with the ideal gas EOS. Shown are results for different resolutions in 2D. The plots for $N = 100$ has been scaled by a factor of 4, the one for $N = 200$ by a factor of 16.

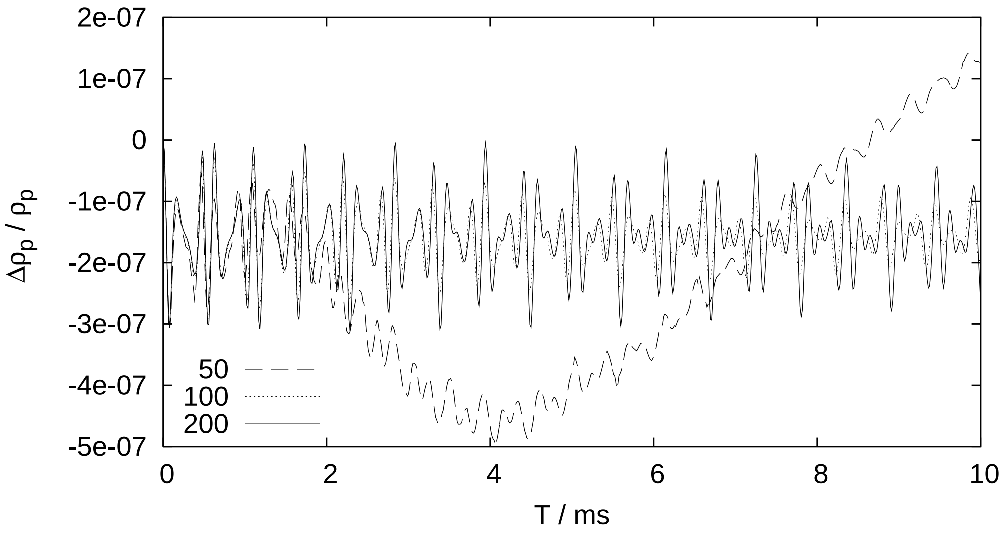


Figure 3.32: Change of the polytropic constant ρ_p at $r = 7$ km during evolution with the ideal gas EOS. Shown are results for different resolutions in 1D. The plots for $N = 100$ has been scaled by a factor of 4, the one for $N = 200$ by a factor of 16.

For the ideal gas case, there are still many open questions. First, it should be investigated how the adiabaticity of the evolution depends on the oscillation amplitude. Second, the convergence tests shown for the polytropic case have not been repeated yet for the ideal gas case due to time constraints. For this, one would also need to use different quality measures than for the polytropic case to accommodate to the presence of convective movements.

The tests shown here deal with small deviations from stationarity, and contain no shock waves. To study the behaviour in highly dynamical systems with strong gravitational forces, there is still need for a testbed (maybe using artificial spacetimes) with known solutions. HRSC schemes are specially designed to handle shocks, but only in flat space. One should therefore study the propagation of shocks along a star profile, at least in 1D. The shock tube tests in flat space cannot assure the correct treatment of shocks in strong gravitational fields.

Chapter 4

Neutron star oscillations

As a first application, after the convergence and stability of the code has been established, I investigate oscillations of rotating and nonrotating neutron stars in the Cowling approximation. Additionally, I present first results with coupled spacetime evolution for a nonrotating star, although this is ongoing work.

4.1 Frequencies and Eigenfunctions

In this section, I present the frequencies and eigenfunctions of the seven lowest order axisymmetric oscillation modes with $l = 0$ and $l = 2$, in Cowling approximation, for the stellar models N and R.

For both models, the mode frequencies can also be found in [21, 54], where a code called `ToniK` is used. In contrast to `Pizza`, the `ToniK` code uses a standard HRSC scheme, a spherical coordinate system, and the artificial atmosphere method to treat the surface. Hence the comparison of the frequencies provides a meaningful validation for both codes.

The eigenfunctions are not available in the literature; with the `ToniK` code, only the frequencies were extracted. For comparison, I use (unpublished) eigenfunctions provided by [13], which have been obtained with the `CoCoNuT` code described in [17, 16], but using the Cowling approximation.

4.1.1 Frequencies

To obtain the frequency of a given mode, it was selectively excited by means of the mode recycling procedure described in Sec. 2.3.4. If only one mode is present in the evolution, the frequency can be extracted more accurately by a fitting procedure than by using a Fourier spectrum. For this, the damped oscillation given by Eq. (3.12) is fitted to the time evolution of the velocity at a suitable sample point (at which the mode has no node).

Although the frequencies are measured with respect to the coordinate time, they are identical to the physical frequencies measured by normal observers at infinity, where the spacetime is asymptotically flat. This is because in the chosen coordinates, the spacetime is manifestly stationary. Imaginary light signals starting at point A at regular intervals with respect to coordinate time arrive at point B in the same interval with respect to coordinate time. Since the lapse function was normalised to a value of 1 at infinity, the coordinate time at infinity is equal to the proper time of a normal observer. Thus the frequency measured with respect to coordinate time equals the physical frequency a normal observer at infinity would see.

Tables 4.1 and 4.2 show the frequencies of the seven lowest order modes. The errors originating from the fitting procedure are generally negligible compared to the simulation errors. For the H_3- and ${}^2p_2$-modes of model R however, the excitation of other modes could not be suppressed strongly enough to allow for the fitting procedure above. Instead, the frequencies were obtained from the Fourier spectrum. The error due to its finite resolution is included in the error estimates stated in Table 4.2.

My estimates for the frequency errors are based on the convergence tests shown in Sec. 3.2.5, in particular Fig. 3.23. For this, one also has to consider that for higher order modes a better resolution is needed to achieve the same accuracy, since the distances between the nodes are smaller, and the eigenfunctions concentrate near the problematic star surface.

Compared to the results from [21], the mode frequencies agree very well. By far the biggest difference of 1.7% is found for the ${}^2f$-mode, which is still inside the error range given by [21]. For the available frequencies computed with the `CoCoNuT` code, shown in Table 4.3, the maximum deviation of 0.7% is even smaller.

For the rotating model, low frequency modes at 0.80 ± 0.1 kHz and 0.38 ± 0.1 kHz occurred in addition to the previously described modes. A spectrum featuring the 0.80 kHz mode is shown in Fig. 4.1. The only known low frequency modes for isentropic stars in Cowling approximation are rotation modes, i.e. modes for which the Coriolis force is the restoring force. For a discussion of r-modes, see [32]. The eigenfunctions corresponding to the unknown modes have not been extracted yet and their frequencies for model R are not available in the literature. Therefore, the identity of these modes remains unclear.

For some reason, the suppression of the unknown modes via mode recycling is difficult when the H_3- or ${}^2p_2$-modes have been excited. The amplitude ratio of 9% in Fig. 4.1 is the best suppression achieved so far.

Resolution	Mode	f/kHz	$\delta f/f$	f_1/kHz	$(f_1 - f)/f$
1D 1133	F	2.6846	0.5 %	2.706	0.80 %
1D 1133	H_1	4.5457	0.5 %	4.547	0.03 %
1D 1133	H_2	6.3343	0.5 %	6.320	−0.23 %
1D 1133	H_3	8.0970	0.5 %	8.153	0.69 %
2D 189	${}^2f$	1.8786	1.0 %	1.846	−1.7 %
2D 189	${}^2p_1$	4.0919	1.0 %	4.100	0.20 %
2D 189	${}^2p_2$	6.0068	1.0 %	6.019	0.20 %

Table 4.1: Best results for the oscillation frequencies f of model N, and comparison to the frequencies f_1 as published in [21]. δf is the estimated frequency error, compare Sec. 3.2.5. According to [21], the error of f_1 is in the range 1–2%.

Mode	f/kHz	$\delta f/f$	f_1/kHz	$(f_1 - f)/f$
F	2.5626	1.0 %	2.579	0.64 %
H_1	4.3911	1.0 %	4.385	−0.14 %
H_2	6.2333	1.0 %	6.234	0.01 %
H_3	8.0877	1.6 %	8.096	0.10 %
${}^2f$	1.8893	1.0 %	1.857	−1.7 %
${}^2p_1$	3.8111	1.0 %	3.814	0.08 %
${}^2p_2$	5.5000	1.9 %	5.521	0.38 %

Table 4.2: Like Table 4.1, but for model R. All simulations were performed in 2D with a resolution of $R_p/\Delta x \approx 149$, $R_e/\Delta x \approx 175$, where R_p and R_e are the star's polar and equatorial coordinate radius and Δx is the grid spacing.

Resolution	Mode	f/kHz	$\delta f/f$	f_2/kHz	$(f_2 - f)/f$
1D 1133	F	2.6846	0.5 %	2.700	0.57 %
1D 1133	H_1	4.5457	0.5 %	4.567	0.47 %
2D 189	${}^2f$	1.8786	1.0 %	1.891	0.66 %
2D 189	${}^2p_1$	4.0919	1.0 %	4.121	0.71 %

Table 4.3: Oscillation frequencies f of model N in comparison to the frequencies f_2 obtained with the `CoCoNuT` code.

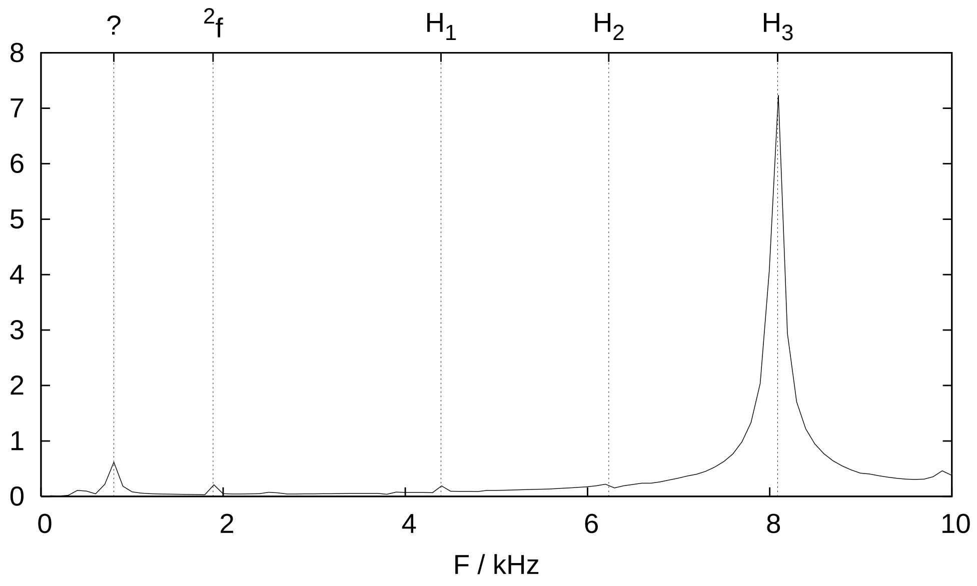


Figure 4.1: A Fourier spectrum of v^ϑ for model R, which shows an unidentified low frequency mode.

4.1.2 Eigenfunctions

In the following, the eigenfunctions of the low order oscillation modes are shown, which have been extracted during the simulations performed to compute the frequencies shown in the previous section.

The error of the numerically extracted eigenfunctions depends mainly on two factors. First, it depends on the accuracy of the simulation itself, which depends on the resolution. Second, since the Fourier transform is taken only over a finite time interval, the presence of other modes induces an error. The time interval is restricted not only by the evolution time, but also by the numerical damping timescale of the mode.

For most simulations, the unwanted modes could be suppressed almost completely, like in Fig. 3.19. Only for the H_2-, H_3- and ${}^2p_2$-modes of the rotating star model, other modes are still present, with amplitudes up to 9% of the main peak in the Fourier spectrum.

To get an impression of the accuracy of the eigenfunctions, I extracted the H_1-mode in 1D with a medium and a very high resolution. The results are shown in Fig. 4.2. Obviously a good convergence is already reached for the resolution of 47 points. Only at the last point inside the star, the velocity eigenfunction shows a significant drop for the medium resolution. This is a typical feature of the surface treatment scheme, which is also present for the other modes. Note this only means the surface velocities are damped in time average. As shown in Fig. 3.20, also increased surface velocities occur during the oscillation.

The radial F-, H_2-, and H_3-mode eigenfunctions of model N are shown in Fig. 4.3 to Fig. 4.5. The nodes are closer together for the higher modes, and the gradients near the troublesome stellar surface are steeper. Because of this, the simulation of even higher order modes would become numerically very demanding with a general purpose code.

As stated in [59, 53], perturbative calculations for nonrotating star models predict that the eigenfunctions separate into an angular and a radial part. In detail, the density eigenfunction $\delta\rho$ can be written as

$$\delta\rho(r,\vartheta) = \delta\rho(r) Y_l^0(\vartheta) \tag{4.1}$$

This provides us with an additional check for the accuracy of nonradial modes: The cuts of the eigenfunctions along $\vartheta = 0$ and $\vartheta = \pi/2$ should be identical up to a factor $f = Y_l^0(0)/Y_l^0(\pi/2)$. For the $l = 2$ modes, $f = -2$.

The cuts for the ${}^2f$-, ${}^2p_1$-, and ${}^2p_2$-modes of model N are shown in Fig. 4.6, 4.8, and 4.10. Indeed, the eigenfunctions are compatible with Eq. (4.1) to a good accuracy. Not surprisingly, the deviations are smaller for the lower order modes.

The 2D-eigenfunctions of the nonradial modes of model N are shown in Fig. 4.7, 4.9, and 4.11. The location of the density eigenfunction nodes cannot be determined very accurately near the origin and the crossing of nodes, since the gradient of the real eigenfunction is zero at these points and the slightest error in the eigenfunction can significantly shift the node position. Apart from this inaccuracy, the nodes behave like expected from Eq. (4.1).

The eigenfunctions for the rotating model R are shown in Fig. 4.13 to Fig. 4.25. In the rotating case, no separation into angular and radial functions is expected. For the rotation rate of model R, such a separation is not even approximately given, as one can see from the location of the nodes. Another difference to the nonrotating case is that the higher order quasiradial modes are concentrated towards the rotational axis, and that the velocity fields are mainly parallel to the axis.

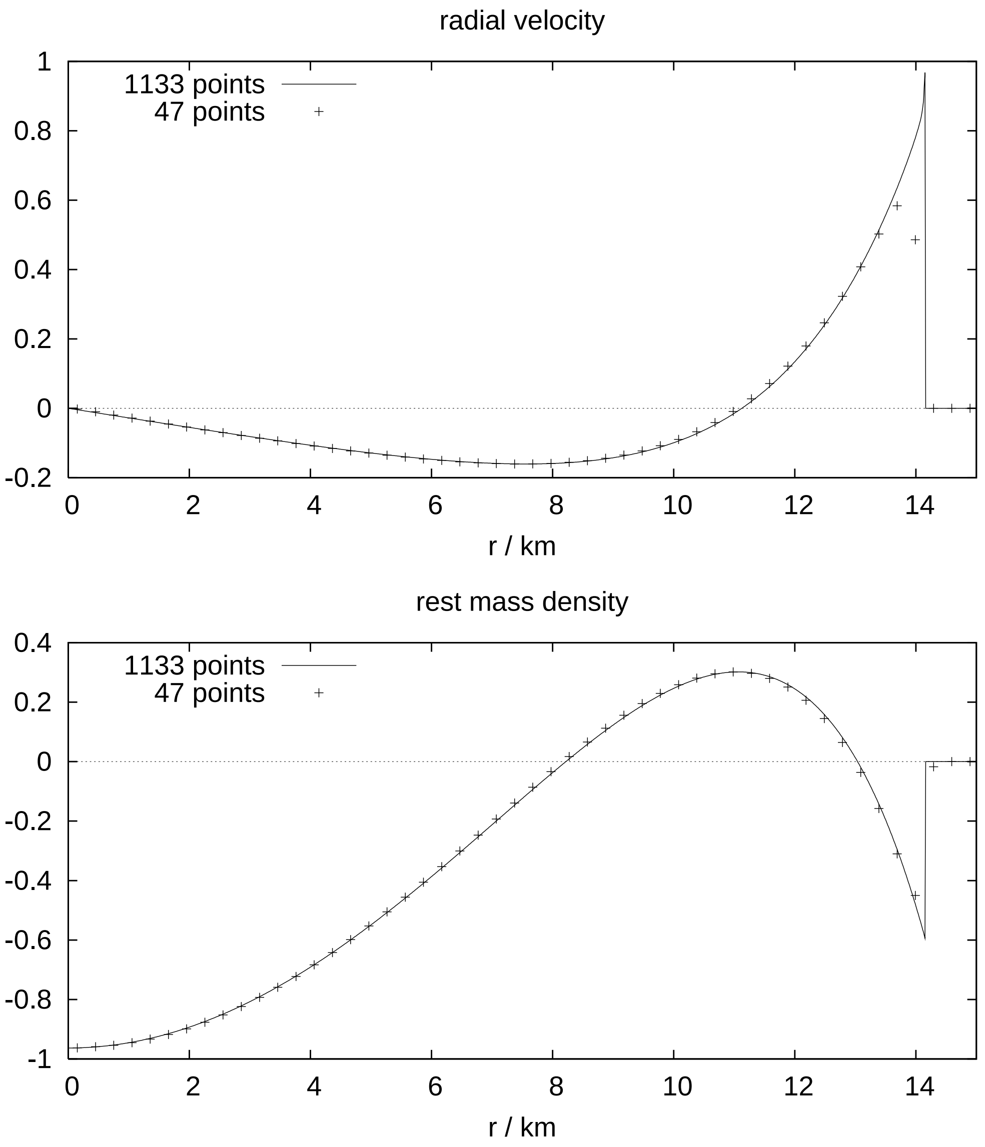


Figure 4.2: Convergence of extracted eigenfunctions on the example of the radial H_1-mode, obtained from a 1D simulation. The eigenfunctions are plotted in arbitrary units, r is the circumferential radius. Evolution time was 20 ms.

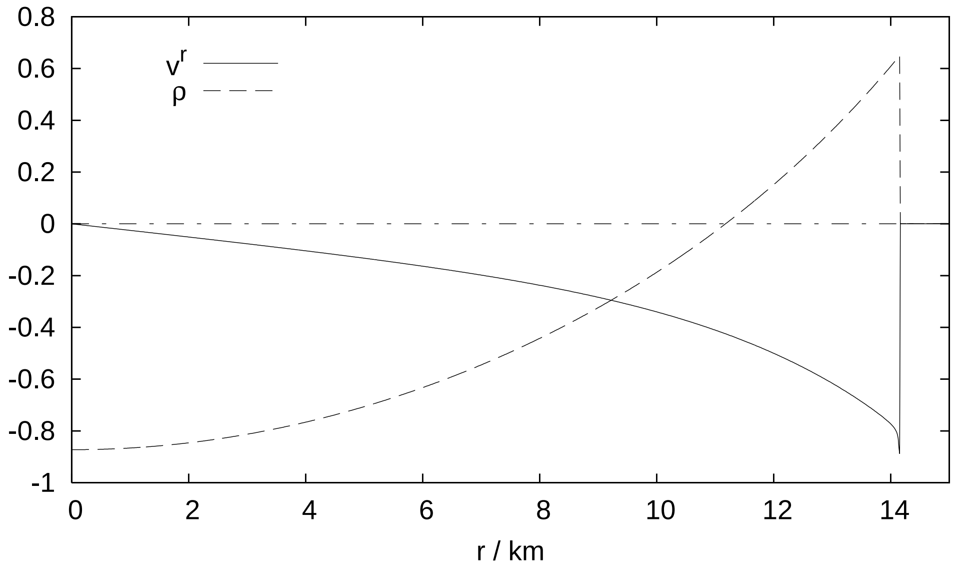


Figure 4.3: Velocity and density Eigenfunctions of the F-mode, obtained from a 1D simulation with a resolution of 1133 points and 10 ms evolution time. The eigenfunctions are plotted in arbitrary units, r is the circumferential radius.

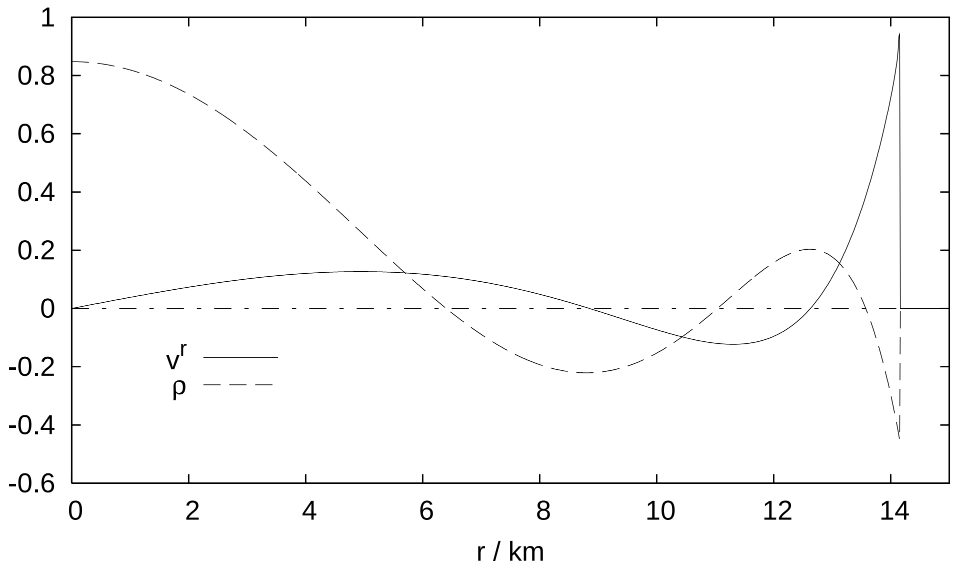


Figure 4.4: Like Fig. 4.3, but showing the H_2-mode.

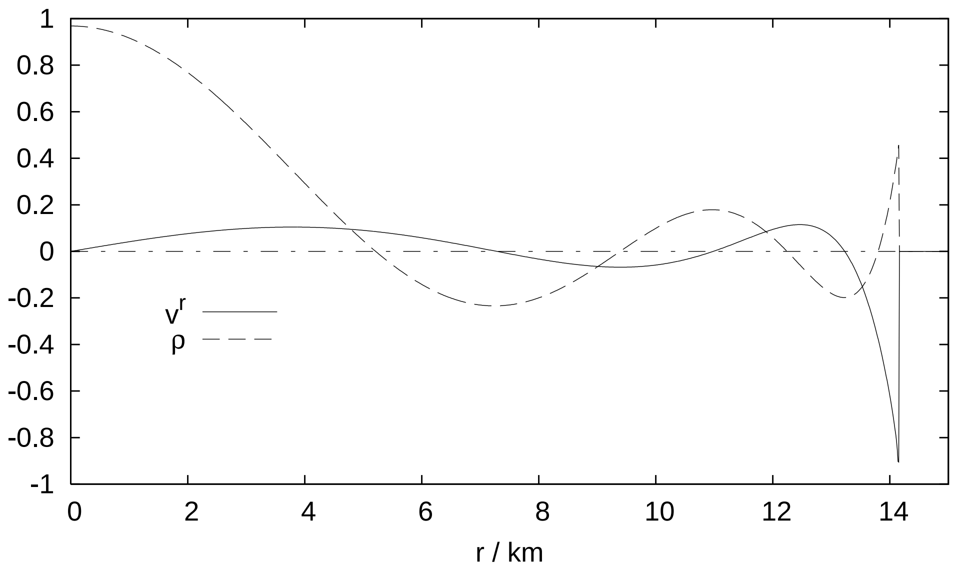


Figure 4.5: Like Fig. 4.3, but showing the H_3-mode.

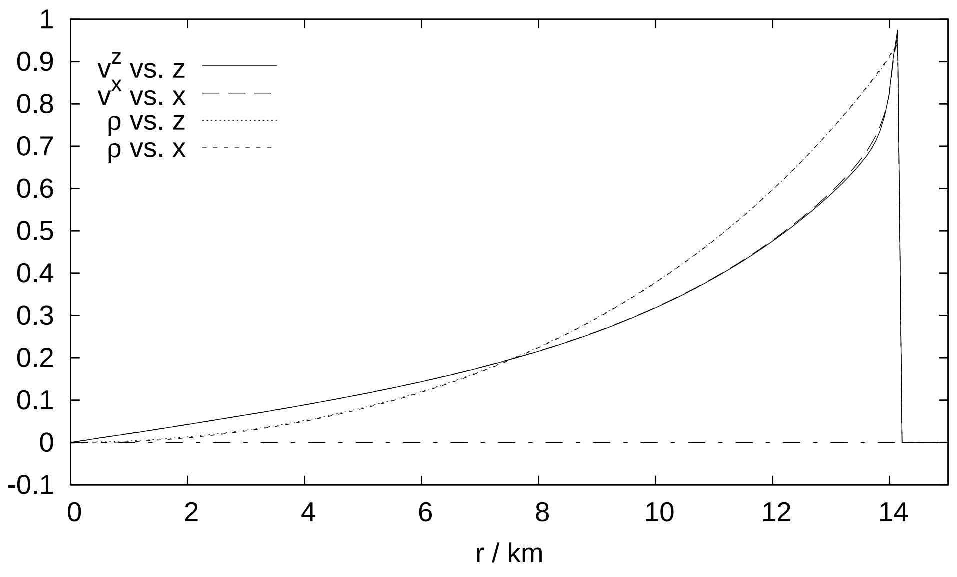


Figure 4.6: Velocity and density eigenfunction of the ${}^{2}f$-mode of model N, plotted along the x- and z-axis in arbitrary units. The cuts along the x-axis are scaled by a factor of -2 with respect to the ones along the z-axis. r is the circumferential radius. The results are obtained with a resolution of 189^2 in 2D.

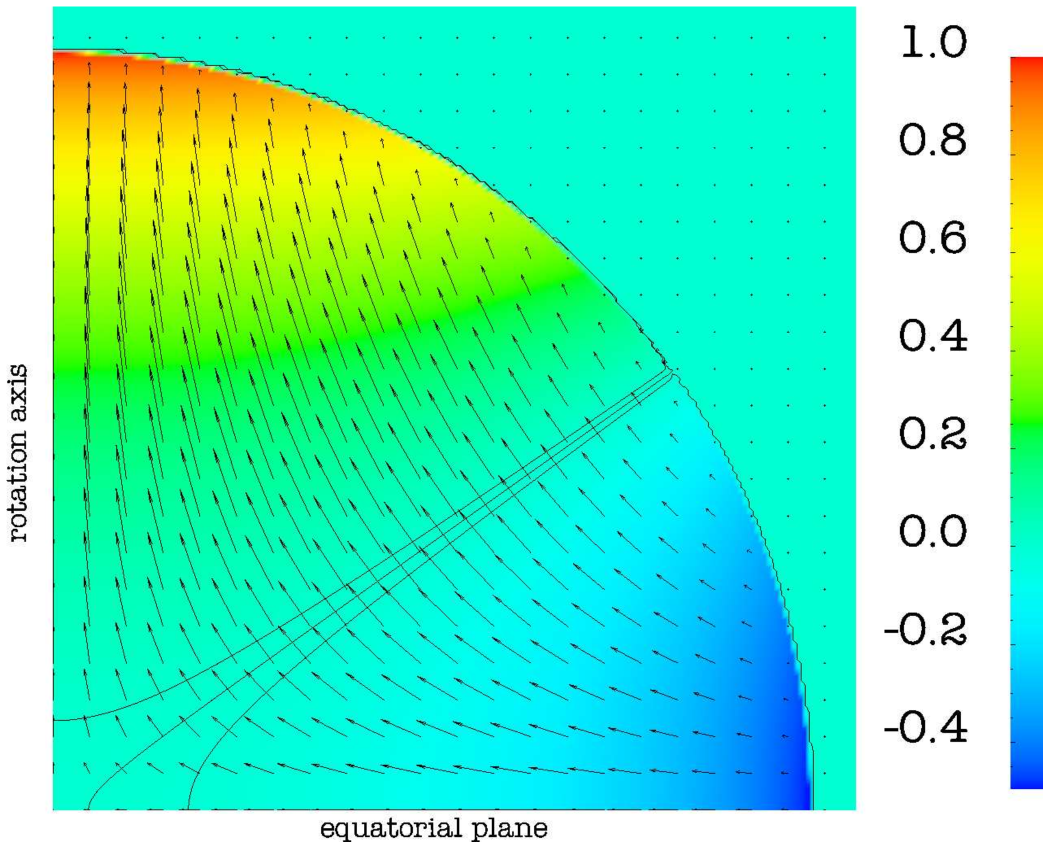


Figure 4.7: Eigenfunctions of the ${}^2f$-mode of model N, computed with a resolution of 189^2 in 2D. Colour coded (in arbitrary units) is the eigenfunction of the rest mass density change. The solid line in the middle marks the node of the density eigenfunction, the other two lines correspond to $\pm 1\%$ of the maximum norm of the eigenfunction. The distance of these lines from the node correlates with the accuracy of the extracted node location. The arrows show the velocity eigenfunction times the (equilibrium) rest mass density.

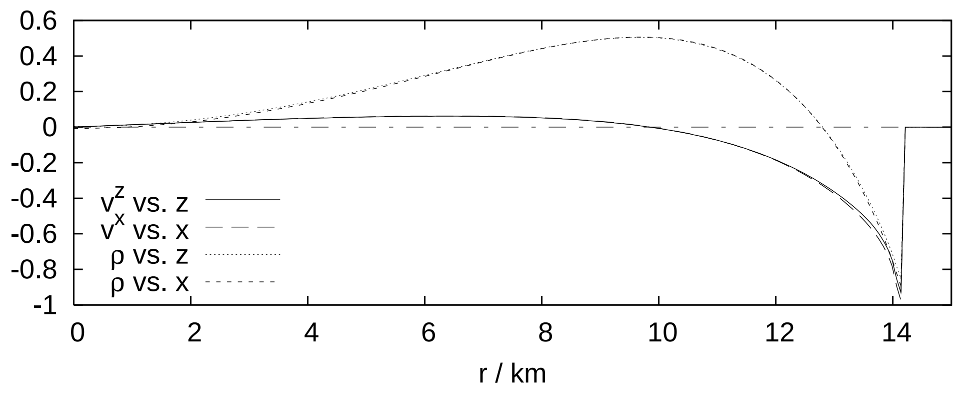


Figure 4.8: Like Fig. 4.6, but displaying the ${}^2p_1$-mode.

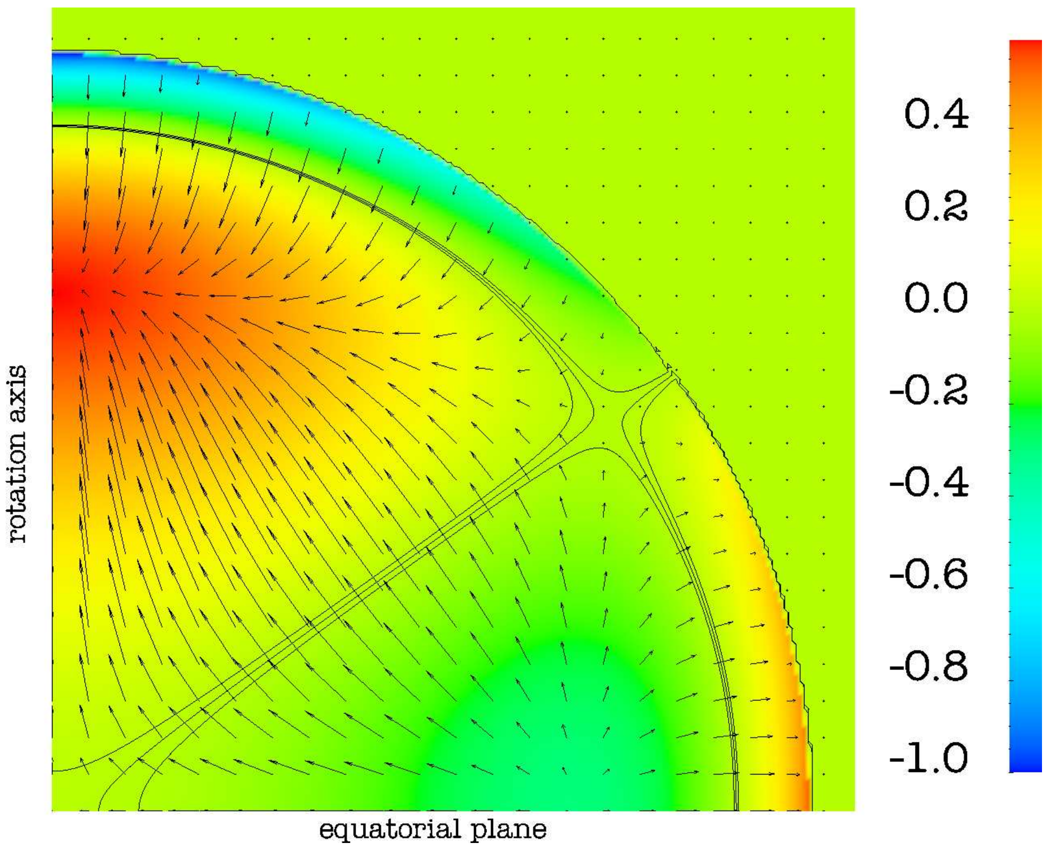


Figure 4.9: Like Fig. 4.7, but displaying the ${}^2p_1$-mode.

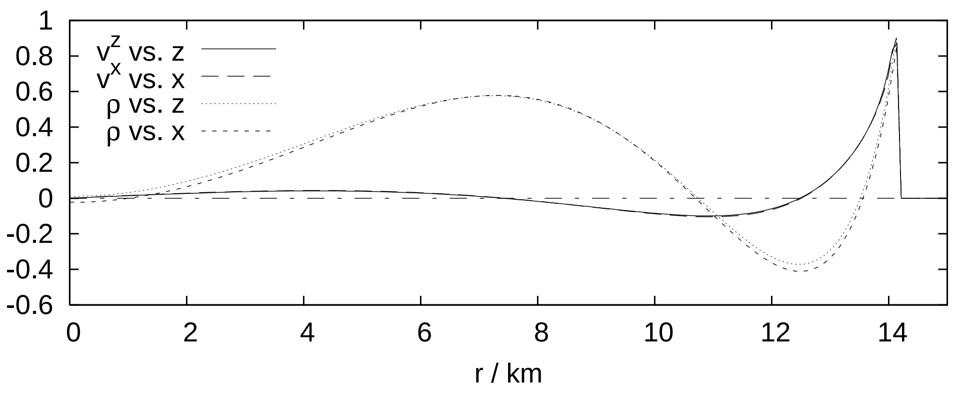


Figure 4.10: Like Fig. 4.6, but displaying the ${}^2p_2$-mode.

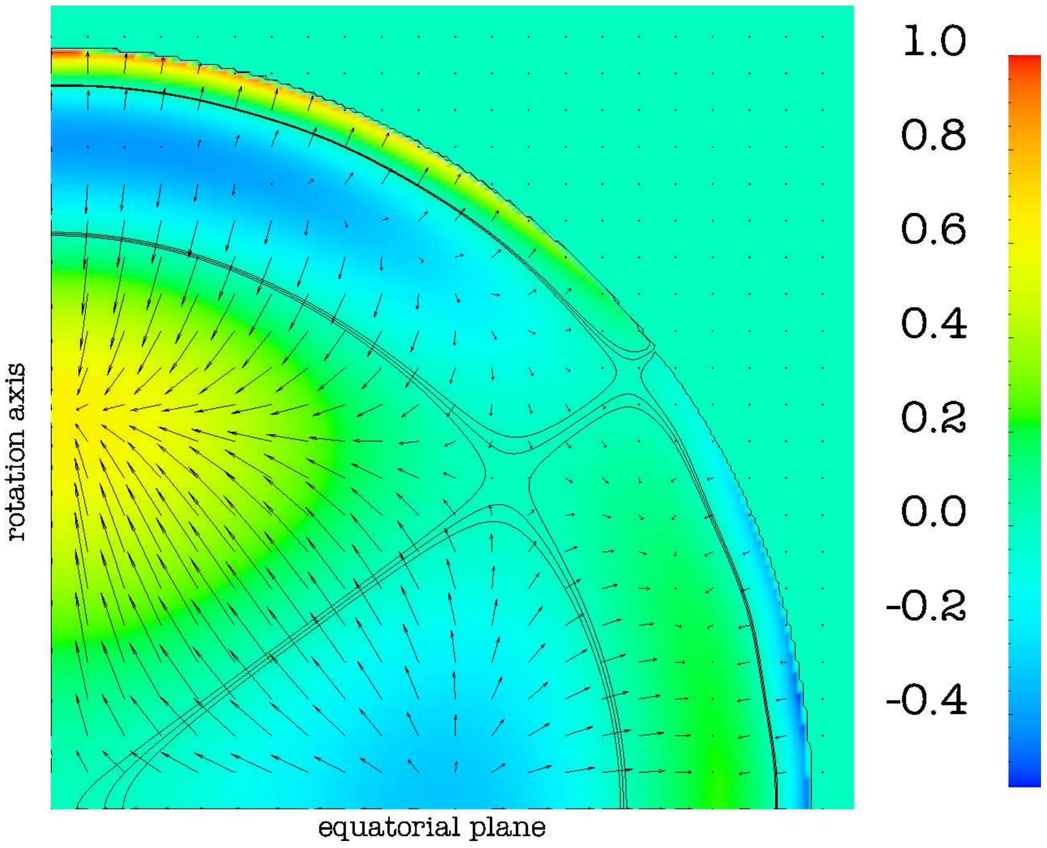


Figure 4.11: Like Fig. 4.7, but displaying the ${}^2p_2$-mode.

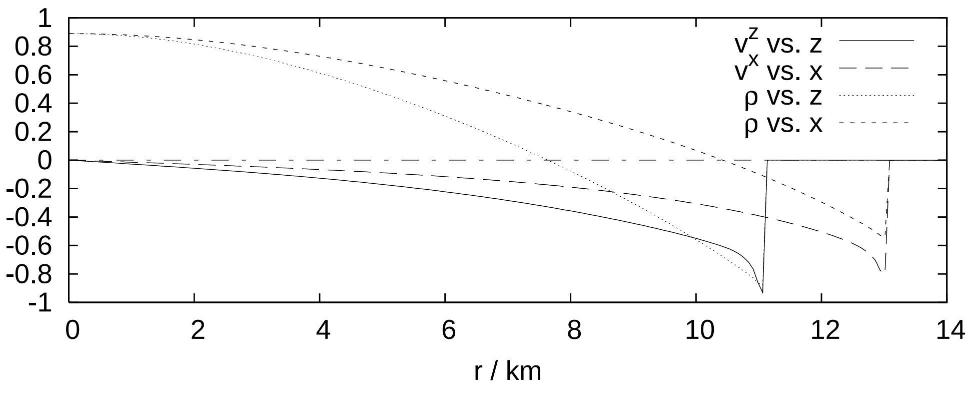


Figure 4.12: Like Fig. 4.6, but depicting the F-mode of model R. Also, the cuts along the x- and z-axis are scaled the same.

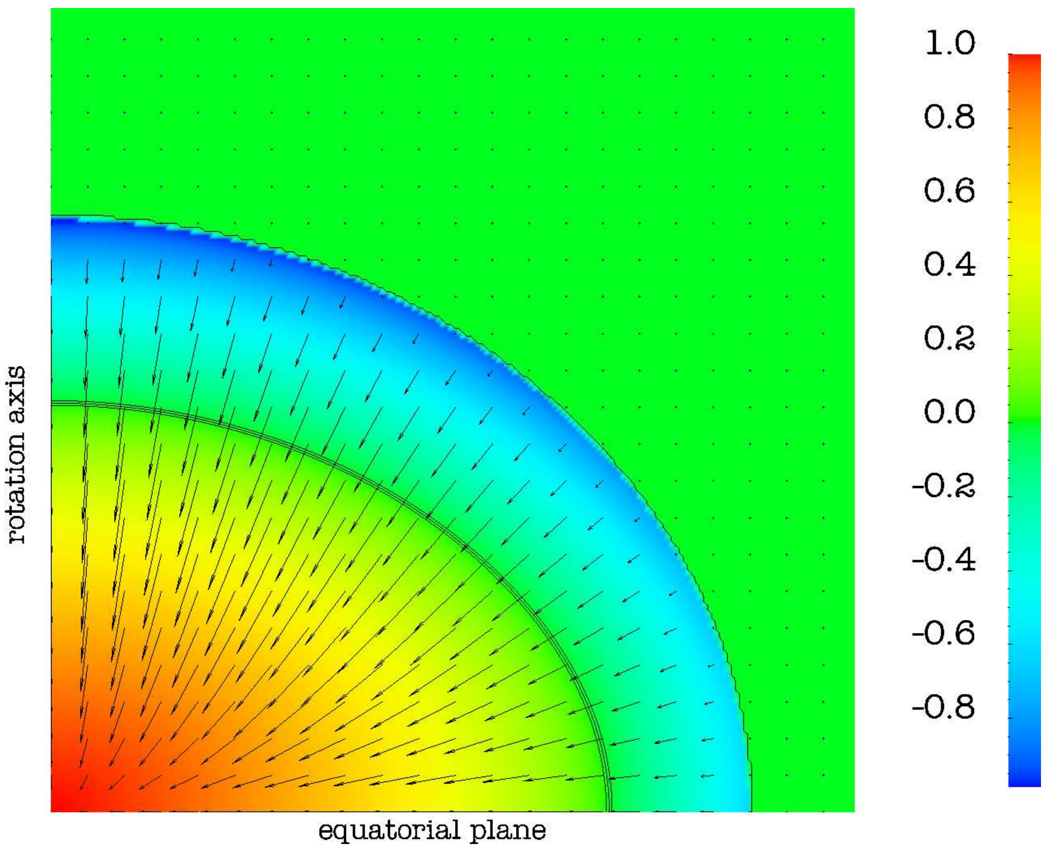


Figure 4.13: Like Fig. 4.7, but displaying the F-mode of the rotating model R. The resolution is $R_p/\Delta x \approx 149$, $R_e/\Delta x \approx 175$, where R_p and R_e are the star's polar and equatorial coordinate radius and Δx is the grid spacing.

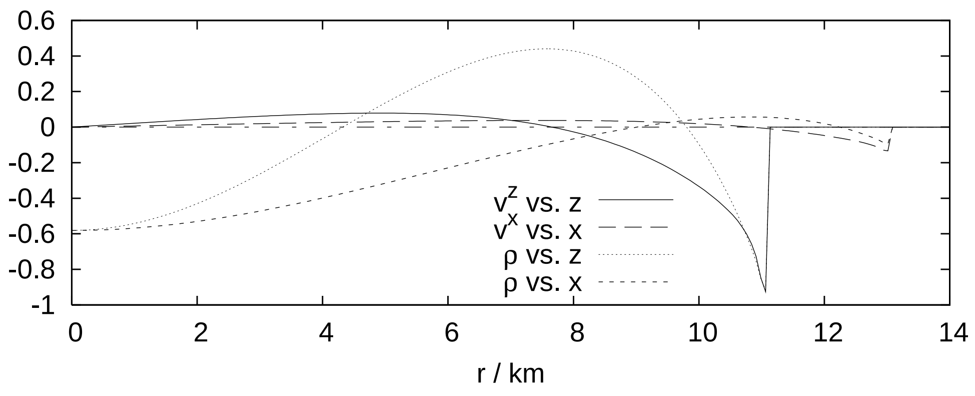


Figure 4.14: Like Fig. 4.12, but displaying the H_1-mode.

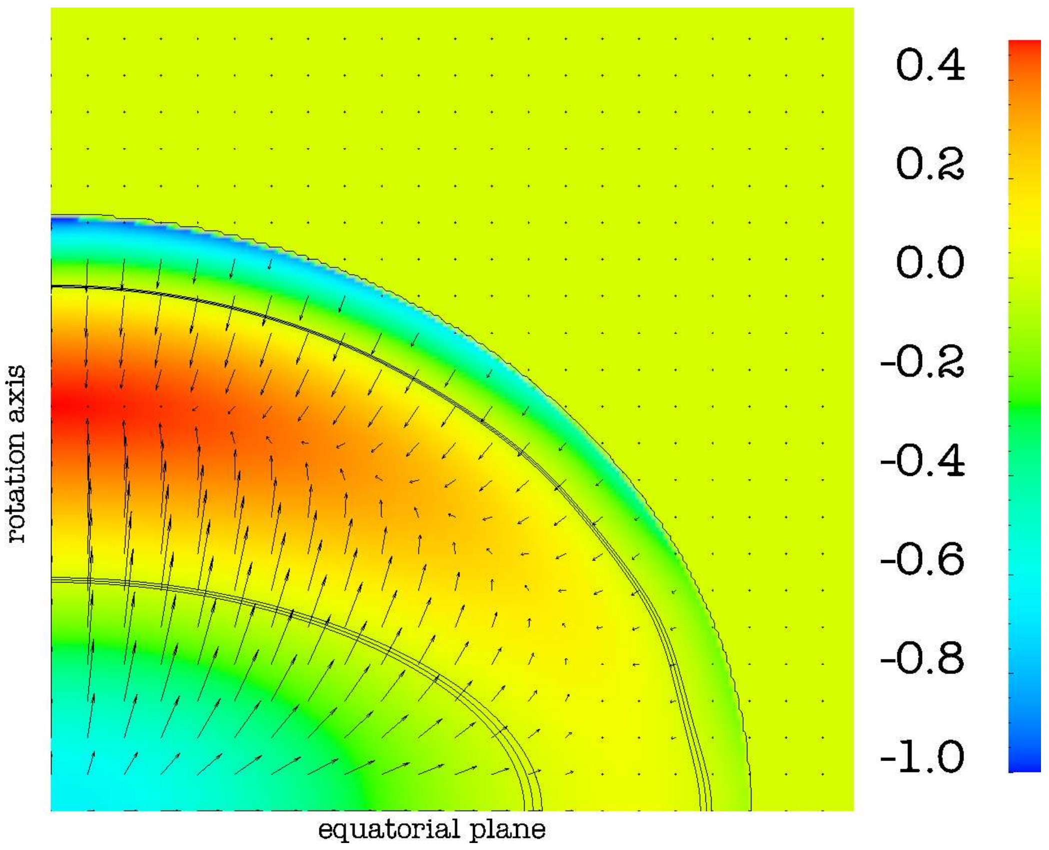


Figure 4.15: Like Fig. 4.13, but displaying the H_1-mode.

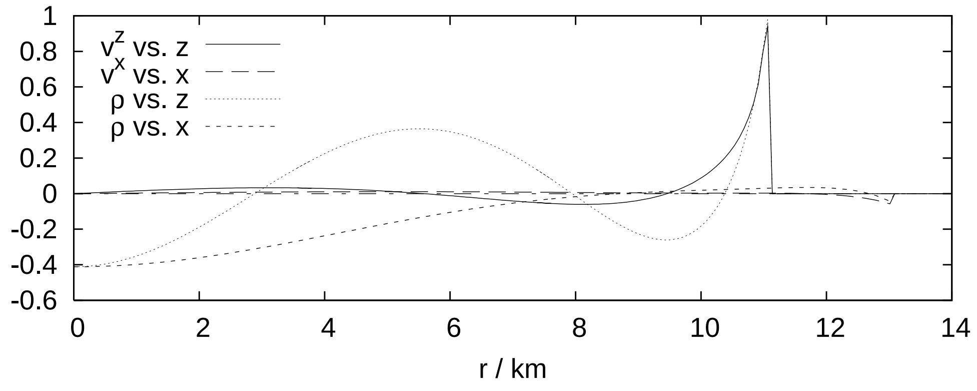


Figure 4.16: Like Fig. 4.12, but displaying the H_2-mode.

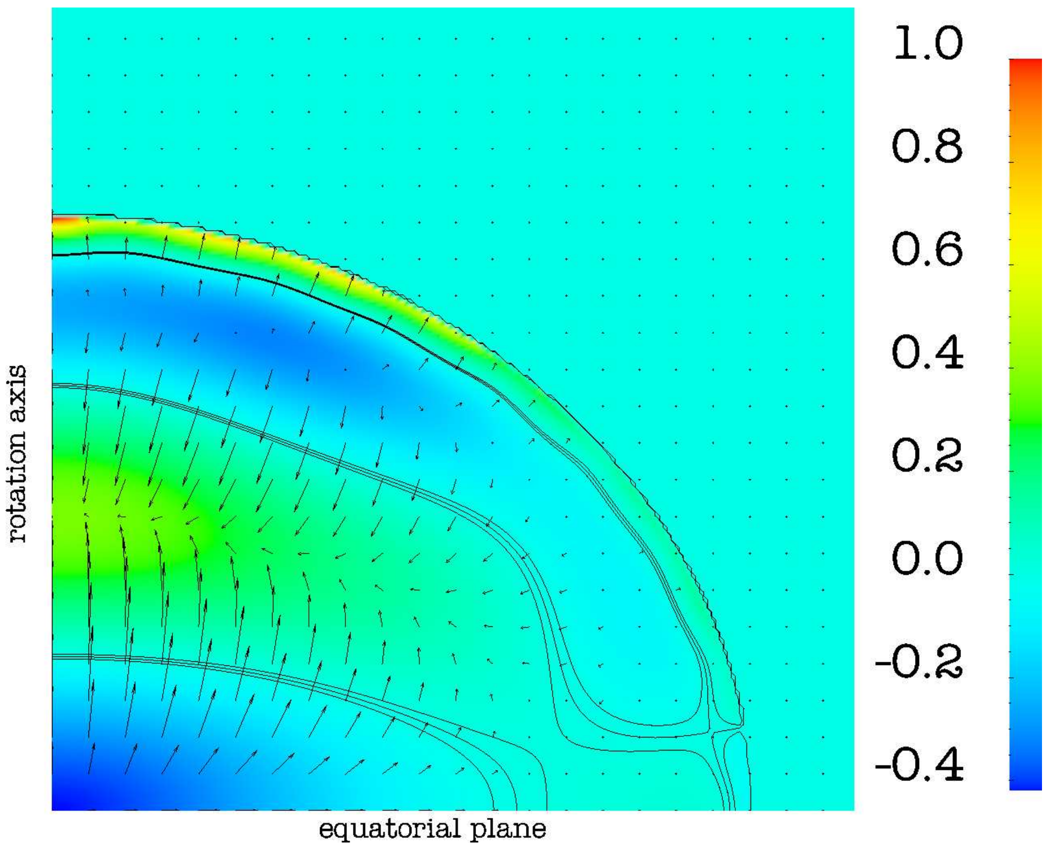


Figure 4.17: Like Fig. 4.13, but displaying the H_2-mode. The location of the nodes in the lower right part is highly inaccurate since the eigenfunctions almost vanish in this region.

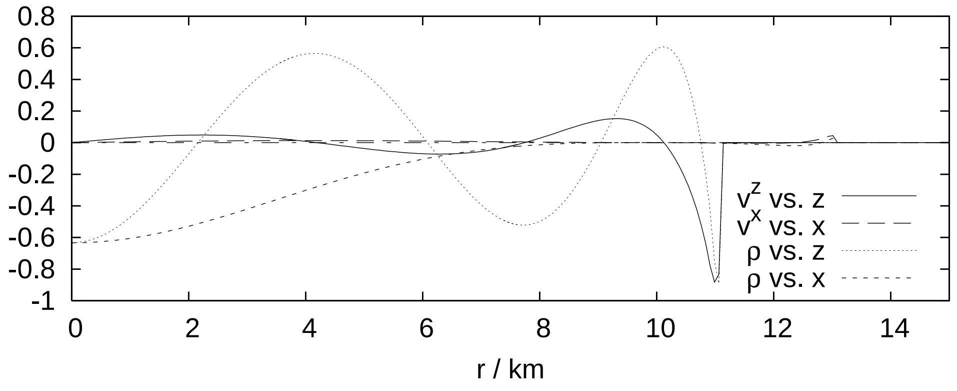


Figure 4.18: Like Fig. 4.12, but displaying the H_3-mode.

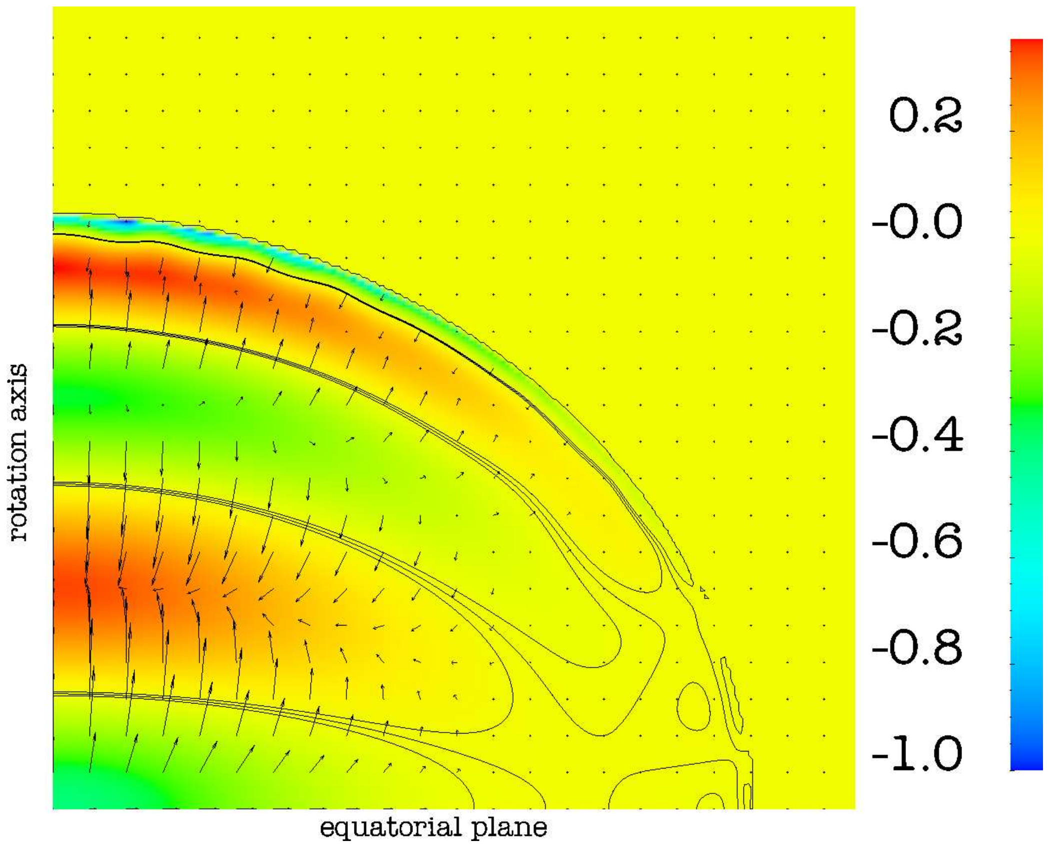


Figure 4.19: Like Fig. 4.13, but displaying the H_3-mode. The eigenfunctions have very small amplitude in the lower right part and the extracted node location in this region is meaningless. The waviness of the outermost node is most probably an artifact due to the presence of other modes in the evolution.

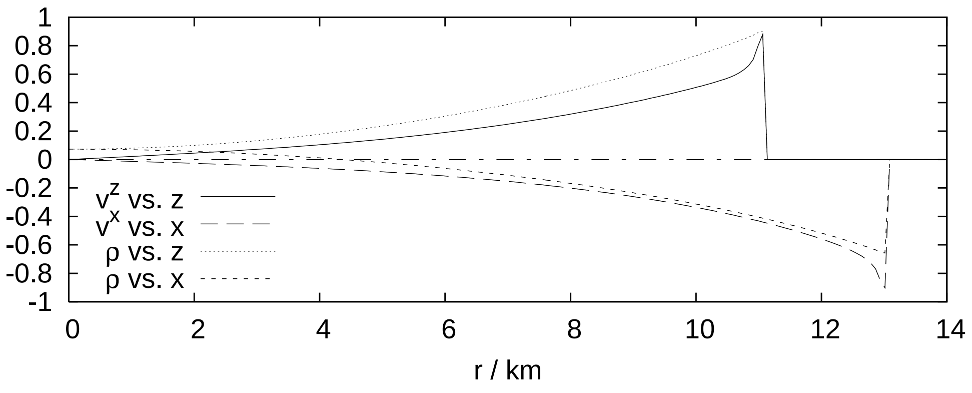


Figure 4.20: Like Fig. 4.12, but displaying the ${}^2f$-mode.

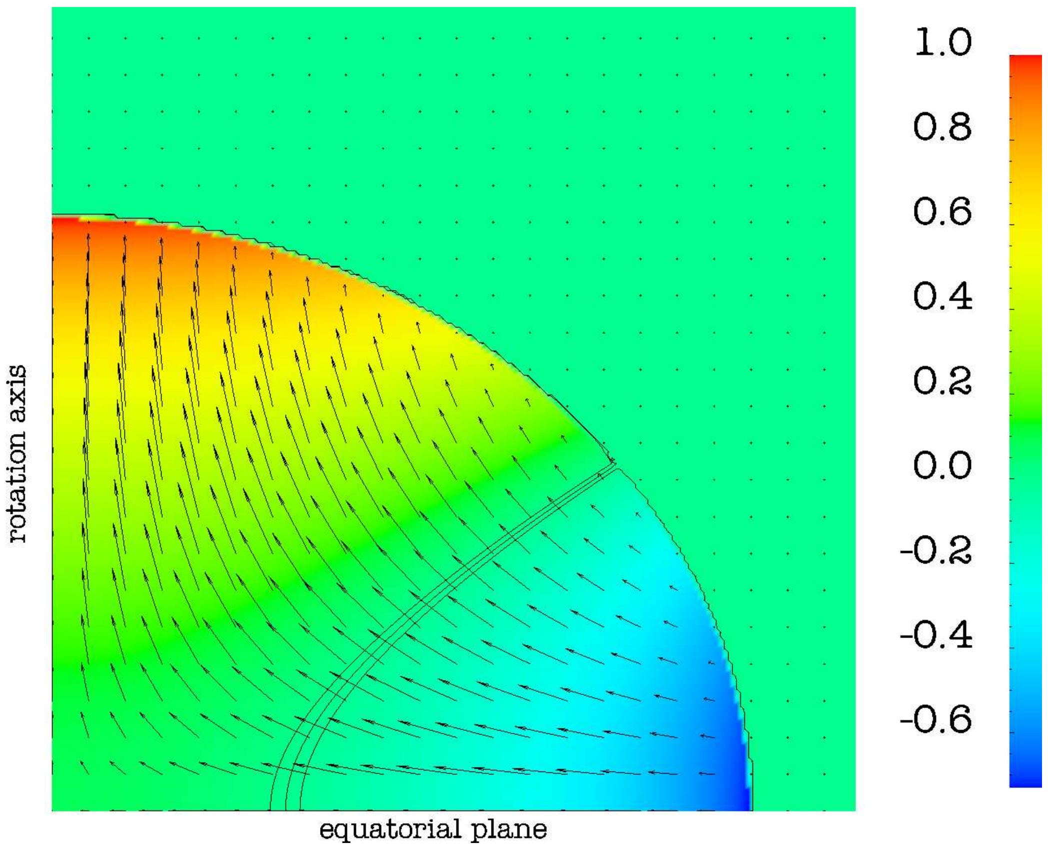


Figure 4.21: Like Fig. 4.13, but displaying the ${}^2f$-mode.

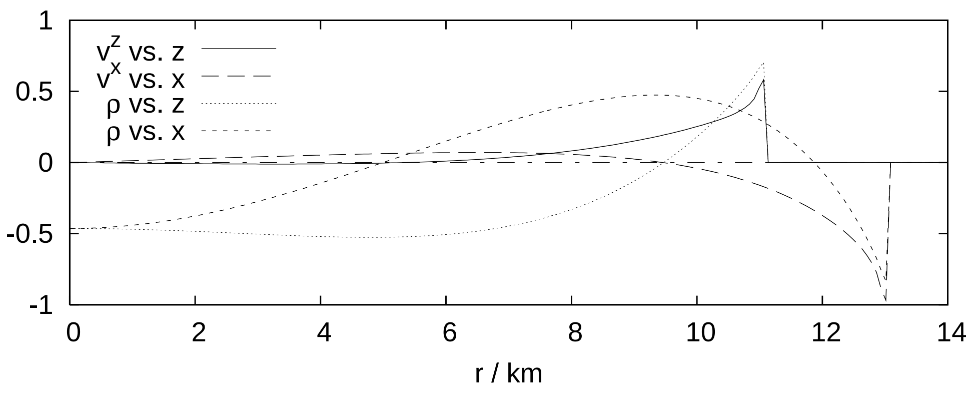


Figure 4.22: Like Fig. 4.12, but displaying the ${}^2p_1$-mode.

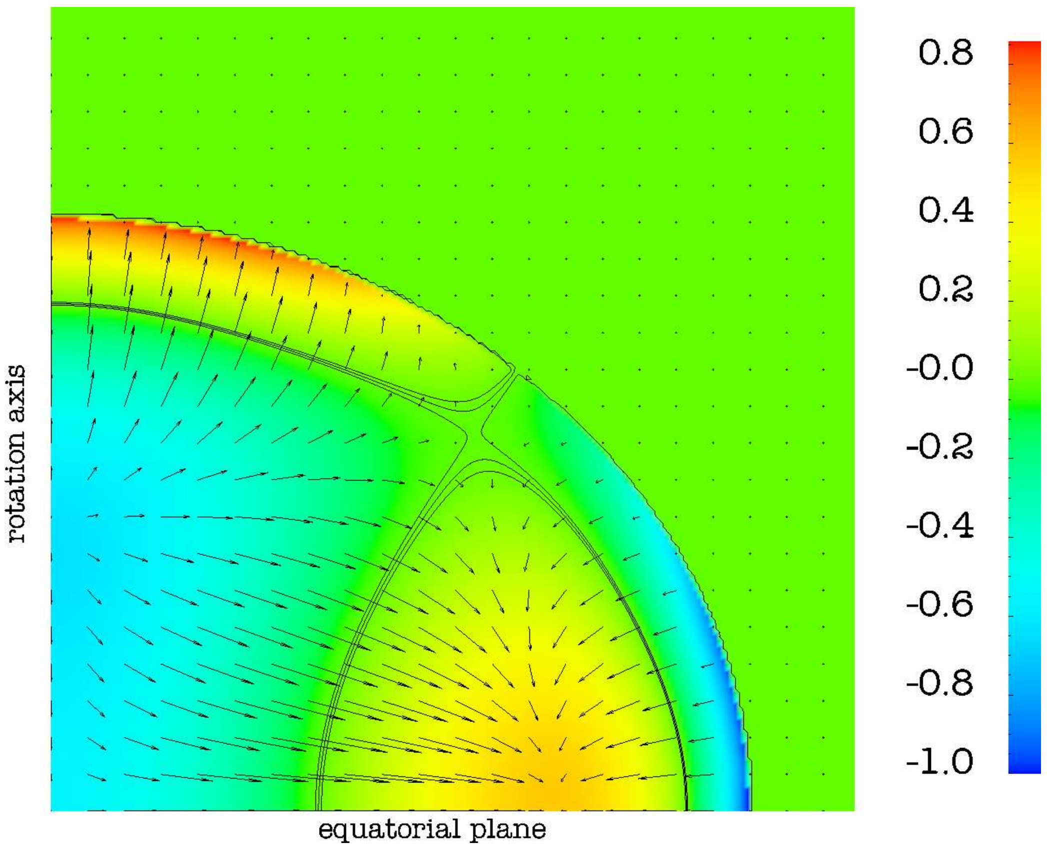


Figure 4.23: Like Fig. 4.23, but displaying the ${}^2p_1$-mode.

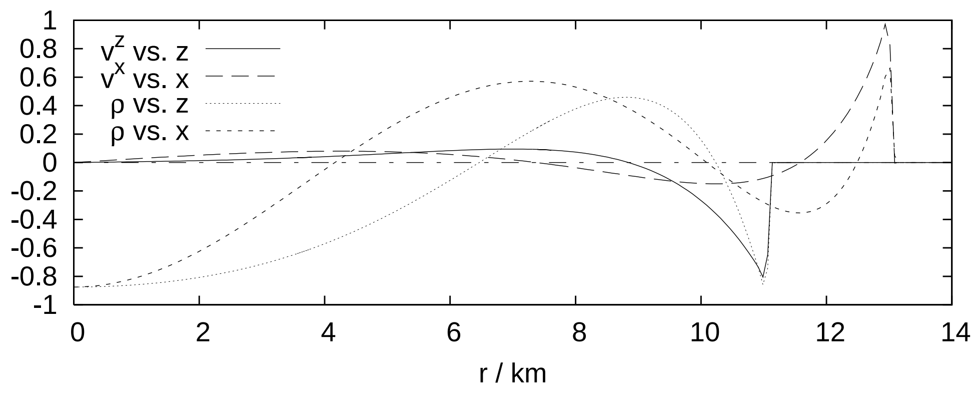


Figure 4.24: Like Fig. 4.12, but displaying the ${}^2p_2$-mode.

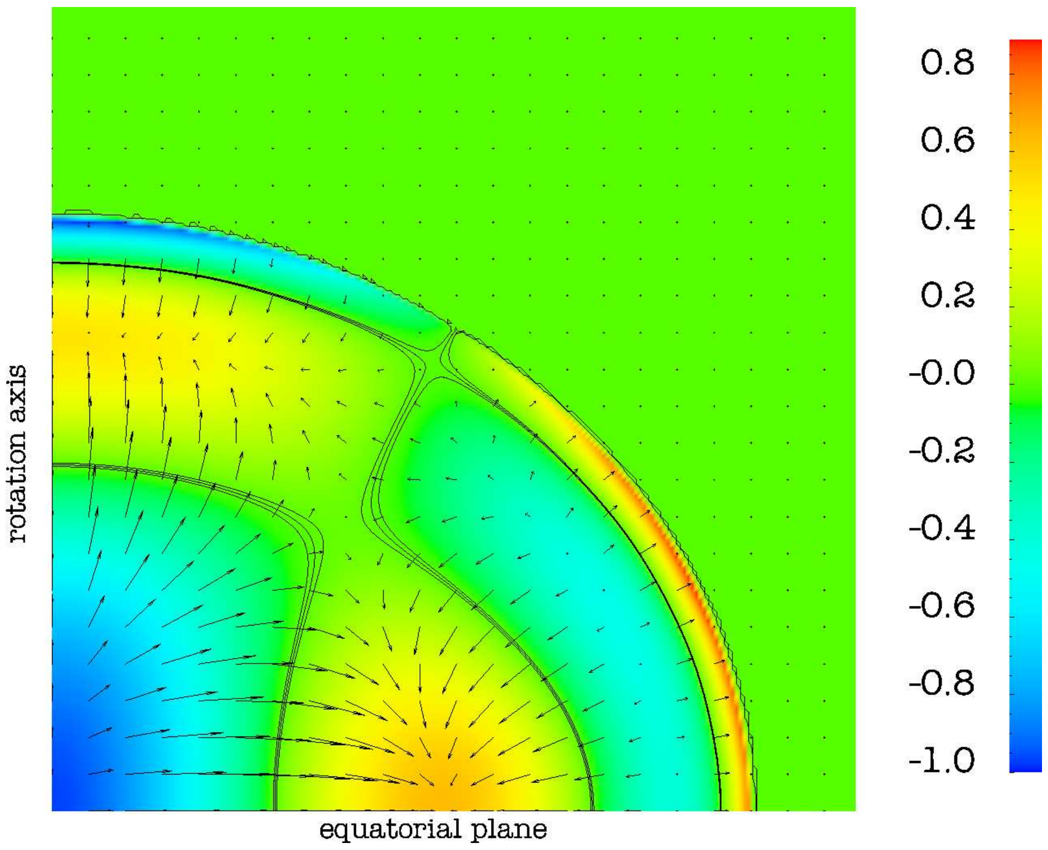


Figure 4.25: Like Fig. 4.13, but displaying the ${}^2p_2$-mode.

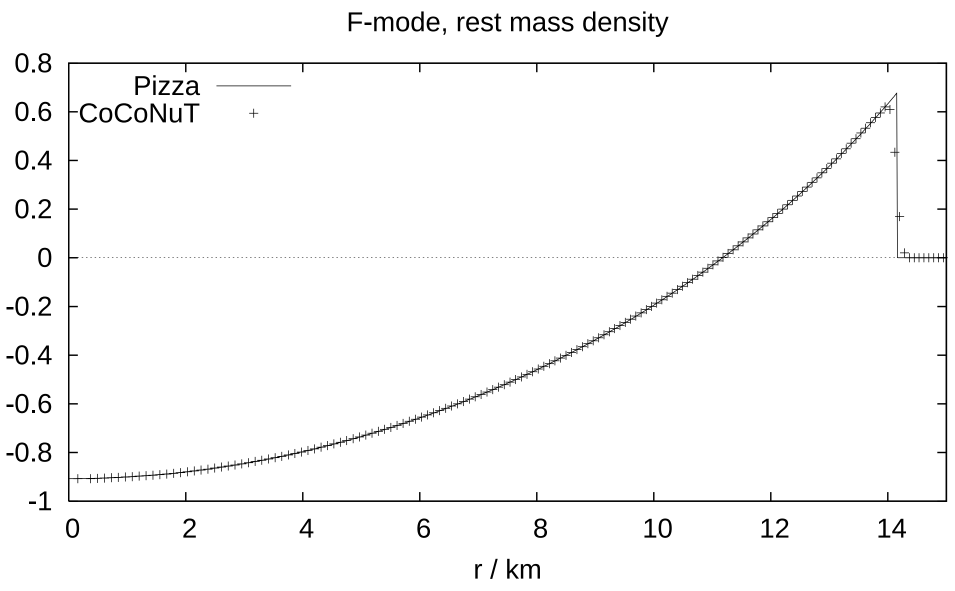
For the F-, H_1-, ${}^2f$-, and ${}^2p_1$-modes of model N, eigenfunctions computed with the `CoCoNuT` code have been provided by [13]. A comparison to the results with `Pizza` is shown in Fig. 4.26 to Fig. 4.29. For the F-, H_1-, and ${}^2p_1$-modes, the eigenfunctions agree very well. Only near the surface, the velocity eigenfunctions obtained with the `CoCoNuT` code expose overshoots in contrast to the `Pizza` code. This is due to the distinctive features of the surface treatment described in Sec. 2.2.5 and the artificial atmosphere method used by the `CoCoNuT` code: The former tends to damp velocities near the surface, while the latter generates a lot of noise at the surface (which does not mean it does not damp the oscillations).

While the bulk properties of the ${}^2f$-mode eigenfunctions agree reasonable well, there is a significant deviation near the centre of the star. The reason is unclear, but one reason might be that the `CoCoNuT` code is using spherical coordinates, which often induce problems at the origin due to the degeneration of the equations at $r = 0$.

As mentioned, the density eigenfunctions for nonrotating stars separate into a radial and an angular part. Due to regularity conditions, the radial part should go to zero at the origin if the angular part is not constant, i.e. for nonradial modes. This indicates that the ${}^2f$-mode density eigenfunction obtained by the `Pizza` code is more realistic. It does not go to zero as well, but the amplitude at the origin is only $7 \cdot 10^{-4}$ of the maximum amplitude.

Although the eigenfunctions shown here are computed within the Cowling approximation, they already provide a qualitative picture of the oscillations. An exception is the ${}^1f$-mode (not shown here), which corresponds to a star oscillating up and down as a whole in the fixed gravitational potential. Quantitatively, the Cowling approximation is very bad, since the frequencies with evolved spacetime already differ by factors up to 2, as will be shown in the next section. A comprehensive comparison for a whole range of rotation rates between the case with and without the Cowling approximation can be found in [17].

To study linear oscillations of nonrotating stars, it is surely more effective to use a perturbative approach and decompose the perturbations into spherical harmonics, like it is done for example in [42, 29, 31, 5]. From the eigenfunctions shown in this section, it is obvious that this approach works well only for nonrotating or slowly rotating stars, since the angular dependency in the rotating case does not correspond to a single spherical harmonic any more. On the other hand, it is computationally less demanding to incorporate the gravitational field in the perturbative framework.



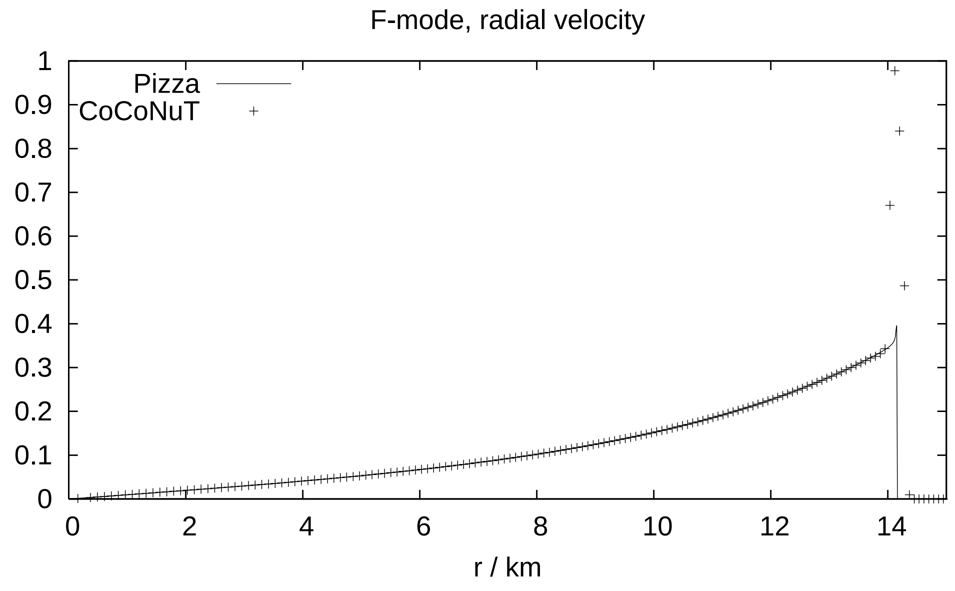


Figure 4.26: Comparison of the radial F-mode eigenfunctions of model N, as obtained with `Pizza` and `CoCoNuT` codes. The eigenfunctions are plotted in arbitrary units along the equatorial plane. The velocities are the physical ones, and r is the circumferential radius. The resolution is 1133 points in 1D for the `Pizza` simulation, and 140×60 points in (r, ϑ) for the `CoCoNuT` simulation.

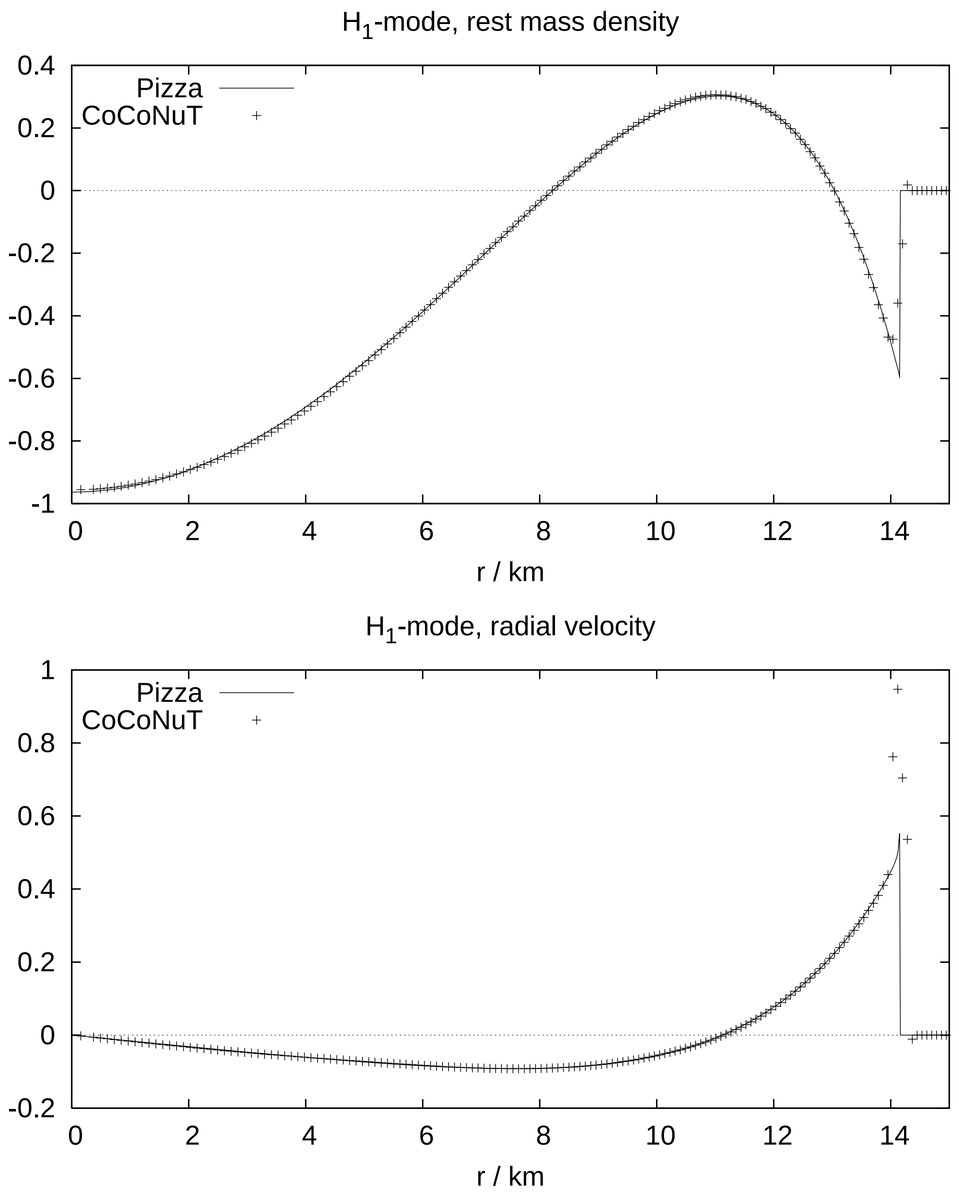


Figure 4.27: Like Fig. 4.26 but for the H_1-mode eigenfunctions.

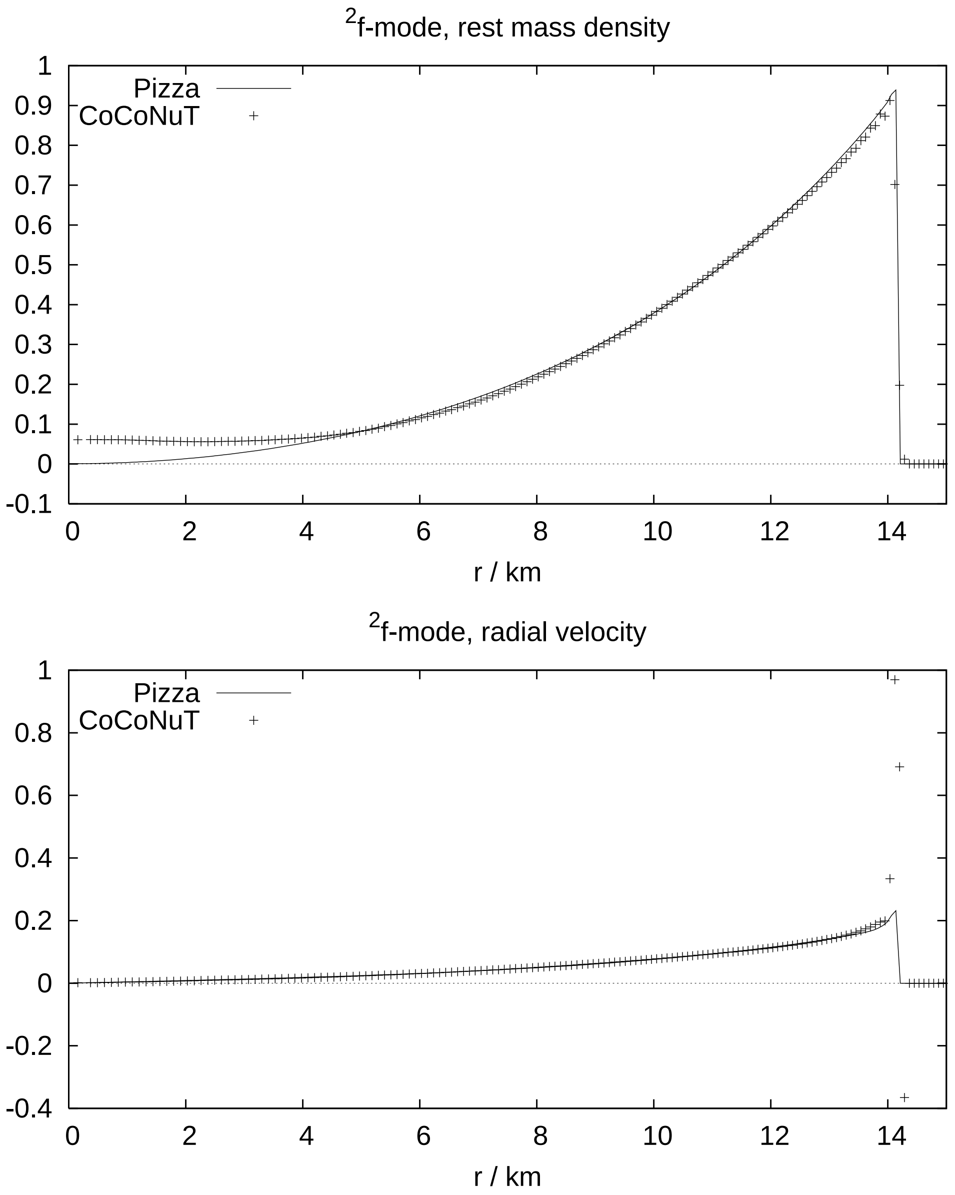


Figure 4.28: Like Fig. 4.26, but showing the ^2f-mode eigenfunctions. The resolution of the `Pizza` result is 189^2 points in 2D.

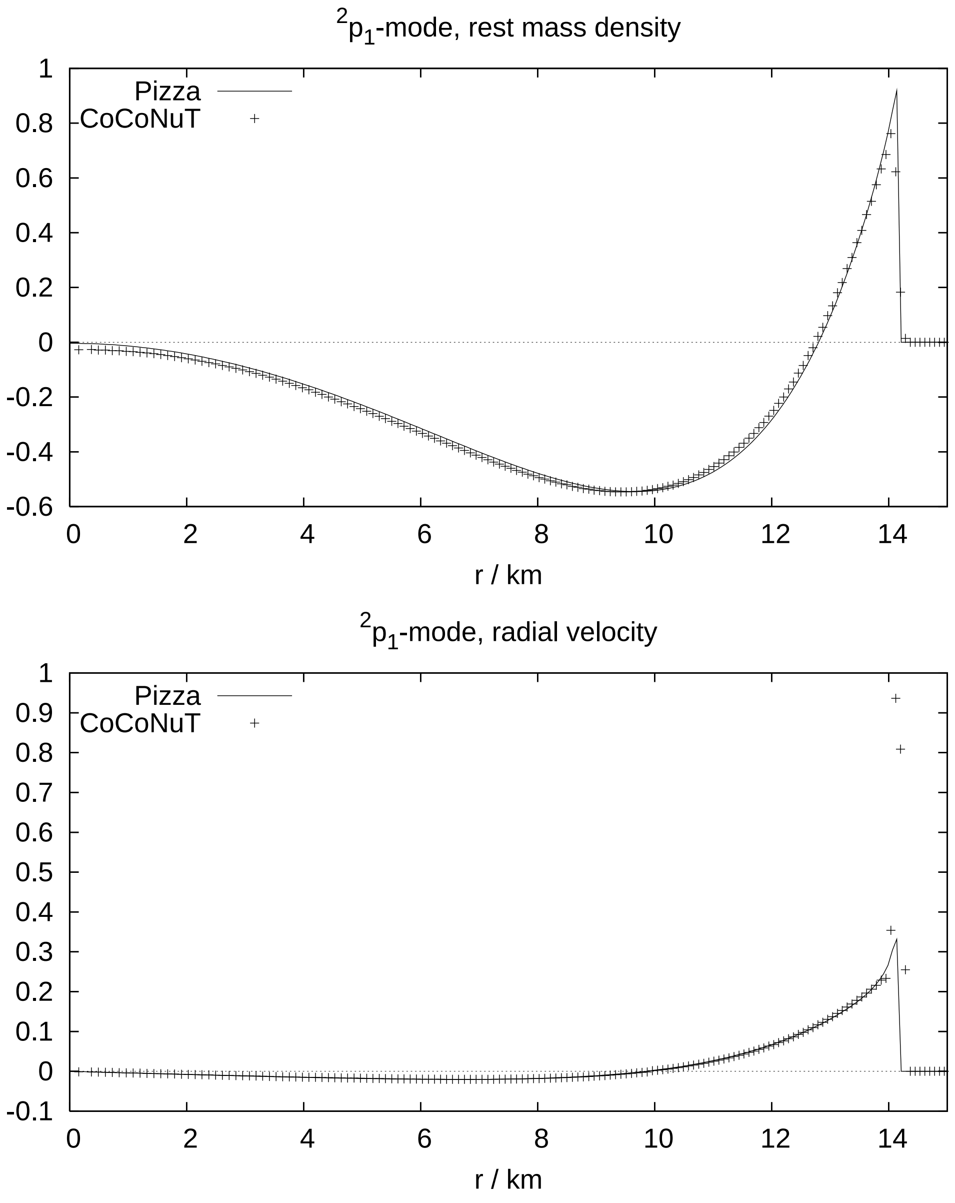


Figure 4.29: Like Fig. 4.28, but displaying the $^2p_1$-mode eigenfunctions.

4.2 Coupled spacetime evolution

In this section, I present first results with coupled spacetime evolution. These are preliminary results, which only demonstrate that the coupling is technically working, and leads to a stable evolution; there are still some open questions. To evolve the spacetime, I use a module available for the `Cactus` computational toolkit called "`BSSN_MoL`". The coupling to the hydrodynamic part is described in Sec. 2.3.1. Any other code which accepts the stress energy tensor as input and provides the metric as output could be used instead. It is beyond the scope of this work to describe how the spacetime evolution code works internally, see [4] for details.

For simulations with coupled spacetime evolution, there are some important practical differences to simulations using the Cowling approximation. First, there are no proper outer boundary conditions for the spacetime evolution equations available. Therefore, the outer boundary has to be pushed as far out as possible in order to minimise its influence. The only practical way to do this in Cartesian coordinates is to use mesh refinement.

Second, there currently exists no genuine twodimensional spacetime evolution code for the `Cactus` framework, so all simulations have to be performed in 3D, even for the study of axisymmetric oscillations.

The third complication arises from the fact that even for a stationary spacetime, the coordinate system can change, depending on the gauge conditions. Ideally, the whole analysis should be based upon coordinate independent quantities like physical distances and proper times. Such analysis is a nontrivial task; for the tests presented here, coordinate based measures will be used. However, this is not a severe restriction since the coordinate system remains quite stationary in practice. Nevertheless, one has to be careful with the interpretation of frequencies measured with respect to coordinate times, since the lapse function is not fixed anymore.

As a first test of the coupling to the spacetime evolution, I evolved model N with the polytropic EOS for 40 ms. For this, fixed mesh refinement was used. The mesh refinement is implemented by the `Carpet` code [43], which is a part of the `Cactus` toolkit. For a description of the method see [44]; I only note that it is designed in a way which requires no changes to the evolution code. For the simulation, the star is covered by an inner grid with a resolution of 47^3 grid points per stellar radius. The inner grid is covered by a second grid, which is twice as big and twice as coarse.

As gauge condition for the lapse function, a variant of the "1+log" slicing using the K-driver method is applied, which is described in [4, 9]. This condition is an approximation for maximal slicing ($K = 0$); it was constructed to provide a numerically stable slicing condition. In practice, it results in an evolution equation for the lapse. The shift vector is simply set to zero. As outer boundary conditions for the spacetime, Sommerfeld (radiation) boundary conditions are applied. Both the gauge conditions and the spacetime boundary conditions are set by the `BSSN_MoL` code.

As shown in Fig. 4.30, the star immediately starts oscillating, with a small amplitude. This is to be expected since in contrast to the hydro code, the spacetime evolution code uses a standard finite difference scheme, which cannot exactly preserve the physical equilibrium state.

Although the oscillation is mainly radial, it contains a small nonradial component, whose magnitude is around 1% compared to the radial one. This is the case even on the space diagonal, where it should vanish completely because the initial data, the numerical grid and the numerical scheme are invariant under coordinate exchange. All 3D simulations performed for model N with the `Pizza` code in Cowling approximation did respect that symmetry. This indicates that there might be a coding error either in the spacetime evolution code, or in the mesh refinement code, or in the interaction between the two and `Pizza`, which breaks the symmetry.

As visible in the Fourier spectra for the velocities shown in Fig. 4.31, the oscillation is dominated by two radial modes. In the spectrum for the ϑ-velocity, two different peaks corresponding to nonradial modes occur. These can only be seen because the ϑ-velocity is zero for the dominant radial oscillation; In the spectrum for the radial velocity, the nonradial modes are invisible.

For the interpretation of the extracted frequencies, one has to study the evolution of the lapse function; see also the discussion in Sec. 4.1.1. The lapse function chosen by `BSSN_MoL` is is almost constant in time, with a relative change smaller than $5 \cdot 10^{-4}$. Compared to the lapse function of the initial data, which is normalised to 1 at infinity, it is smaller by a global factor of 0.99779. The frequencies with respect to coordinate time have to be divided by the same factor to obtain the frequencies observed at infinity.

The resulting frequencies are shown in Table 4.4. For comparison, we can use the values given in [17]. Those results have been obtained using the conformal flatness (CFC) approximation for the spacetime, which is described in [62]. For radial oscillations of a spherically symmetric star, the CFC approximation is exact, so at least the radial mode frequencies should agree. The difference to the frequencies given in [17] is indeed small. On the other hand it is possible that the good agreement is a coincidence, since the influence of the outer boundary location has not been investigated yet. Future simulations with more refinement levels will answer that question.

As one can see from Fig. 4.32, the density profile stays constant up to 0.1%, which is quite satisfactory. A closer look at the time evolution of

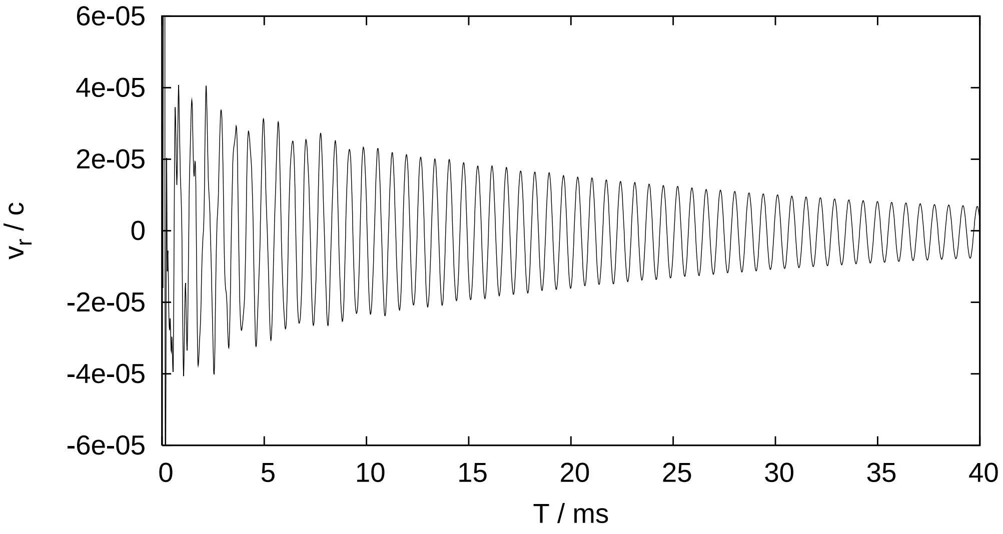


Figure 4.30: Radial velocity at $r = 7$ km on the space diagonal, for a simulation of model N with coupled spacetime evolution.

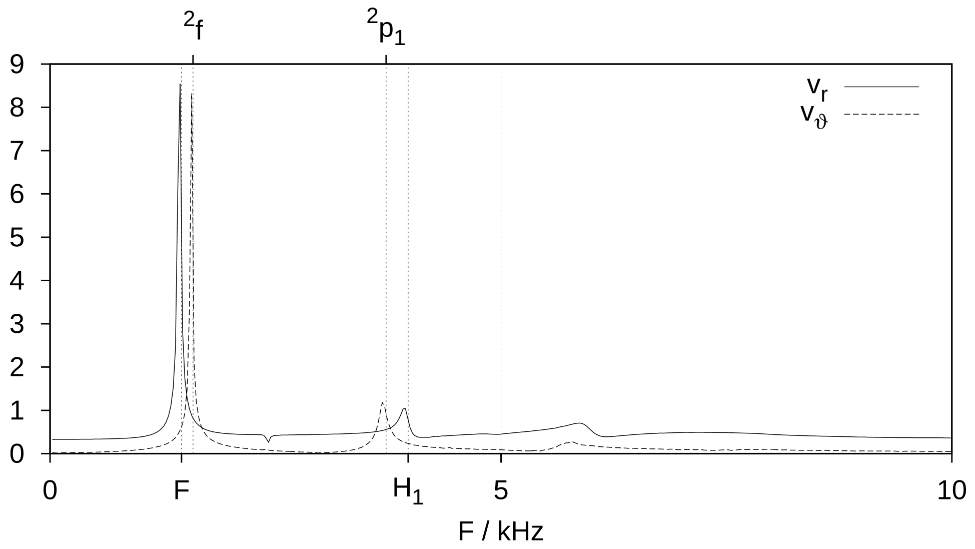


Figure 4.31: Fourier spectrum for the radial velocity shown in Fig. 4.30 and for the velocity in ϑ-direction. The latter has been scaled by a factor of 100 relative to the one for the radial velocity. The vertical lines mark the frequencies given in [17].

Mode	F	H_1	^2f	$^2p_1$
f/kHz	1.4407	3.9375	1.5735	3.6987
f_1/kHz	1.458	3.971	1.586	3.726
$(f_1 - f)/f$	1.2 %	0.9 %	0.8 %	0.8 %

Table 4.4: Mode frequencies f obtained for a simulation of model N with coupled spacetime evolution (see main text), in comparison to the values f_1 given in [17].

the density, as in Fig. 4.33, reveals a small linear drift in addition to the oscillation. Such a drift is not necessarily an error; it could be caused by a deformation of the coordinate system induced by the gauge choices. However, also the central density, which is a coordinate independent quantity, shows a drift of $1.16 \cdot 10^{-3}$ over the evolution time of 40 ms. The 3-metric shows a drift as well, but one order of magnitude smaller. For example, looking at g_{xx} on the space diagonal, one observes a relative change around $1.2 \cdot 10^{-4}$.

An important measure for the errors of a spacetime simulation is given by the constraint equations (1.90) and (1.91), which have to be satisfied for a physical spacetime. The evolution of the constraints is shown in Fig. 4.34 and 4.35. Interestingly, the momentum constraint is rapidly decreasing for some time. Closer investigations revealed that the decrease of the norm is caused globally, while the linear increase at later times originates mainly from the refinement boundary.

To my knowledge, there is no theoretical model that relates the magnitude of the constraint violations to the errors of physical quantities like oscillation frequencies. However, the average violation of the Hamiltonian constraint should at least be small compared to the density scale of the system given by the central density of the star. For the given evolution time of 40 ms, this is indeed the case. The interpretation of the momentum constraint violation is even more difficult. The quantity M_i/ρ_c has the dimension of a velocity; it should at least be small compared with the speed of light. In average, this is the case. On the other hand, it is not small compared to the fluid velocities that occur during the oscillations. From the current data, it is difficult to decide how strong the influence of the constraint violations really is. At least, they do not grow exponentially.

From Fig. 4.36, which shows the constraint violation in the x-y-plane, it is clearly visible that the constraints are violated mainly at the refinement boundary and near the surface of the star. The question wether constraint violations of the observed magnitude are an usual error caused by the mesh refinement procedure, or wether the corresponding code contains an error, is

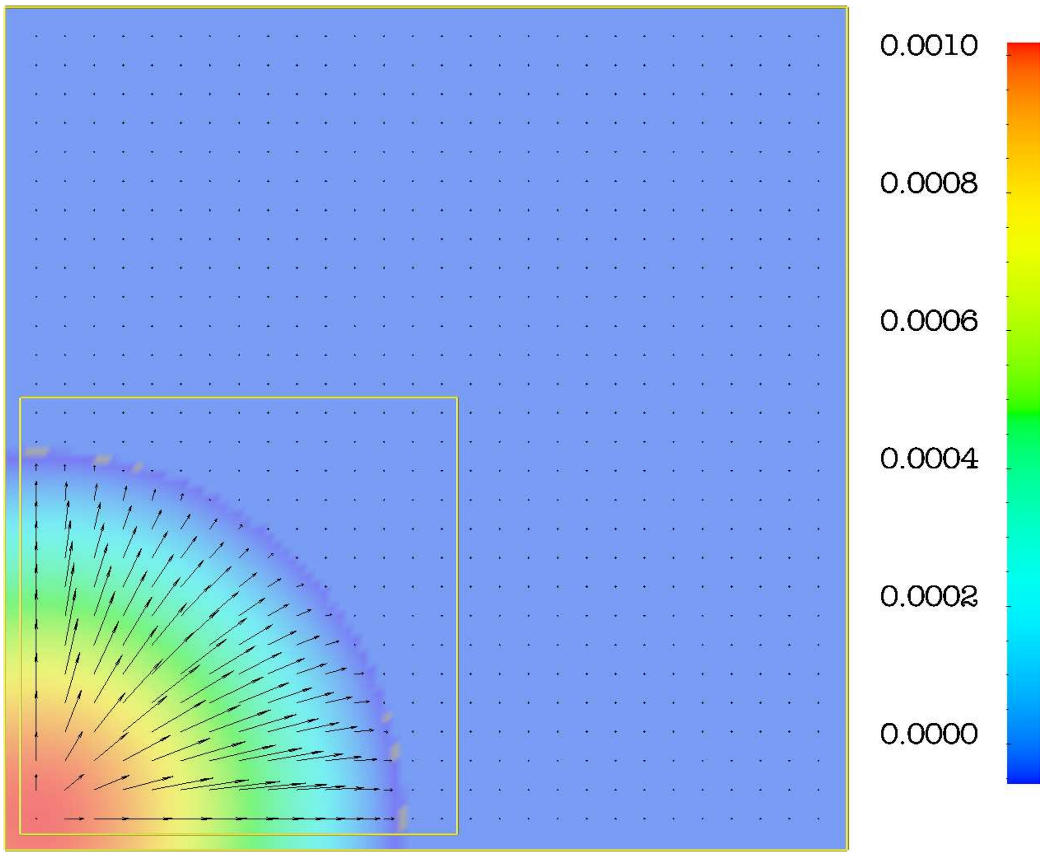


Figure 4.32: Results for a 3D simulation of model N with coupled spacetime evolution. Shown is a cut in the xy-plane. Colour coded is the change of restmass density in units of the central density. The arrows represent the momentum density. The yellow box marks the boundary of the fine grid (including two ghost cells).

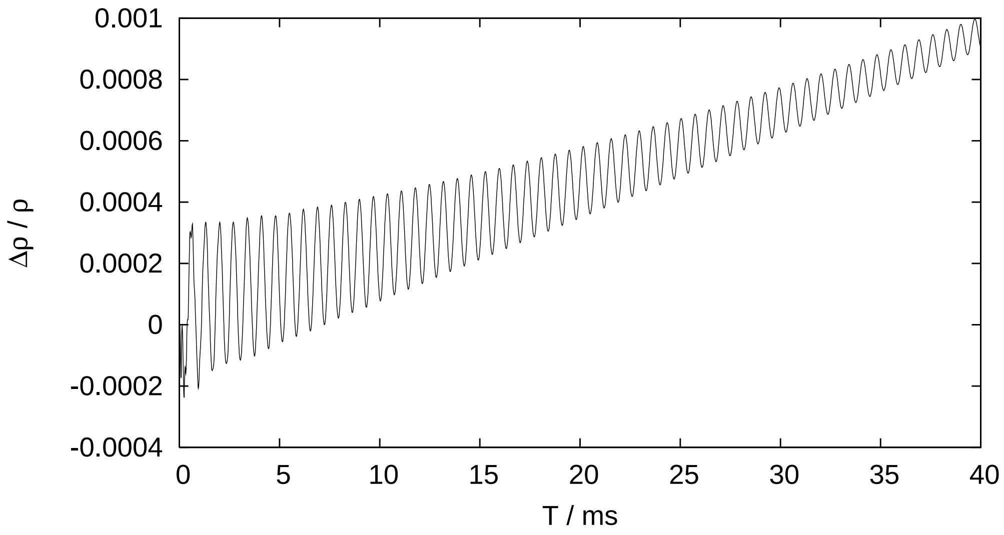


Figure 4.33: Change of restmass density at $r = 7$ km on the space diagonal.

still under investigation.

Recapitulating, the `Pizza` code in conjunction with the `BSSN_MoL` spacetime evolution code is able to evolve a spherically symmetric star at least over a time of 40 ms, with no sign of developing instabilities. It remains stationary to an accuracy around 0.1%. The oscillation frequencies extracted so far agree up to 1.2% with values given in the literature. However, the influence of the outer boundary has to be investigated by extending the computational domain. There seems to be a bug in the spacetime evolution or mesh refinement code which breaks the symmetry of the system under coordinate exchange, although the asymmetry is small. The magnitude of the constraint violations is not alarming, although their concentration at the mesh refinement boundary needs closer investigation. Finally, there is still need for a simulation with higher resolution to confirm that the code converges and to estimate the frequency errors.

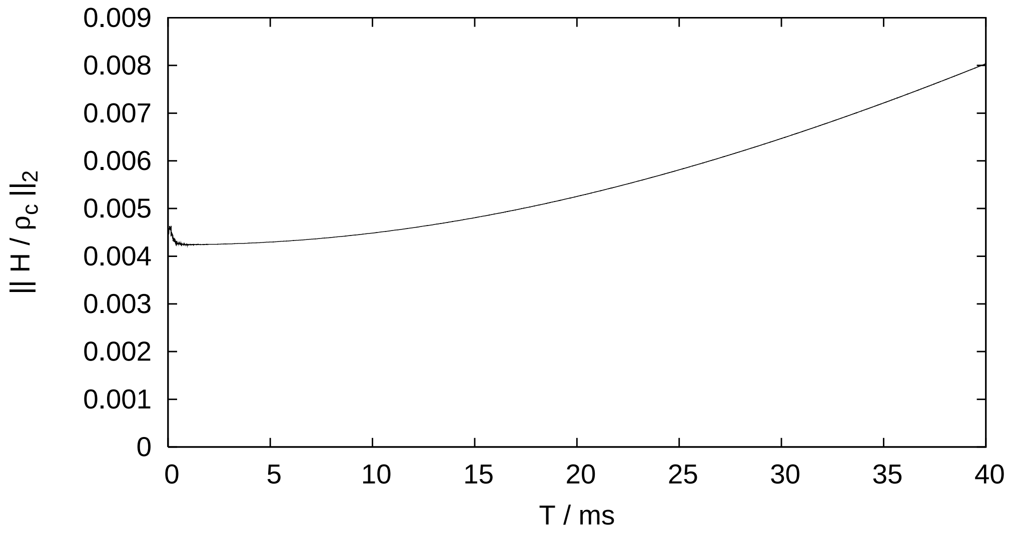


Figure 4.34: Evolution of the L_2 norm of the Hamiltonian constraint defined in Eq. (1.90), in units of the star's central density ρ_c. The norm is taken with respect to coordinate volume, not physical volume.

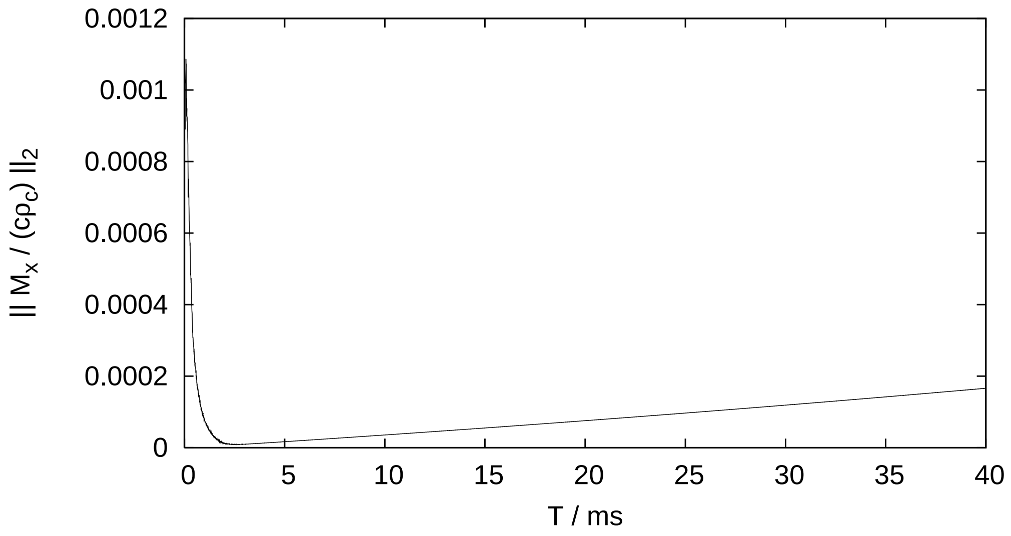


Figure 4.35: Evolution of the L_2 norm of the x-component of the momentum constraint.

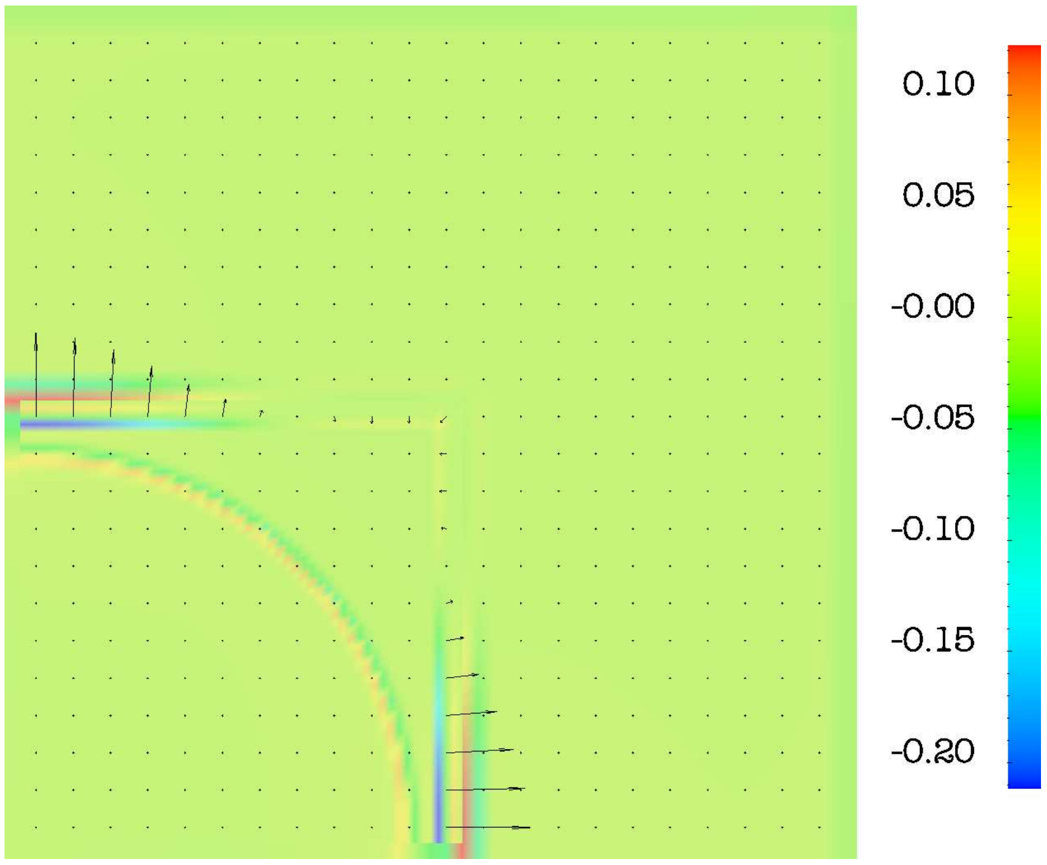


Figure 4.36: Constraint violation in the xy-plane, after an evolution time of 40 ms. Colour coded is the quantity H/ρ_c, the arrows correspond to M_i. The constraints are computed independently on both refinement levels, the plotted values are taken from the finer grid where available.

Summary

In this work, I presented a modified high resolution shock capturing scheme for general relativistic ideal hydrodynamics, which is optimised for, but not confined to, quasi-stationary, nearly isentropic spacetimes such as rigidly rotating cold neutron stars. Together with the evolution scheme I implemented a new method of handling the stellar surface. The modification of the scheme is based on a reformulation of the source terms in the hydrodynamic evolution equations which I derived especially for this purpose.

It could be demonstrated that the new scheme is well suited to study oscillations of single neutron stars. The advantages over existing nonlinear codes are the following: First, the unperturbed background model is preserved almost perfectly for rigidly rotating isentropic stars. Second, it is also possible to study oscillations of very small amplitude, which could be useful for studies of nonlinear phenomena, where the transition from a clearly linear to the nonlinear regime should be investigated. Third, the method of handling the stellar surface is not introducing significant noise, in contrast to artificial atmosphere methods.

As an application of the scheme, I studied axisymmetric oscillation modes of rigidly rotating and nonrotating neutron star models in the Cowling approximation. In particular I extracted the frequencies and also the twodimensional eigenfunctions, which are not available in the literature. I provided solid error estimates for the results on stellar oscillations by carrying out extensive numerical convergence tests. The frequency errors for practical resolutions around 50 points per stellar radius are approximately 1%, which is quite satisfactory for such kind of codes. The most prominent error is the numerical damping, which converges away with nearly second order accuracy.

A comparison of mode frequencies for two of the stellar models studied in [21] confirms these results. Also frequencies and eigenfunctions obtained in Cowling approximation with the code described in [16] agree well with my results. The only exception is the ${}^2f$-mode eigenfunction, which differs significantly near the star's centre; there are analytical hints that my results for this mode are more realistic.

So far I have only studied rigid rotation. For differential rotation, the background model would not be preserved perfectly any more (this requires corotating coordinates). However, assuming a smooth transition between the rigid rotation case and the extremely differentially rotating case, there will be a regime where my scheme still has substantial benefits.

Using a new testbed similar to a neutron star, but without a fluid-vacuum boundary, it could be shown that most of the errors in the star simulations are caused by the treatment of the stellar surface, and not by the evolution scheme itself; especially the damping is much lower without a surface and converges away with 3rd order.

In the author's opinion, the standard method for treating the surface, namely enforcing an artificial atmosphere, leads to a comparable amount of damping. For simulations of stellar oscillations, increasing the accuracy of the evolution scheme itself does not automatically lead to a significant improvement, since the error is determined by the complex interplay between evolution scheme and surface treatment. Instead, a better algorithm for treating the surface is clearly needed.

First tests of the coupling to a spacetime evolution code gave satisfactory results. However, there are still some open questions regarding the influence of the outer boundary, the violation of the constraints, and the reason for a small violation of the symmetry under coordinate exchange.

A future application of the code is the study of oscillation modes of rotating and nonrotating stars with coupled spacetime evolution in 3D. This includes nonaxisymmetric oscillations, in particular the r-mode spectrum. Additionally, the effects of differential rotation could be investigated, which is of some interest since young neutron stars are assumed to rotate differentially.

Another application of the code might be the merger of neutron stars. A binary system prior to the final merger phase does not belong to the class of spacetimes the new scheme is evolving perfectly, but is roughly similar to it in the sense that pressure and gravitational/centrifugal forces are partially balanced. Therefore it is possible that the new scheme has slight benefits also in this scenario.

Bibliography

[1] *The Cactus code*. `http://www.cactuscode.org`

[2] *Geo600 home page*. `http://www.geo600.uni-hannover.de/`

[3] *Ligo laboratory home page*. `http://www.ligo.caltech.edu/`

[4] Alcubierre, M., Brügmann, B., Dramlitsch, T., Font, J. A., Papadopoulos, P., Seidel, E., Stergioulas, N., and Takahashi, R. *Towards a stable numerical evolution of strongly gravitating systems in General Relativity: The conformal treatments*. Phys. Rev. D, vol. 62, no. 4, p. 044034 (2000)

[5] Allen, G., Andersson, N., Kokkotas, K. D., and Schutz, B. F. *Gravitational waves from pulsating stars: Evolving the perturbation equations for a relativistic star*. Phys. Rev. D, vol. 58, no. 12, p. 124012 (1998)

[6] Andersson, N. and Kokkotas, K. D. *Towards gravitational wave asteroseismology*. Mon. Not. R. Astron. Soc., vol. 299, pp. 1059 (1998)

[7] Arnowitt, R., Deser, S., and Misner, C. *The dynamics of General Relativity*. In *Gravitation: An Introduction to Current Research*. John Wiley (1962)

[8] Baiotti, L., Hawke, I., Montero, P. J., Löffler, F., Rezzolla, L., Stergioulas, N., Font, J. A., and Seidel, E. *Three-dimensional relativistic simulations of rotating neutron-star collapse to a Kerr black hole*. Phys. Rev. D, vol. 71, no. 2, p. 024035 (2005)

[9] Balakrishna, J., Daues, G., Seidel, E., Suen, W.-M., Tobias, M., and Wang, E. *Coordinate conditions in three-dimensional numerical relativity*. Classical and Quantum Gravity, vol. 13, no. 12, pp. L135 (1996)

[10] Banyuls, F., Font, J. A., Ibanez, J. M. A., Marti, J. M. A., and Miralles, J. A. *Numerical 3+1 general relativistic hydrodynamics: A local characteristic approach*. Astrophys. J., vol. 476, pp. 221 (1997)

[11] Baumgarte, T. W. and Shapiro, S. L. *Numerical integration of Einstein's field equations*. Phys. Rev. D, vol. 59, no. 2, p. 024007 (1998)

[12] Butterworth, E. and Ipser, J. R. *On the structure and stability of rapidly rotating fluid bodies in general relativity. I. The numerical method for computing structure and its applications to uniformly rotating homogeneous bodies*. Astrophys. J., vol. 204, pp. 200 (1976)

[13] Dimmelmeier, H. private communication (2006)

[14] Dimmelmeier, H., Font, J. A., and Müller, E. *Relativistic simulations of rotational core collapse I. Methods, initial models, and code tests*. A&A, vol. 388, pp. 917 (2002)

[15] Dimmelmeier, H., Font, J. A., and Müller, E. *Relativistic simulations of rotational core collapse II. Collapse dynamics and gravitational radiation*. A&A, vol. 393, pp. 523 (2002)

[16] Dimmelmeier, H., Novak, J., Font, J. A., Ibáñez, J. M., and Müller, E. *Combining spectral and shock-capturing methods: A new numerical approach for 3D relativistic core collapse simulations*. Phys. Rev. D, vol. 71, no. 6, p. 064023 (2005)

[17] Dimmelmeier, H., Stergioulas, N., and Font, J. A. *Non-linear axisymmetric pulsations of rotating relativistic stars in the conformal flatness approximation*. Mon. Not. R. Astron. Soc., vol. 368, pp. 1609 (2006)

[18] Dirac, P. A. M. *General Theory of Relativity*. John Wiley & Sons (1975)

[19] Einfeldt, B. *On Godunov-type methods for gas dynamics*. SIAM J. Num. Anal., vol. 25, no. 2, pp. 294 (1988)

[20] Font, J. A. *Numerical hydrodynamics in General Relativity*. Living Reviews in Relativity, vol. 6, no. 4 (2003)

[21] Font, J. A., Dimmelmeier, H., Gupta, A., and Stergioulas, N. *Axisymmetric modes of rotating relativistic stars in the Cowling approximation*. Mon. Not. R. Astron. Soc., vol. 325, pp. 1463 (2001)

[22] Font, J. A., Goodale, T., Iyer, S., Miller, M., Rezzolla, L., Seidel, E., Stergioulas, N., Suen, W.-M., and Tobias, M. *Three-dimensional numerical general relativistic hydrodynamics. II. long-term dynamics of single relativistic stars*. Phys. Rev. D, vol. 65, no. 8, p. 084024 (2002)

[23] Font, J. A., Miller, M., Suen, W.-M., and Tobias, M. *Three-dimensional numerical general relativistic hydrodynamics: Formulations, methods, and code tests*. Phys. Rev. D, vol. 61, no. 4, p. 044011 (2000)

[24] Font, J. A., Stergioulas, N., and Kokkotas, K. D. *Nonlinear hydrodynamical evolution of rotating relativistic stars: Numerical methods and code tests*. Mon. Not. R. Astron. Soc., vol. 313, p. 678 (2000)

[25] Harten, A., Lax, P. D., and Leer, B. V. *On upstream differencing and Godunov-type schemes for hyperbolic conservation laws*. SIAM Rev., vol. 25, no. 1, pp. 35 (1983)

[26] Hessels, J. W. T., Ransom, S. M., Stairs, I. H., Freire, P. C. C., Kaspi, V. M., and Camilo, F. *A radio pulsar spinning at 716 Hz*. Science, vol. 311, no. 5769, pp. 1901 (2006)

[27] James W. York, J. *Conformally invariant orthogonal decomposition of symmetric tensors on Riemannian manifolds and the initial-value problem of General Relativity*. Journal of Mathematical Physics, vol. 14, no. 4, pp. 456 (1973)

[28] Kley, W. private communication (2004)

[29] Leins, M., Nollert, H. P., and Soffel, M. H. *Nonradial oscillations of neutron stars: A new branch of strongly damped normal modes*. Phys. Rev. D, vol. 48, no. 8, pp. 3467 (1993)

[30] LeVeque, R., Mihalas, D., Dorfi, E., and Müller, E. *Computational Methods for Astrophysical Fluid Flows*. Springer (1997)

[31] Lindblom, L. and Detweiler, S. L. *The quadrupole oscillations of neutron stars*. Astrophys. J. Supp. Series, vol. 53, pp. 73 (1983)

[32] Lockitch, K. H. and Friedman, J. L. *Where are the r-modes of isentropic stars ?*. Astrophys. J., vol. 521, pp. 764 (1999)

[33] Martí, J. M., Ibáñez, J. M., and Miralles, J. A. *Numerical relativistic hydrodynamics: Local characteristic approach*. Phys. Rev. D, vol. 43, no. 12, pp. 3794 (1991)

[34] Martí, J. M. and Müller, E. *The analytical solution of the Riemann problem in relativistic hydrodynamics*. J. Fluid Mech., vol. 258, pp. 317 (1994)

[35] Martí, J. M. and Müller, E. *Numerical hydrodynamics in Special Relativity*. Living Reviews in Relativity, vol. 6, no. 7 (2003)

[36] Misner, C., Thorne, K., and Wheeler, J. *Gravitation*. W.H. Freeman and Company (1973)

[37] Oechslin, R., Rosswog, S., and Thielemann, F.-K. *Conformally flat smoothed particle hydrodynamics application to neutron star mergers*. Phys. Rev. D, vol. 65, no. 10, p. 103005 (2002)

[38] Oloff, R. *Geometrie der Raumzeit*. Vieweg (1999)

[39] O'Neill, B. *Semi-Riemannian Geometry: with applications to Relativity*. Acad. Pr. (1983)

[40] Oppenheimer, J. and Volkoff, G. *On massive neutron cores*. Phys. Rev., vol. 55 (1939)

[41] Pons, J., Martí, J., and Müller, E. *The exact solution of the Riemann problem with non-zero tangential velocities in relativistic hydrodynamics*. J. Fluid Mech., vol. 422, pp. 125 (2000)

[42] Ruoff, J. *New approach to the evolution of neutron star oscillations*. Phys. Rev. D, vol. 63, no. 6, p. 064018 (2001)

[43] Schnetter, E. *The Carpet code*. `http://www.carpetcode.org/`

[44] Schnetter, E., Hawley, S. H., and Hawke, I. *Evolutions in 3D numerical relativity using fixed mesh refinement*. Classical and Quantum Gravity, vol. 21, no. 6, pp. 1465 (2004)

[45] Schwarz, H. R. *Numerische Mathematik*. B.G. Teubner, Stuttgart (1993)

[46] Sexl, R. U. and Urbantke, H. K. *Gravitation und Kosmologie*. Spektrum Akademischer Verlag (2002)

[47] Shapiro, S. L. and Teukolsky, S. A. *Black holes, white dwarfs, and neutron stars : the physics of compact objects*. Wiley (1983)

[48] Shibata, M. *Fully general relativistic simulation of coalescing binary neutron stars: Preparatory tests*. Phys. Rev. D, vol. 60, no. 104052 (1999)

[49] Shibata, M., Baumgarte, T. W., and Shapiro, S. L. *The bar-mode instability in differentially rotating neutron stars: Simulations in full General Relativity*. Astrophys. J., vol. 542, pp. 453 (2000)

[50] Shibata, M. and Nakamura, T. *Evolution of three-dimensional gravitational waves: Harmonic slicing case*. Phys. Rev. D, vol. 52, no. 10, pp. 5428 (1995)

[51] Sotani, H., Kokkotas, K. D., and Stergioulas, N. *Torsional oscillations of relativistic stars with dipole magnetic fields*. ArXiv Astrophysics e-prints, p. 0608626 (2006)

[52] Sotani, H., Kokkotas, K. D., Stergioulas, N., and Vavoulidis, M. *Torsional oscillations of relativistic stars with dipole magnetic fields II. global Alfvén modes*. ArXiv Astrophysics e-prints, p. 0611666 (2006)

[53] Stergioulas, N. *Rotating stars in Relativity*. Living Reviews in Relativity, vol. 6, no. 3 (2003)

[54] Stergioulas, N., Apostolatos, T. A., and Font, J. A. *Nonlinear pulsations in differentially rotating neutron stars: Mass-shedding-induced damping and splitting of the fundamental mode*. Mon. Not. R. Astron. Soc., vol. 352, p. 1089 (2004)

[55] Stergioulas, N. and Friedman, J. L. *Comparing models of rapidly rotating relativistic stars constructed by two numerical methods*. Astrophys. J., vol. 444, pp. 306 (1995)

[56] Stroustrup, B. *The C++ programming language*. Addison-Wesley (2000)

[57] Toro, E. *Riemann Solvers and Numerical Methods for Fluid Dynamics*. Springer (1999)

[58] Wald, R. M. *General Relativity*. The University of Chicago Press (1984)

[59] Wasaburo Unno, Y. O. *Nonradial oscillations of stars*. Univ. of Tokyo Press, 2nd edn. (1989)

[60] Watts, A. L. and Strohmayer, T. E. *Detection with RHESSI of high-frequency X-ray oscillations in the tail of the 2004 hyperflare from SGR 1806-20*. Astrophys. J., vol. 637, pp. L117 (2006)

[61] Wilson, J. R. *Numerical study of fluid flow in a Kerr space*. Astrophys. J., vol. 173, p. 431 (1972)

[62] Wilson, J. R., Mathews, G. J., and Marronetti, P. *Relativistic numerical model for close neutron-star binaries*. Phys. Rev. D, vol. 54, no. 2, pp. 1317 (1996)