Monthly Notices of the ROYAL ASTRONOMICAL SOCIETY

MNRAS **508,** 2279–2301 (2021) https://doi.org/10.1093/mnras/stab2606
Advance Access publication 2021 September 16

Gmunu: paralleled, grid-adaptive, general-relativistic magnetohydrodynamics in curvilinear geometries in dynamical space–times

Patrick Chi-Kit Cheong,[1★] Alan Tsz-Lok Lam,[1,2] Harry Ho-Yin Ng[1,3] and Tjonnie Guang Feng Li[1,4,5]

[1]*Department of Physics, The Chinese University of Hong Kong, Shatin, N. T., Hong Kong*
[2]*Max Planck Institute for Gravitational Physics (Albert Einstein Institute), Am Mühlenberg 1, D-Postdam-Golm 14476, Germany*
[3]*Institut für Theoretische Physik, Goethe Universität, Max-von-Laue-Str 1, D-60438 Frankfurt am Main, Germany*
[4]*Institute for Theoretical Physics, KU Leuven, Celestijnenlaan 200D, B-3001 Leuven, Belgium*
[5]*Department of Electrical Engineering (ESAT), KU Leuven, Kasteelpark Arenberg 10, B-3001 Leuven, Belgium*

Accepted 2021 September 9. Received 2021 September 6; in original form 2021 May 13

ABSTRACT

We present an update on the General-relativistic multigrid numerical (Gmunu) code, a parallelized, multidimensional curvilinear, general relativistic magnetohydrodynamics code with an efficient non-linear cell-centred multigrid elliptic solver, which is fully coupled with an efficient block-based adaptive mesh refinement module. To date, as described in this paper, Gmunu is able to solve the elliptic metric equations in the conformally flat condition approximation with the multigrid approach and the equations of ideal general-relativistic magnetohydrodynamics by means of high-resolution shock-capturing finite-volume method with reference metric formularised multidimensionally in Cartesian, cylindrical, or spherical geometries. To guarantee the absence of magnetic monopoles during the evolution, we have developed an elliptical divergence cleaning method by using the multigrid solver. In this paper, we present the methodology, full evolution equations and implementation details of Gmunu and its properties and performance in some benchmarking and challenging relativistic magnetohydrodynamics problems.

Key words: hydrodynamics – (magnetohydrodynamics) MHD – relativistic processes – methods: numerical – software: simulations.

1 INTRODUCTION

Many astrophysical scenarios involving neutron stars and black holes such as core-collapse supernovae, mergers of compact objects are the most important events in gravitational wave physics or multimessenger astrophysics. In order to have a better understanding of such detected events and gain our understanding of the physics at nuclear densities in the post-merger remnant of binary neutron mergers (e.g. Rosswog 2015) and neutron star–black hole mergers (e.g. Metzger 2017), accurate general relativistic (magneto-)hydrodynamic (GR(M)HD) simulations are essential.

Depending on the configuration and focus of the problems, the computational cost can be significantly reduced if some symmetries can be imposed or simulating the problems in certain geometries, e.g. core-collapse supernovae (Janka et al. 2007; Burrows 2013), mangetars (Mereghetti, Pons & Melatos 2015; Turolla, Zane & Watts 2015; Kaspi & Beloborodov 2017), pulsars (Lorimer 2005), compact binary merger remnants (Shibata & Taniguchi 2011; Faber & Rasio 2012; Baiotti & Rezzolla 2017; Duez & Zlochower 2019; Radice, Bernuzzi & Perego 2020), and self-gravitating accretion discs (Abramowicz & Fragile 2013). While these problems can be simulated in 3D Cartesian coordinate, these systems with approximate symmetries are better captured in spherical or cylindrical coordinates due to better angular momentum conservation. Furthermore, the dimensionality of the problems and the computational cost can be reduced significantly if any symmetry can be imposed. Moreover, the simulation code is required not only to be robust in the highly relativistic region but also be able to resolve different scales accurately since most of such astrophysical systems are usually highly relativistic, include multitime-scale and multilength-scale physics. For instance, in stellar core-collapse problem, the length-scale could vary from the pre-supernova stellar core (thousands of kilometres) and down to a small length-scale such as the turbulence in the post-bounce flow (on the order of meters), and a typical time-step size is of the order $\mathcal{O}(10^{-6})$ s and one needs to evolve such systems up to 1–2 s for the development of a full explosion or for black hole formation (Ott 2009). Thus, to numerically model these systems accurately within a reasonable time and affordable computational resources, a multiscale, multidimensional, fully parallelized, support different geometries general relativistic (magneto-)hydrodynamics code is desired.

Several GRMHD codes are developed recently (Porth et al. 2017; Liska et al. 2019; Olivares et al. 2019; Ripperda et al. 2019; Cipolletta et al. 2020; Mewes et al. 2020). However, most of them are either designed for particular coordinates, or does not allow for a dynamical evolution of space–time. In our previous work (Cheong, Lin & Li 2020), we presented an axisymmetric general relativistic hydordynamics code Gmunu (General-relativistic multigrid numerical solver) and show that cell-centred multigrid method is an efficient and robust approach of solving the elliptic metric equations in

★ E-mail: kidcheong@gmail.com

the conformally flat condition (CFC) approximation (Dimmelmeier, Font & Müller 2002; Cordero-Carrión et al. 2009). However, the previous version of Gmunu has limitations such as it has no GRMHD solver, not parallelized, not grid adaptive, supports 2D spherical coordinates only. The aim of this work is to extend the capabilities of Gmunu to overcome these difficulties and enable us to apply it to more generic astrophysical problems. The key updates of Gmunu are the following:

(i) 1D, 2D, and 3D Cartesian, cylindrical, and spherical coordinates are supported;
(ii) general relativistic magnetohydrodynamics (GRMHD) solver is implemented;
(iii) multigrid based elliptic divergence cleaning is implemented for magnetic fields divergenceles handling;
(iv) fully parallelized with Message Passing Interface (MPI);
(v) block-based adaptive mesh refinement module is included.

The parallelization and the adaptive mesh refinement module of current Gmunu are provided by coupling with MPI-AMRVAC PDE toolkit (Xia et al. 2018; Keppens et al. 2020), a Message Passing Interface (MPI) based parallelized toolkit with a block-based quadtree-octree (in 2D–3D) Adaptive Mesh Refinement (AMR) module. In this paper, we present the methodology and the implementation details of the code and valid our code though some benchmarking tests.

The paper is organized as follows. In Section 2, we outline the formalism we used in this work. The details of the numerical settings and, the methodology, implementation of our magnetohydrodynamics solver and our multigrid solver are presented in respectively. The code tests and results are presented in Section 3. This paper ends with a discussion section in Section 5. Unless explicitly stated, we work in geometrized Heaviside–Lorentz units, for which the speed of light c, gravitational constant G, solar mass $M_\odot$, vacuum permittivity ϵ_0, and vacuum permeability μ_0 are all equal to one ($c = G = M_\odot = \epsilon_0 = \mu_0 = 1$). Greek indices, running from 0 to 3, are used for 4-quantities while the Roman indices, running from 1 to 3, are used for 3-quantities.

2 FORMULATION AND NUMERICAL METHODS

2.1 GR(M)HD in the reference-metric formalism

We use the standard ADM (Arnowitt–Deser–Misner) 3 + 1 formalism (Gourgoulhon 2007; Alcubierre 2008). The metric can be written as

$$\mathrm{d}s^2 = g_{\mu\nu}\mathrm{d}x^\mu\mathrm{d}x^\nu = -\alpha^2\mathrm{d}t^2 + \gamma_{ij}(\mathrm{d}x^i + \beta^i\mathrm{d}t)(\mathrm{d}x^j + \beta^j\mathrm{d}t) \tag{1}$$

where α is the lapse function, β^i is the space-like shift vector and γ_{ij} is the spatial metric. We adopt a conformal decomposition of the spatial metric γ_{ij} with the conformal factor ψ:

$$\gamma_{ij} = \psi^4\bar{\gamma}_{ij}, \tag{2}$$

where $\bar{\gamma}_{ij}$ is the conformally related metric.

The evolution equations for matter are derived from the local conservations of the rest mass and energy momentum and the homogeneous Faraday's law:

$$\nabla_\mu(\rho u^\mu) = 0, \tag{3}$$

$$\nabla_\mu T^{\mu\nu} = 0, \tag{4}$$

$$\nabla_\mu {}^*F^{\mu\nu} = 0, \tag{5}$$

where ρ is the rest-mass density of the fluid, u^μ is the fluid four velocity, $T^{\mu\nu}$ is the total energy-momentum tensor and ${}^*F^{\mu\nu}$ is the dual Faraday tensor. From Faraday tensor, we define the magnetic field four vector (the projection of the Faraday tensor parallel to the fluid four velocity):

$$b^\mu \equiv {}^*F^{\mu\nu}u_\nu. \tag{6}$$

With b^μ and u^μ, the total energy-momentum tensor can be expressed as:

$$T^{\mu\nu} = \rho h^* u^\mu u^\nu + p^* g^{\mu\nu} - b^\mu b^\nu, \tag{7}$$

where we further define the square of the fluid frame magnetic field strength $b^2 \equiv b^\mu b_\mu = B^2 - E^2$, the magnetically modified specific enthalpy $h^* \equiv 1 + \epsilon + (p + b^2)/\rho$ and the magnetically modified isotropic pressure $p^* \equiv p + b^2/2$.

The reference-metric formalism was originally presented in Montero, Baumgarte & Müller (2014) for GRHD and is recently extended to GRMHD (Mewes et al. 2020). By introducing a time-independent reference metric $\hat{\gamma}_{ij}$, the Valencia formulation can be generalized as the following form:

$$\partial_t(\boldsymbol{q}) + \hat{\nabla}_i(\boldsymbol{f}^i) = \boldsymbol{s}, \tag{8}$$

$$\boldsymbol{q} = \begin{bmatrix} q_D \\ q_{S_j} \\ q_\tau \\ q_{B^j} \end{bmatrix}, \boldsymbol{f}^i = \begin{bmatrix} (f_D)^i \\ (f_{S_j})^i \\ (f_\tau)^i \\ (f_{B^j})^i \end{bmatrix}, \boldsymbol{s} = \begin{bmatrix} s_D \\ s_{S_j} \\ s_\tau \\ s_{B^j} \end{bmatrix}, \tag{9}$$

where the $\hat{\nabla}_i$ here is the covariant derivatives associated with the *time-independent* reference metric $\hat{\gamma}_{ij}$. Here, $\boldsymbol{q}$ are the conserved quantities:

$$q_D \equiv \psi^6\sqrt{\bar{\gamma}/\hat{\gamma}}D = \psi^6\sqrt{\bar{\gamma}/\hat{\gamma}}[\rho W], \tag{10}$$

$$q_{S_j} \equiv \psi^6\sqrt{\bar{\gamma}/\hat{\gamma}}S_j = \psi^6\sqrt{\bar{\gamma}/\hat{\gamma}}[\rho h^* W^2 v_j - \alpha b^0 b_j], \tag{11}$$

$$q_\tau \equiv \psi^6\sqrt{\bar{\gamma}/\hat{\gamma}}\tau = \psi^6\sqrt{\bar{\gamma}/\hat{\gamma}}[\rho h^* W^2 - p^* - (\alpha b^0)^2 - D], \tag{12}$$

$$q_{B^j} \equiv \psi^6\sqrt{\bar{\gamma}/\hat{\gamma}}B^j, \tag{13}$$

where $v^i = u^i/W + \beta^i/\alpha$ is the 3-velocity seen by an Eulerian observer at rest in current spatial 3-hypersurface, $W \equiv 1/\sqrt{1 - v^i v_i}$ is the Lorentz factor. The magnetic field in fluid's rest frame can be obtained by

$$b^0 = \frac{W B^k v_k}{\alpha}, \qquad b^i = \frac{B^i}{W} + b^0\hat{v}^i, \tag{14}$$

$$b^2 = \frac{B^i B_i}{W^2} + (B^k v_k)^2, \tag{15}$$

where $\hat{v}^i \equiv (\alpha v^i - \beta^i)$. Note that $b_i = b^\mu g_{\mu i} = B_i/W + \alpha b^0 v_i$. The corresponding fluxes $\boldsymbol{f}^i$ are given by

$$(f_D)^i \equiv \psi^6\sqrt{\bar{\gamma}/\hat{\gamma}}[D\hat{v}^i], \tag{16}$$

$$(f_{S_j})^i \equiv \psi^6\sqrt{\bar{\gamma}/\hat{\gamma}}[S_j\hat{v}^i + \delta^i_j\alpha p^* - \alpha b_j B^i/W], \tag{17}$$

$$(f_\tau)^i \equiv \psi^6\sqrt{\bar{\gamma}/\hat{\gamma}}[\tau\hat{v}^i + \alpha p^* v^i - \alpha^2 b^0 B^i/W], \tag{18}$$

$$(f_{B^j})^i \equiv \psi^6\sqrt{\bar{\gamma}/\hat{\gamma}}[\hat{v}^i B^j - \hat{v}^j B^i]. \tag{19}$$

Finally, the corresponding source terms $\boldsymbol{s}$ are given by

$$s_D = 0, \tag{20}$$

$$s_{S_i} = \alpha\psi^6\sqrt{\bar{\gamma}/\hat{\gamma}}\Big\{-T^{00}\alpha\partial_i\alpha + T^0_k\hat{\nabla}_i\beta^k + \frac{1}{2}(T^{00}\beta^j\beta^k + 2T^{0j}\beta^k + T^{jk})\hat{\nabla}_i\gamma_{jk}\Big\}, \quad (21)$$

$$s_\tau = \alpha\psi^6\sqrt{\bar{\gamma}/\hat{\gamma}}\{T^{00}(K_{ij}\beta^i\beta^j - \beta^k\partial_k\alpha) + T^{0j}(2K_{jk}\beta^k - \partial_j\alpha) + T^{ij}K_{ij}\}, \quad (22)$$

$$s_{B^i} = 0, \quad (23)$$

where K_{ij} is the extrinsic curvature.

In order to solve equation (8) with the finite volume formulation, we further express the equations in the following form:

$$\partial_t\boldsymbol{q} + \frac{1}{\sqrt{\hat{\gamma}}}\partial_j[\sqrt{\hat{\gamma}}\boldsymbol{f}^j] = \boldsymbol{s} + \boldsymbol{s}_{\text{geom}}, \quad (24)$$

where $\boldsymbol{s}_{\text{geom}}$ are so-called geometrical source terms which contain the 3-Christoffel symbols $\hat{\Gamma}^l_{ik}$ associated with the reference metric $\hat{\gamma}_{ij}$. Explicitly, equation (24) can be expressed as:

$$\partial_t(q_D) + \frac{1}{\sqrt{\hat{\gamma}}}\partial_j[\sqrt{\hat{\gamma}}(f_D)^j] = 0, \quad (25)$$

$$\partial_t(q_{S_i}) + \frac{1}{\sqrt{\hat{\gamma}}}\partial_j[\sqrt{\hat{\gamma}}(f_{S_i})^j] = s_{S_i} + \hat{\Gamma}^l_{ik}(f_{S_l})^k, \quad (26)$$

$$\partial_t(q_\tau) + \frac{1}{\sqrt{\hat{\gamma}}}\partial_j[\sqrt{\hat{\gamma}}(f_\tau)^j] = s_\tau, \quad (27)$$

$$\partial_t(q_{B^i}) + \frac{1}{\sqrt{\hat{\gamma}}}\partial_j[\sqrt{\hat{\gamma}}(f_{B^i})^j] = 0. \quad (28)$$

Note that the momentum conservation in this expression are satisfied to *machine precision* rather than to the level of truncation error because the geometrical source terms $\boldsymbol{s}_{\text{geom}}$ are identically vanishing for the components associated with ignorable coordinates in the metric. For example, in spherical coordinates (r, θ, ϕ), since the coordinate ϕ does not explicitly enter into the metric, the corresponding geometrical source term vanish for the q_{S_ϕ} equations. Physically, unlike in the expression in Montero et al. (2014) and Mewes et al. (2020) where the angular momentum is conserved to the level of truncation error due to the explicit expression of the covariant derivatives, in our expression, the angular momentum conservation is numerically satisfied to machine precision since the corresponding geometrical source term is identically equal to zero. Similar implementations that minimize coordinate-dependent part of the code can be found in Gammie, McKinney & Tóth (2003) and in a recent work Skinner et al. (2019)

We then discretize the volume averages of equation (24). Using divergence theorem and some algebra, the discretized version of equation (24) in the cell (i, j, k) can be expressed as

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\langle\boldsymbol{q}\rangle_{\mathtt{i,j,k}} = &\frac{1}{\Delta V_{\mathtt{i,j,k}}}\\ &\times\{[(\langle\boldsymbol{f}\rangle^1\Delta A^1)|_{\mathtt{i+1/2,j,k}} - (\langle\boldsymbol{f}\rangle^1\Delta A^1)|_{\mathtt{i-1/2,j,k}}]\\ &+[(\langle\boldsymbol{f}\rangle^2\Delta A^2)|_{\mathtt{i,j+1/2,k}} - (\langle\boldsymbol{f}\rangle^2\Delta A^2)|_{\mathtt{i,j-1/2,k}}]\\ &+[(\langle\boldsymbol{f}\rangle^3\Delta A^3)|_{\mathtt{i,j,k+1/2}} - (\langle\boldsymbol{f}\rangle^3\Delta A^3)|_{\mathtt{i,j,k-1/2}}]\}\\ &+\langle\boldsymbol{s}\rangle_{\mathtt{i,j,k}} + \langle\boldsymbol{s}_{\text{geom}}\rangle_{\mathtt{i,j,k}},\end{aligned} \quad (29)$$

where the cell volume and volume-average are defined as

$$\Delta V \equiv \int_{\text{cell}}\sqrt{\hat{\gamma}}\mathrm{d}x^1\mathrm{d}x^2\mathrm{d}x^3, \quad (30)$$

$$\langle\bullet\rangle \equiv \frac{1}{\Delta V}\int_{\text{cell}}\bullet\sqrt{\hat{\gamma}}\mathrm{d}x^1\mathrm{d}x^2\mathrm{d}x^3, \quad (31)$$

while the surface area and surface-average is defined as

$$\Delta A^i \equiv \int_{\text{surface}}\sqrt{\hat{\gamma}}\mathrm{d}x^{j,j\neq i}, \quad (32)$$

$$\langle\bullet\rangle^i \equiv \frac{1}{\Delta A^i}\int_{\text{surface}}\bullet^i\sqrt{\hat{\gamma}}\mathrm{d}x^{j,j\neq i}. \quad (33)$$

Here, we note that, as the reference metric $\hat{\gamma}_{ij}$ is *time-independent*, the volume-averaged 3-Christoffel symbols $\langle\hat{\Gamma}^l_{ik}\rangle$ in the geometrical source terms, cell volume ΔV and surface area ΔA are fixed once the coordinate system is chosen. For completeness, we included these quantities in both cylindrical and spherical coordinates in Appendix A.

2.2 Divergenceless handling and elliptic divergence cleaning

The time-component of equation (5) implies that the divergence of the magnetic field is zero, namely:

$$\nabla\cdot\boldsymbol{B} \equiv \frac{1}{\sqrt{\gamma}}\partial_i(\sqrt{\gamma}B^i) = 0$$
$$\Rightarrow \hat{\nabla}_i q_{B^i} = \frac{1}{\sqrt{\hat{\gamma}}}\partial_i(\sqrt{\hat{\gamma}}q_{B^i}) = 0. \quad (34)$$

In practice, this condition is not satisfied if we evolve the induction equation (28) directly without any treatment due to the accumulating numerical error. As a result, non-vanishing monopoles are introduced and the code returns non-physical results. Various treatments are introduced to enforce this constraint in (GR)MHD calculations. The most common approaches recently are (i) hyperbolic divergence cleaning through a generalized Lagrange multiplier (GLM) (e.g. Porth et al. 2017); (ii) constrained transport (CT) scheme which updates the magnetic fields while controlling the divergence-free constraint to numerical round-off accuracy (e.g. Porth et al. 2017; Olivares et al. 2019); and (iii) evolving the vector potentials directly and compute the magnetic field by taking the curl of the vector potential (e.g. Mewes et al. 2020). Here, we adopt a different approach, the so-called *elliptic* divergence cleaning, by solving Poisson's equation and enforce the magnetic field is divergence-free:

$$\hat{\nabla}^2\Phi = \hat{\nabla}_i q^{\text{old}}_{B^i}, \quad (35)$$

$$q^{\text{new}}_{B^i} = q^{\text{old}}_{B^i} - (\hat{\nabla}\Phi)^i. \quad (36)$$

The BHAC code (Porth et al. 2017), elliptic divergence cleaning is available to be used only for the magnetic fields initialization (Teunissen & Keppens 2019).

In the current implementation of Gmunu with elliptic divergence cleaning, the magnetic field is defined at cell centres. Whenever the conserved magnetic field q_{B^i} is updated at each time-step, we first solve Poisson's equation in equation (35) through the multigrid solver (see Section 2.7), then we update the magnetic field with the solution Φ as shown in equation (36).

In addition to the elliptic cleaning mentioned above, generalized Lagrange multiplier (GLM), constrained transport (CT), and the vector potential schemes are planned for Gmunu. The implementations and comparisons of these divergence-free treatments will be presented in future work. Here, we will only focus on the elliptic divergence cleaning approach as our main divergence-free treatment for evolution.

2.3 Characteristic speed

In relativistic magnetohydrodynamics, one has to solve a quartic equation if we wish to obtain the exact form of the characteristic wave speeds $\lambda_\pm$ (e.g. Anile 1990). To reduce the computational cost and complexity of the implementation, instead of obtaining the exact characteristic speeds, we follow the approach presented in Gammie et al. (2003). In this approach, the upper bound a for the fast wave speed is

$$a^2 = c_s^2 + c_a^2 - c_s^2 c_a^2, \tag{37}$$

where c_s is the sound speed and c_a is the Alfven speed which can be obtained by

$$c_a^2 = \frac{b^2}{\rho h + b^2} = \frac{b^2}{\rho h^*}. \tag{38}$$

The characteristic velocities can then be calculated by

$$\lambda^i_\pm = \alpha \bar{\lambda}^i_\pm - \beta^i, \tag{39}$$

$$\bar{\lambda}^i_\pm = \frac{(1-a^2)v^i \pm \sqrt{a^2(1-v^2)[(1-v^2a^2)\gamma^{ii} - (1-a^2)(v^i)^2]}}{(1-v^2a^2)}. \tag{40}$$

2.4 Positivity preserving limiter

Positivity preserving limiter was originally introduced in Hu, Adams & Shu (2013) for Newtonian hydrodynamics and was successfully applied on GR(M)HD Radice, (Rezzolla & Galeazzi 2014; Porth et al. 2017). Here, we will discuss the basic concept of positivity preserving limiter and its implementation in Gmunu. For simplicity, let us consider the evolution equation of conserved density in 1D case:

$$\partial_t(u) + \frac{1}{\sqrt{\hat{\gamma}}}\partial_1[\sqrt{\hat{\gamma}}(f(u))] = 0. \tag{41}$$

Note that if the positivity of u is guaranteed over one first-order Euler time-step, then the positivity is also guaranteed for any strong-stability preserving Runge–Kutta (SSPRK) time integrator since the time integrator is always constructed as a convex combination of Euler steps. So, we discretized equation (41) as the following form:

$$\frac{u^{\mathtt{n+1}}_{\mathtt{i}} - u^{\mathtt{n}}_{\mathtt{i}}}{\Delta t} = -\frac{1}{\Delta V_{\mathtt{i}}}\left\{f_{\mathtt{i+1/2}}\Delta A_{\mathtt{i+1/2}} - f_{\mathtt{i-1/2}}\Delta A_{\mathtt{i-1/2}}\right\}$$
$$\Rightarrow u^{\mathtt{n+1}}_{\mathtt{i}} = \frac{1}{2}\left[u^-_{\mathtt{i}} + u^+_{\mathtt{i}}\right], \tag{42}$$

where

$$u^-_{\mathtt{i}} \equiv \left(u^{\mathtt{n}}_{\mathtt{i}} - 2\frac{\Delta t}{\Delta V_{\mathtt{i}}} f_{\mathtt{i+1/2}}\Delta A_{\mathtt{i+1/2}}\right),$$
$$u^+_{\mathtt{i}} \equiv \left(u^{\mathtt{n}}_{\mathtt{i}} + 2\frac{\Delta t}{\Delta V_{\mathtt{i}}} f_{\mathtt{i-1/2}}\Delta A_{\mathtt{i-1/2}}\right). \tag{43}$$

To ensure both $u^+_{\mathtt{i}}$ and $u^-_{\mathtt{i}}$ are positive, we modify the flux as

$$f_{\mathtt{i+1/2}} = \theta f^{\mathrm{HO}}_{\mathtt{i+1/2}} + (1-\theta) f^{\mathrm{LF}}_{\mathtt{i+1/2}}, \tag{44}$$

where $f^{\mathrm{HO}}_{\mathtt{i+1/2}}$ is the high-order flux of the original scheme while $f^{\mathrm{LF}}_{\mathtt{i+1/2}}$ is the first-order Lax–Friedrichs flux and $\theta \in [0, 1]$ is the maximum value such that both $u^+_{\mathtt{i+1}}$ and $u^-_{\mathtt{i}}$ are positive. Since the Lax–Friedrichs scheme is positivity preserving, it is always possible to choose some θ such that positivity is guaranteed. In Gmunu, we implemented this limiter to preserve the positivity of conserved density D and energy density τ, to preserve the positivity of density ρ and pressure p. For multidimensional cases, we apply the limiter component by component.

2.4.1 *Implementation of positivity preserving limiter*

After calculating $f^{\mathrm{HO}}_{\mathtt{i+1/2}}$ and $f^{\mathrm{LF}}_{\mathtt{i+1/2}}$, we check if the the relationship $f_{\mathtt{i+1/2}} = f^{\mathrm{HO}}_{\mathtt{i+1/2}}$ needs to be modified with a small value ϵ (which is set as 10^{-16} in Gmunu), i.e. we check if the following relations hold:

$$u^-_{\mathtt{i}} = \left(u^{\mathtt{n}}_{\mathtt{i}} - 2\frac{\Delta t}{\Delta V_{\mathtt{i}}} f_{\mathtt{i+1/2}}\Delta A_{\mathtt{i+1/2}}\right) > \epsilon$$
$$u^+_{\mathtt{i}} = \left(u^{\mathtt{n}}_{\mathtt{i}} + 2\frac{\Delta t}{\Delta V_{\mathtt{i}}} f_{\mathtt{i-1/2}}\Delta A_{\mathtt{i-1/2}}\right) > \epsilon. \tag{45}$$

If relations above do not hold, we then work out θs by substituting (44) into (45):

$$\theta^-_{\mathtt{i}} = \frac{\frac{\Delta V_{\mathtt{i}}}{2\Delta t}\left(u^{\mathtt{n}}_{\mathtt{i}} - \epsilon\right) - f^{\mathrm{LF}}_{\mathtt{i+1/2}}\Delta A_{\mathtt{i+1/2}}}{f^{\mathrm{HO}}_{\mathtt{i+1/2}}\Delta A_{\mathtt{i+1/2}} - f^{\mathrm{LF}}_{\mathtt{i+1/2}}\Delta A_{\mathtt{i+1/2}}},$$
$$\theta^+_{\mathtt{i}} = \frac{\frac{\Delta V_{\mathtt{i}}}{2\Delta t}\left(u^{\mathtt{n}}_{\mathtt{i}} - \epsilon\right) + f^{\mathrm{LF}}_{\mathtt{i-1/2}}\Delta A_{\mathtt{i-1/2}}}{f^{\mathrm{LF}}_{\mathtt{i-1/2}}\Delta A_{\mathtt{i-1/2}} - f^{\mathrm{HO}}_{\mathtt{i-1/2}}\Delta A_{\mathtt{i-1/2}}},$$
$$\theta_{\mathtt{i}} = \min(\theta^+_{\mathtt{i}}, \theta^-_{\mathtt{i}}),$$
$$\theta_{\mathtt{i}} = \min(\max(\theta_{\mathtt{i}}, 0), 1). \tag{46}$$

After obtaining θs for both continuity and energy equations, we pick the maximum one and substitute the resulting θ into (44) to modify the *all* the flux terms at that particular grid point.

2.5 Conserved to primitive variables conversion

Recovery of the primitive variables (ρ, Wv^i, p) from the conservative variables (D, S_i, τ) in GR(M)HD is non-trivial, one has to solve non-linear equations numerically. Most of the root-finding methods used in GR(M)HD simulations are Newton–Raphson method which works fine with analytic equations of state. However, it might return inaccurate results with tabulated equations of state because it requires the partial derivatives $\partial\hat{p}/\partial\rho$ and $\partial\hat{p}/\partial\epsilon$. In Gmunu, two conserved to primitive variables conversions which do not require derivatives are implemented for GRHD and GRMHD, respectively. For the GRHD cases, the implementation basically follows the formulation presented in appendix C in Galeazzi et al. (2013) while for the GRMHD cases we mainly follow a recent work (Kastaun, Kalinani & Ciolfi 2021). Although GRHD can be reduced from GRMHD by letting all magnetic field $B^i = 0$ and the recovery of primitive variables method presented in Kastaun et al. (2021) actually works well for vanishing magnetic fields, for different applications and development purposes (e.g. for the systems that have no magnetic fields, it is better to use the GRHD module to lower the computational cost), we implemented two separate modules called grhd and grmhd correspondingly. For completeness, we included the implementation details of both GRHD and GRMHD here.

2.5.1 *Implementation of recovery of primitive variables in GRHD*

(i) Calculate the rescaled variables and the following useful relations which are fixed during the iterations.

$$S \equiv \sqrt{S^i S_i}, \tag{47}$$

$$r \equiv \frac{S}{D}, \qquad q \equiv \frac{\tau}{D}, \qquad k \equiv \frac{S}{\tau + D}, \tag{48}$$

(ii) Determine the bounds of the root z_- and z_+, where

$$z_- \equiv \frac{k/2}{\sqrt{1-k^2/4}}, \qquad z_+ \equiv \frac{k}{\sqrt{1-k^2}} \tag{49}$$

(iii) In the inverval $[z_-, z_+]$, solve:

$$f(z) = z - \frac{r}{\hat{h}(z)}, \tag{50}$$

where

$$\hat{h}(z) = (1 + \hat{\epsilon}(z))(1 + \hat{a}(z)), \tag{51}$$

$$\hat{p}(z) = p(\hat{\rho}(z), \hat{\epsilon}(z)), \qquad \hat{a}(z) = \frac{\hat{p}(z)}{\hat{\rho}(z)(1 + \hat{\epsilon}(z))}, \tag{52}$$

$$\hat{\rho}(z) = \frac{D}{\hat{W}(z)}, \tag{53}$$

$$\hat{\epsilon}(z) = \hat{W}(z)q - zr + \frac{z^2}{1 + \hat{W}(z)}, \tag{54}$$

$$\hat{W}(z) = \sqrt{1 + z^2}. \tag{55}$$

In Gmunu, we numerically solve equation (50) with the Illinois algorithm Dowell & Jarratt (1971), which is an improved version of Regula–Falsi method. Note that during the iterations, we enforce that the density ρ and the specific energy ϵ fall within the validity region of the EOS, i.e. we evaluate the updated ρ and ϵ with $\hat{\rho} = \max(\min(\rho_{\max}, \hat{\rho}), \rho_{\min})$ and $\hat{\epsilon} = \max(\min(\epsilon_{\max}(\hat{\rho}), \hat{\epsilon}), \epsilon_{\min}(\hat{\rho}))$.

(iv) With the root z_0 of equation (50), we can then work out the primitive variables $[\rho, \epsilon, p]$, respectively, with the equations used in step 3. The velocity v^i can be obtained with z by

$$\hat{v}^i(z) = \frac{S^i/D}{\hat{h}(z)\hat{W}(z)}. \tag{56}$$

2.5.2 Implementation of recovery of primitive variables in GRMHD

(i) Calculate the rescaled variables and the following useful relations which are fixed during the iterations.

$$q \equiv \frac{\tau}{D}, \qquad r_i \equiv \frac{S_i}{D}, \qquad \mathcal{B}^i \equiv \frac{B^i}{\sqrt{D}}, \tag{57}$$

and then calculate

$$r^2 \equiv r^i r_i, \quad \mathcal{B}^2 \equiv \mathcal{B}^i \mathcal{B}_i \quad \text{and} \quad \mathcal{B}^2 r_\perp^2 \equiv \mathcal{B}^2 r^2 - (r^l \mathcal{B}_l)^2. \tag{58}$$

(ii) In the interval $(0, h_0^{-1}]$, solve:

$$f_a(\mu) = \mu\sqrt{h_0^2 + \bar{r}^2(\mu)} - 1, \tag{59}$$

where h_0 is the relativistic enthalpy lower bound over the entire validity region of the EOS and

$$\bar{r}^2(\mu) = r^2\chi^2(\mu) + \mu\chi(\mu)(1 + \chi(\mu))(r^l\mathcal{B}_l)^2, \tag{60}$$

$$\chi(\mu) = \frac{1}{1 + \mu\mathcal{B}^2}. \tag{61}$$

Here, the root of f_a in equation(59) is denoted as μ_+. Since f_a is smooth and its derivative can be expressed analytically, we numerically solve equation (59) with Newton–Raphson method, which is usually more efficient than bracketing methods. In case the Newton–Raphson method fails to converge, we use the Illinois algorithm to solve this equation.

(iii) In the interval $(0, \mu_+]$, solve:

$$f(\mu) = \mu - \frac{1}{\hat{\nu} + \mu\bar{r}^2(\mu)}, \tag{62}$$

where

$$\hat{\nu}(\mu) = \max(\nu_A(\mu), \nu_B(\mu)), \tag{63}$$

$$\nu_A(\mu) = (1 + \hat{a}(\mu))\frac{1 + \hat{\epsilon}(\mu)}{\hat{W}(\mu)}, \tag{64}$$

$$\nu_B(\mu) = (1 + \hat{a}(\mu))(1 + \bar{q}(\mu) - \mu\bar{r}^2(\mu)) \tag{65}$$

$$\hat{p}(\mu) = p(\hat{\rho}(\mu), \hat{\epsilon}(\mu)), \qquad \hat{a}(\mu) = \frac{\hat{p}(\mu)}{\hat{\rho}(\mu)(1 + \hat{\epsilon}(\mu))}, \tag{66}$$

$$\hat{\rho}(\mu) = \frac{D}{\hat{W}(\mu)},$$

$$\hat{\epsilon}(\mu) = \hat{W}(\mu)(\bar{q}(\mu) - \mu\bar{r}^2(\mu)) + \hat{v}^2(\mu)\frac{\hat{W}^2(\mu)}{1 + \hat{W}(\mu)}, \tag{67}$$

$$\hat{v}^2(\mu) = \min(\mu^2\bar{r}^2(\mu), v_0^2), \qquad \hat{W}(\mu) = \frac{1}{\sqrt{1 - \hat{v}^2(\mu)}}, \tag{68}$$

$$\bar{q}(\mu) = q - \frac{1}{2}\mathcal{B}^2 - \frac{1}{2}\mu^2\chi^2(\mu)(\mathcal{B}^2 r_\perp^2), \tag{69}$$

$\bar{r}^2(\mu)$ and $\chi(\mu)$ are defined in equations (60) and (61), and the upper velocity limit square v_0^2 is defined as $v_0^2 \equiv r^2/(h_0^2 + r^2) < 1$. In Gmunu, we numerically solve equation (62) with the Illinois algorithm. Note that during the iterations, we enforce the density ρ and the specific energy ϵ fall within the validity region of the EOS, i.e. we evaluate the updated ρ and ϵ with $\hat{\rho} = \max(\min(\rho_{\max}, \hat{\rho}), \rho_{\min})$ and $\hat{\epsilon} = \max(\min(\epsilon_{\max}(\hat{\rho}), \hat{\epsilon}), \epsilon_{\min}(\hat{\rho}))$.

(iv) With the root μ of equation (62), we can then work out the primitive variables $[\rho, \epsilon, p]$ respectively with the equations used in step 3. The velocity v^i can be obtained with μ by

$$\hat{v}^i(\mu) = \mu\chi(\mu)(r^i + \mu(r^l\mathcal{B}_l)\mathcal{B}^i). \tag{70}$$

Note that, as in Kastaun et al. (2021), we assume positive baryon number density, positive total energy density, and positive pressure. For the case of the classical ideal-gas equation of state, the specific energy is also non-negative, i.e. $\epsilon \geq 0$. It is worth to point out that, since the definitions of the specific energy ϵ and the relativistic enthalpy depend on the arbitrary choice of the mass constant m_B, relations such as $\epsilon > 0$ or $h \geq 1$ may not hold in general. For example, negative specific energy ϵ is possible in nuclear physics equations of state.

The recovery scheme described here has been shown to be robust and efficient in typical scenarios such as binary neutron star mergers and core-collapse supernovae, where the rescaled magnetic field $\mathcal{B}^i := B^i/\sqrt{D}$ should below 10^2 Kastaun et al. (2021). Although $\mathcal{B}^i \sim 10^4$ is far beyond practical uses, this scheme still converges in this case with around 40 iterations (Kastaun et al. 2021).

2.5.3 Error handling

Although some primitive variables are enforced to fall within the validity region during the iterations, this conserved to primitive variables conversion occasionally return unphysical results, especially at the surfaces of neutron stars. These errors are mostly harmless and can be corrected. After the primitive variables are obtained, we check whether any correction is needed. By following Galeazzi et al. (2013), some corrections are allowed only for low density region or at black hole centre. Low density region is defined as $\rho < \rho_{\rm low}$ and inside a black hole is defined as $\alpha < \alpha_{\rm BH}$. In this work, we set $\rho_{\rm low} = \rho_{\rm atmo} \times 10^2$ while $\alpha_{\rm BH} = 10^{-2}$. The error handling processes are the following:

(i) $\rho < \rho_{\min}$: set everything to atmosphere. In particular, we enforce the rest-mass density to be $\rho_{\rm atmo}$ and the velocity is set to be zero, then update the rest of the primitive variables such as pressure p and specific energy density ϵ by using polytropic equation of state.

(ii) $\rho > \rho_{\max}$: a fatal error, stop the code.
(iii) $\epsilon < \epsilon_{\min}$: set $\epsilon = \epsilon_{\min}$.
(iv) $\epsilon > \epsilon_{\max}$: set $\epsilon = \epsilon_{\max}$ for low density or at black hole centre, otherwise it is a fatal error.
(v) $v > v_{\max}$: adjust for low density or at black hole centre, otherwise it is a fatal error. In particular, we rescale the velocity such that $v = v_{\max}$ as well as the Lorentz factor $W_{\max}$. Here, we keep conserved density D fixed, the rest-mass density ρ increase slightly. We limit the rest-mass density and the specific energy ϵ again.
(vi) The electron fraction Y_e out of range: adjust for low density or at black hole centre, otherwise it is a fatal error.

2.6 Metric equations and conformal flatness approximation

In this work, we adopt conformal flatness approximation and solve the Einstein field equations with xCFC scheme as in Cheong et al. (2020). For the details of CFC/ xCFC schemes and how to numerically solve the metric equations, we refer readers to Dimmelmeier et al. (2002), Cordero-Carrión et al. (2009), Bucciantini & Del Zanna (2011), and Cheong et al. (2020). Here, we briefly outline the basic equations and the formulations.

In a CFC approximation (Dimmelmeier et al. 2002; Bucciantini & Del Zanna 2011), the three metric γ_{ij} is assumed to be decomposed according to

$$\gamma_{ij} := \psi^4 f_{ij}, \tag{71}$$

where f_{ij} is a time-independent flat background metric and ψ is the conformal factor which is a function of space and time. In the updated implementation, we let the flat background metric f_{ij} equals the reference metric $\hat{\gamma}_{ij}$. Another assumption is the maximal slicing condition of foliations $K = 0$. For the matter sources, we define: $U \equiv n_\mu n_\nu T^{\mu\nu}$, $S^i \equiv -n_\mu \gamma^i_\nu T^{\mu\nu}$, and $S^{ij} \equiv \gamma^i_\mu \gamma^j_\nu T^{\mu\nu}$, where $T^{\mu\nu}$ is the energy-momentum tensor. In the xCFC scheme, one introduces a vector potential X^i, and the metric can be solved by the following equations:

$$\tilde{\Delta} X^i + \frac{1}{3}\tilde{\nabla}^i(\tilde{\nabla}_j X^j) = 8\pi f^{ij}\tilde{S}_j, \tag{72}$$

$$\tilde{\Delta}\psi = -2\pi\tilde{U}\psi^{-1} - \frac{1}{8} f_{ik} f_{jl}\tilde{A}^{kl}\tilde{A}^{ij}\psi^{-7}, \tag{73}$$

$$\tilde{\Delta}(\alpha\psi) = (\alpha\psi)\left[2\pi(\tilde{U} + 2\tilde{S})\psi^{-2} + \frac{7}{8} f_{ik} f_{jl}\tilde{A}^{kl}\tilde{A}^{ij}\psi^{-8}\right], \tag{74}$$

$$\tilde{\Delta}\beta^i + \frac{1}{3}\tilde{\nabla}^i(\tilde{\nabla}_j\beta^j) = 16\pi\alpha\psi^{-6} f^{ij}\tilde{S}_i + 2\tilde{A}^{ij}\tilde{\nabla}_j(\alpha\psi^{-6}), \tag{75}$$

where $\tilde{\nabla}_i$ and $\tilde{\Delta}$ are the covariant derivative and the Laplacian with respect to the flat three metric f_{ij}, respectively, and $\tilde{U} := \psi^6 U$, $\tilde{S}_i := \psi^6 S_i$ and $\tilde{S} := \psi^6 S = \psi^6 \gamma_{ij} S^{ij}$ are the rescaled fluid source terms. The tensor field $\tilde{A}^{ij}$ can be approximated on the CFC approximation level by (see the appendix of Cordero-Carrión et al. 2009)

$$\tilde{A}^{ij} \approx \tilde{\nabla}^i X^j + \tilde{\nabla}^j X^i - \frac{2}{3}\tilde{\nabla}_k X^k f^{ij}. \tag{76}$$

Once the conformally rescaled hydrodynamical conserved variables (q_D, q_{S_i}, q_τ) (for their definitions, see Section 2.1) are given, the metric can be solved by the following steps:

(i) Solve equation (72) for the vector potential X^i from the conserved variables q_{S_i}.
(ii) Calculate the tensor $\tilde{A}^{ij}$ in equation (76) from the vector potential X^i.
(iii) Solve equation (73) for the conformal factor ψ.
(iv) With the updated conformal factor ψ, calculate the conserved variables (D, S_i, τ) and thus convert the conserved variables to the primitive variables (ρ, Wv^i, P). Then $\tilde{S}$ can be worked out consistently.
(v) Solve equation (74) for the lapse function α.
(vi) Solve equation (75) for the shift vector β^i.

2.6.1 *Boundary conditions*

As in Cheong et al. (2020), in the simulations of spheric-like astrophysical systems (e.g. isolated neutron star and core-collapse supernova), we set the Schwarzschild solution as the outer boundary condition. In particular, we impose the following boundary conditions:

$$\left.\frac{\partial\psi}{\partial r}\right|_{r_{\max}} = \frac{1-\psi}{r}, \tag{77}$$

$$\left.\frac{\partial\alpha}{\partial r}\right|_{r_{\max}} = \frac{1-\alpha}{r}, \tag{78}$$

$$\left.\beta^i\right|_{r_{\max}} = 0, \tag{79}$$

$$\left.X^i\right|_{r_{\max}} = 0, \tag{80}$$

Note that due to the non-linearity of the scalar equations equation (73) and equation (74), instead of solving ψ and α directly, we solve for its deviation, e.g. $\delta_\psi \equiv \psi - 1$ as in Cheong et al. (2020) and Bucciantini & Del Zanna (2011). The boundary condition to equation (77) that we implemented in Gmunu for the equation of the conformal factor ψ is

$$\frac{\partial}{\partial r}(r\delta_\psi) = 0. \tag{81}$$

In spherical coordinates (r, θ, ϕ), the implementation of this Robin boundary condition equation (81) on the cell-face is straightforward. However, this is not the case when we are working in Cartesian coordinates (x, y, z) or cylindrical coordinates (R, z, φ). In these particular cases, we define the outer boundary at the outermost cell-centre, the boundary condition equation (81) can then be implemented as

$$\begin{cases} \delta_\psi + x\frac{\partial\delta_\psi}{\partial x} + y\frac{\partial\delta_\psi}{\partial y} + z\frac{\partial\delta_\psi}{\partial z} = 0 & \text{in Cartesian coordinates (x,y,z),} \\ \delta_\psi + R\frac{\partial\delta_\psi}{\partial R} + z\frac{\partial\delta_\psi}{\partial z} = 0 & \text{in cylindrical coordinates (R,z,}\varphi\text{).} \end{cases} \tag{82}$$

2.6.2 *Frequency of solving the metric*

In most of the applications, the metric quantities do not change too rapidly with time, it is in general not necessary to solve the metric equations at every time step in order to reduce the computational time. In practice, the metric equations are solved for every Δn time-steps, and extrapolation could also be used to obtain the metric quantities in between, e.g. Dimmelmeier et al. (2002). The number of time-steps between solving the metric Δn could vary from case to case, typically vary from 10 to 100 in spherical coordinates (Dimmelmeier et al. 2002; Dimmelmeier, Stergioulas & Font 2006; Bucciantini & Del Zanna 2011; Cheong et al. 2020). In our previous study (Cheong et al. 2020), we found that $\Delta n \sim 50$ is sufficient to extract the oscillation modes of isolated neutron stars correctly, and extrapolation has negligible effects on the results. Since the time step Δt is usually determined by Courant–Friedrichs–Lewy condition (see Section 2.9), we empirically found that the choice of the number of time steps between solving the metric $\Delta n \sim 10$–50 usually works well for isolated neutron star simulations even in Cartesian coordinates (x, y, z) or cylindrical coordinates (R, z, φ).

For more dynamical situations, the metric variation time-scale may differ from time to time, keeping Δn fixed during the entire simulation may be problematic. For instance, the evolution of the metric is not correct if Δn is too large while the computational power is wasted if Δn is too small. To have an adaptive Δn, we use the metric equation of ψ (equation 73) to monitor numerical errors. In particular, at each time-step, the metric will be updated if the L^∞ norm of the residual of equation (73) (i.e. the maximum value of the absolute of equation 73) below a threshold $\epsilon_{\rm residual}$. It is worth to point out that, the metric equation of ψ (equation 73) is actually originated from Hamiltonian constraint equation (see e.g. Shibata 2015), which is widely used to monitor numerical errors in dynamical numerical relativity simulations. We experimentally found out that, with this approach only (i.e. set Δn as an extremely large number), the choice of $\epsilon_{\rm residual} = 10^{-3}$ (the tolerance of the metric solver is typically set as 10^{-6}) is sufficient to obtain correct results in both stable neutron star evolution and migration tests (e.g. see Section 3.3).

Unless explicitly stated, in this work, we set $\Delta n = 50$, $\epsilon_{\rm residual} = 10^{-3}$ and the tolerance of the metric solver is set as 10^{-6}. That is, the metric variables are updated at every $\leq$50 time-steps, and keep them fixed in between.

2.7 Non-linear cell-centred multigrid solver

To solve the *elliptical* metric equations (72)–(75), as in the previous version of Gmunu, we use the non-linear cell-centred multigrid (CCMG) elliptic solver (Cheong et al. 2020). Since the current version of Gmunu is developed on top of MPI-AMRVAC 2.0 framework (Xia et al. 2018; Keppens et al. 2020), it is natural to couple Gmunu to the existing open-source geometric multigrid library octree-mg[1] (Teunissen & Keppens 2019). This library is parallelized with MPI, supports coupling with quadtree/octree AMR grids and provides Dirichlet, Neumann and periodic boundary conditions.

However, the library has its limitations, e.g. polar and spherical grids are not supported, supports only simple and non-varying source terms and has no Robin boundary conditions and thus cannot be applied directly on the metric equations or on spherical polar coordinates. Although the convergence rate is reduced when using point-wise smoothers directly on spherical polar/3D-cylindrical coordinates (Briggs, Henson & McCormick 2000), in the current implementation, we still adopt point-wise smoothers, and extend the library based on our previous implementation (Cheong et al. 2020) so that the extended multigrid library can be applied to solve the elliptical metric equations on cylindrical and spherical coordinates. The extension of supporting curvilinear coordinates also benefits us when handling divergenceless constraint of the magnetic field in different geometries. In the following, we outline the key elements of our multigrid solver.

2.7.1 Cell-centred discretization and operators

To solve a non-linear elliptic equation $\mathcal{L}(u) = f$, where $\mathcal{L}$ is an elliptic operator, u is the solution, and f is the source term, we can discretize the equation on a grid with resolution h as

$$\mathcal{L}_h(u_h) = f_h, \tag{83}$$

where all the solution u_h and the right-hand side f_h are defined at the cell centres. The elliptic operators are discretized with a standard 5/7-point (in 2D/3D) second-order accurate discretization. These operators contain Laplacian operators, first- and second-order derivatives etc. Here, we list some discretized operators used in Gmunu.

The Laplacian of a scalar function $u(x, y, z)$ in Cartesian coordinate is

$$\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}, \tag{84}$$

which is discretized as

$$\begin{aligned}(\nabla^2 u)_{\tt i,j,k} := &\frac{1}{\Delta x^2}(u_{\tt i+1,j,k} - 2u_{\tt i,j,k} + u_{\tt i-1,j,k}) \\ &+ \frac{1}{\Delta y^2}(u_{\tt i,j+1,k} - 2u_{\tt i,j,k} + u_{\tt i,j-1,k}) \\ &+ \frac{1}{\Delta z^2}(u_{\tt i,j,k+1} - 2u_{\tt i,j,k} + u_{\tt i,j,k-1}),\end{aligned} \tag{85}$$

where Δx, Δy, and Δz are the grid spacing in the x, y, and z directions. On the other hand, the Laplacian of a scalar function $u(R, z, \varphi)$ in cylindrical coordinate is

$$\nabla^2 u = \frac{1}{R}\frac{\partial}{\partial R}\left(R\frac{\partial u}{\partial R}\right) + \frac{\partial^2 u}{\partial z^2} + \frac{1}{R^2}\frac{\partial^2 u}{\partial \varphi^2}, \tag{86}$$

which is discretized as

$$\begin{aligned}(\nabla^2 u)_{\tt i,j,k} := &\frac{1}{R_{\tt i,j,k}}\frac{1}{\Delta R}\left(R_{\tt i+1/2,j,k}\frac{u_{\tt i+1,j,k} - u_{\tt i,j,k}}{\Delta R}\right. \\ &\left. - R_{\tt i-1/2,j,k}\frac{u_{\tt i,j,k} - u_{\tt i-1,j,k}}{\Delta R}\right) \\ &+ \frac{1}{\Delta z^2}\left(u_{\tt i,j+1,k} - 2u_{\tt i,j,k} - u_{\tt i,j-1,k}\right) \\ &+ \frac{1}{R^2_{\tt i,j,k}}\frac{1}{\Delta\varphi^2}\left(u_{\tt i,j,k+1} - 2u_{\tt i,j,k} - u_{\tt i,j,k-1}\right).\end{aligned} \tag{87}$$

Finally, in spherical coordinate, the Laplacian of a scalar function $u(r, \theta, \phi)$ reads

$$\nabla^2 u = \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial u}{\partial r}\right) + \frac{1}{r^2\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial u}{\partial\theta}\right) + \frac{1}{r^2\sin^2\theta}\frac{\partial^2 u}{\partial\phi^2}, \tag{88}$$

which is discretized as

$$\begin{aligned}(\nabla^2 u)_{\tt i,j,k} := &\frac{1}{r^2_{\tt i,j,k}}\frac{1}{\Delta r}\left(r^2_{\tt i+1/2,j,k}\frac{u_{\tt i+1,j,k} - u_{\tt i,j,k}}{\Delta r}\right. \\ &\left. - r^2_{\tt i-1/2,j,k}\frac{u_{\tt i,j,k} - u_{\tt i-1,j,k}}{\Delta r}\right) \\ &+ \frac{1}{r^2_{\tt i,j,k}\sin\theta_{\tt i,j,k}}\frac{1}{\Delta\theta} \\ &\times\left(\sin\theta_{\tt i,j+1/2,k}\frac{u_{\tt i,j+1,k} - u_{\tt i,j,k}}{\Delta\theta}\right. \\ &\left. - \sin\theta_{\tt i,j-1/2,k}\frac{u_{\tt i,j,k} - u_{\tt i,j-1,k}}{\Delta\theta}\right) \\ &+ \frac{\left(u_{\tt i,j,k+1} - 2u_{\tt i,j,k} - u_{\tt i,j,k-1}\right)}{r^2_{\tt i,j,k}\sin^2\theta_{\tt i,j,k}\Delta\phi^2}.\end{aligned} \tag{89}$$

The first- and second-order derivatives are discretized as, for example,

[1]As the authors did not name their code in Teunissen & Keppens (2019), here we use the name of the git repository, octree-mg, as the name of the library.

$$\left(\frac{\partial u}{\partial x}\right)_{\mathtt{i,j,k}} = \frac{u_{\mathtt{i+1,j,k}} - u_{\mathtt{i-1,j,k}}}{2\Delta x}, \tag{90}$$

$$\left(\frac{\partial^2 u}{\partial x^2}\right)_{\mathtt{i,j,k}} = \frac{u_{\mathtt{i+1,j,k}} - 2u_{\mathtt{i,j,k}} + u_{\mathtt{i-1,j,k}}}{\Delta x^2}, \tag{91}$$

$$\left(\frac{\partial^2 u}{\partial x \partial y}\right)_{\mathtt{i,j,k}} = \frac{u_{\mathtt{i+1,j+1,k}} - u_{\mathtt{i+1,j-1,k}} - u_{\mathtt{i-1,j+1,k}} + u_{\mathtt{i-1,j-1,k}}}{4\Delta x \Delta y}. \tag{92}$$

Note that the diagonal ghost cells (i.e. layers of cells around every grid blocks, which is used to contain data from neighbouring blocks for parallel communication) are not passed when different processors are communicating as in Teunissen & Keppens (2019), to calculate some mixed differentiation such as $\frac{\partial^2}{\partial x \partial y}$ in equations (72) and (75) without a large amount of communication between processors, at the *block corner*, we adopt the following discretization which requires not all diagonal elements:

$$\left(\frac{\partial^2 f}{\partial x \partial y}\right)_{\mathtt{i,j,k}} \approx \frac{1}{2\Delta x_{\mathtt{i}} \Delta y_{\mathtt{j}}}(-f_{\mathtt{i+1,j-1,k}} - f_{\mathtt{i-1,j+1,k}} + f_{\mathtt{i+1,j,k}} + f_{\mathtt{i-1,j,k}} + f_{\mathtt{i,j+1,k}} + f_{\mathtt{i,j-1,k}} - 2f_{\mathtt{i,j,k}}). \tag{93}$$

2.7.2 *Smoothers and solvers*

Relaxation method can be used as a smoother since it smooths the error in the solution. In this work, Gauss–Seidel type point-wise smoothers is included, in which the solution $u_{\mathtt{i,j,k}}$ is solved while keeping the neighbours values fixed. To deal with the case where the operator $\mathcal{L}$ is non-linear in u, instead of implementing the traditional Gauss–Seidel iteration, we implement a Newton Gauss–Seidel iteration (Press et al. 1992):

$$u^{\text{new}}_{\mathtt{i,j,k}} = u^{\text{old}}_{\mathtt{i,j,k}} - \left(\mathcal{L}(u^{\text{old}}_{\mathtt{i,j,k}}) - f_{\mathtt{i,j,k}}\right) \Big/ \left(\left.\frac{\partial \mathcal{L}}{\partial u_{\mathtt{i,j,k}}}\right|_{u=u^{\text{old}}_{\mathtt{i,j,k}}}\right). \tag{94}$$

Note that equation (94) reduces to the standard Gauss–Seidel iteration if $\mathcal{L}$ is linear in u.

The update ordering of the indices `i,j,k` affects the smoothing behaviour. Two orderings are available:

(i) Linear ordering: all the indices `i,j,k` are looped linearly (in the order they are stored in the computer's memory);

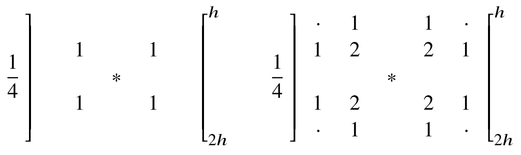
(ii) Red-black ordering: also known as odd-even ordering, the solution at the points where `i+j+k` is even will be updated first, and then update the rest where the points `i+j+k` is odd.

As mentioned, while the convergence rate is reduced when using point-wise smoothers directly on spherical polar or 3D cylindrical coordinates (Briggs et al. 2000), in the current implementation, we still adopt point-wise smoothers for all coordinates.

Although the computational cost for solving elliptic equations at the coarsest grid is low even with other robust direct solver, for simplicity, in this work, the Newton Gauss–Seidel relaxation is used as a direct solver.

2.7.3 *Intergrid transfer operators: Prolongation and restriction*

Grids at different levels are connected by inter grid transfer operators. The operators that map the values from a fine to a coarse grid are called *restriction* and the mapping from a coarse to the fine grid are called *prolongation*. Although there are many possible choices of restriction and prolongation operators, they cannot be chosen arbitrarily (Mohr & Wienands 2004). Kwak prolongation Kwak (1999) is adopted in this work, the data at finer grids (in 2D, for example) can be written as



(a) piecewise constant (b) Kwak (1999)

Figure 1. The stencil notation of the interpolation operators in 2D. The '$*$' denotes the location of the coarse grid node. The notation shows the weighting of the value which are the neighbours of the coarse grid node '$*$'. Kwak interpolations here is second-order accurate while the price-wise constant here is first-order accurate.

$$u_{x+\Delta x/4, y+\Delta y/4} = \frac{1}{4}(2u_{x,y} + u_{x+\Delta x,y} + u_{x,y+\Delta y}),$$
$$u_{x-\Delta x/4, y+\Delta y/4} = \frac{1}{4}(2u_{x,y} + u_{x-\Delta x,y} + u_{x,y+\Delta y}). \tag{95}$$

This can also be shown in the stencil notation, as shown in Fig. 1b. The communication costs can be saved significantly since this prolongation requires no diagonal ghost cells, thus, this is used by default. On the other hand, for the restriction, we adopt the first-order accurate piecewise constant restriction (Fig. 1a). It is worth to point out that the choice of restriction and prolongation operators affect the convergence rate but not the order of accuracy of the solution, the latter is determined by the accuracy of the smoothers and solvers.

2.8 Adaptive mesh refinement

As mentioned, the simulation code is required to resolve different scales accurately since most of astrophysical systems include multi-time scale and multilength scale physics.

Adaptive mesh refinement (AMR) algorithms enable us to resolve different length-scale accuracy and significantly reduce the computational costs. In particular, AMR algorithms change the grid spacing and the structure of the computational domains during the numerical calculations. One of the widely adopted AMR algorithms in grid-based codes is the *patch-based* AMR, which is based on overlapping patches, was introduced by Berger & Oliger (1984). Some examples can be found, for example, Berger & Colella (1989), `PLUTO` (Mignone et al. 2007, 2012a,b), `Athena` (Stone et al. 2008), and `Enzo` (Bryan et al. 2014). Although the patch-based AMR strategy has been successfully applied in various astrophysical studies, it was found not to perform well on modern highly parallel architectures. Alternatively, the so-called *block-based* AMR (e.g. Stout et al. 1997) has great performance and scaling on parallel architectures, and the implementation of which is much simpler. Notable examples are `MPI-AMRVAC` (Xia et al. 2018; Keppens et al. 2020) and its sister code `BHAC` (Porth et al. 2017), `ECHO` (Zanotti & Dumbser 2015), an updated version of `Athena++` (Stone et al. 2020), `RAM` (Zhang & MacFadyen 2006), `FLASH` (Fryxell et al. 2010) which is based on `PARAMESH` AMR library (MacNeice et al. 2000), and a GPU-accelerated code called `GAMER` (Schive, Tsai & Chiueh 2010; Schive et al. 2018).

The parallelization and the adaptive mesh refinement module of current `Gmunu` are provided by coupling with `MPI-AMRVAC`

PDE toolkit (Xia et al. 2018; Keppens et al. 2020), a open-source Message Passing Interface (MPI) based parallelized toolkit with a pure block-tree (i.e. block-based quadtree-octree (in 2D–3D), as well as their 1D equivalent) AMR module. Since `MPI-AMRVAC` is a stand alone parallelized block-grid adaptive framework and provides user-defined physics interface modules, only minor modifications are needed to use this library. In this section, we briefly summarise the essential elements of the adaptive mesh refinement module from `MPI-AMRVAC`. For more details of implementations, we refer readers to Keppens et al. (2012).

2.8.1 *Block-tree AMR*

The computational domain is considered to be logically rectangular region, i.e. it is bounded by $[x^i_{\min}, x^i_{\max}]$ in each dimension $i \in \{1, \cdots, N_{\rm dim}\}$, where $N_{\rm dim}$ is the number of dimensions. The domain decomposition on the lowest grid level $l = 1$ is determined by specifying the total number of grid cells per dimension i, which is denoted as $N^i_{l=1}$, and the number of grid cells per dimension per block $N_{\rm grid}$, where $N^i_{l=1}$ must be an integer multiple of $N_{\rm grid}$. Note that the number of grid cells per dimension $N_{\rm grid}$ is independent of the grid level l and also the direction i. Based on this decomposition, by using the refinement strategies discussed in Section 2.8.3, the code check whether refinement is needed for each block at each level $l < l_{\max}$. When a block at level $l < l_{\max}$ is identified for refinement, $2^{N_{\rm dim}}$ child blocks with the resolution Δx^i_{l+1} are activated. Currently, the refinement ratios between grid levels is fixed to 2, namely, $\Delta x^i_{l+1} = \Delta x^i_l/2$. The newly generated blocks will be marked as 'active grid leafs' while their parent blocks will be removed from the active grid leafs. Obviously, the total number of active grid leafs $N_{\rm leaf}$ may change after the regridding.

Overall, the computational domain is decomposed into $N_{\rm block}$ blocks, each block contains $(N_{\rm grid})^{N_{\rm dim}}$ grid cells. The decomposition of the computational domain into blocks is arbitrary, and the total cell number must be the number of blocks times the block size. To minimize times for interprocessor communications and improve data locality, all the blocks are connected with a Morton-ordered space-filling curve (also known as Z-order curve) (Xia et al. 2018).

2.8.2 *Prolongation and restriction*

The prolongation–restriction formulae adopted in this work follow (van der Holst & Keppens 2007; Keppens et al. 2012), which can be used in curvilinear coordinate systems for which the Jacobian is separable (van der Holst & Keppens 2007). For simplicity, in this section, we express the formula for 2D case, which can be generalized straightforwardly to any dimensional cases. The prolongation adopted in this work (for two-dimensional case) is

$$\langle q\rangle^{l+1}_{\tt I,J} = \langle q\rangle^{l}_{\tt i,j} + \sum_{k=1}^{N_{\rm dim}=2} \overline{\Delta_k q}^{\,l}_{\tt i,j} \times 2\frac{x^{l+1}_{\tt I,J} - x^{l}_{\tt i,j}}{\Delta x^l_k} \times \left(1 - \frac{\Delta V^{l+1}_{\tt I,J}}{\sum_{(k)} \Delta V^{l+1}_{\tt I,J}}\right), \quad (96)$$

where `i,j` are the location indices for the variable $<q>$ at the level l while `I,J` is used at level $l+1$, and the volume summation indicated with $\sum_{(k)}$ sums up the 2 fine cell volumes along direction k, e.g. for $k = 2$, $\sum_{(k=2)} \Delta V^{l+1}_{\tt I,J} = \Delta V^{l+1}_{\tt I,J} + \Delta V^{l+1}_{\tt I,J+1}$. Here, $\overline{\Delta_k q}^{\,l}_{\tt i,j}$ denotes the slope limited linear reconstruction of $\langle q\rangle^l_{\tt i,j}$ along k direction. The Total-Variation-Diminishing (TVD) slope limiters such as minmod or MC limiters can be used.

For the restriction formula, since the coarse cell values $\langle q\rangle^l_{\tt i,j}$ can be obtained by using the prolongation formula (equation 96) as well, the prolongation formula is also used for restriction.

2.8.3 *Refinement criteria*

The criteria of controlling grid refinement significantly affects the efficiency and accuracy of adaptive mesh refinement calculations. With block-tree data structure, the needs of refining or coarsening are determined on a block-by-block basis. The regridding process for each block can be summarized as follows:

(i) Check weather the block level l is in the valid range, i.e. $1 \le l < l_{\max}$, where $l_{\max}$ is the maximal grid level;

(ii) At *each* grid point $\boldsymbol{x}$, compute the local error $E_{\boldsymbol{x}}$, and compare with a user-set tolerance ϵ_l;

(iii) If *any* point has the local error $E_{\boldsymbol{x}}$ larger than a user-set tolerance ϵ_l, namely, if $E_{\boldsymbol{x}} > \epsilon_l$, refine this block;

(iv) If *all* points has the local error below the user-set tolerance with a user-defined fraction $f^\epsilon_l < 1$, namely, if $E_{\boldsymbol{x}} < f^\epsilon_l \epsilon_l$, coarsen the block if the level l is larger than 1 ($l > 1$).

The key of the refinement criteria here is to compute the local error $E_{\boldsymbol{x}}$ at each point, and with a properly set tolerance ϵ_l and a fraction $f^\epsilon_l < 1$. Depending on different application, the local error needed to be estimated on various primitive or auxiliary variables. For example, consider a set of variables q_k, the final local error $E_{\boldsymbol{x}}$ are calculated by the formula

$$E_{\boldsymbol{x}} = \sum_k \sigma_k E^{\rm rel}_{\boldsymbol{x},k}, \quad (97)$$

where k is a variable index for q_k, $E^{\rm rel}_{\boldsymbol{x},k}$ is the local relative variable errors of variable q_k at grid $\boldsymbol{x}$, and σ_k is the corresponding weighting which is defined by users and obey $\sum_k \sigma_k = 1$. The local relative variable errors $E^{\rm rel}_{\boldsymbol{x},k}$ can be obtained by the so-called error estimators. In the following, we describe various possible estimators.

2.8.4 *Historical estimator*

Historical estimator requests the information of previous time steps such as t^{n-1} and t^n. Richardson extrapolation can be used to compute the local error by the following:

$$E^{\rm rel}_{\boldsymbol{x},k} = \frac{\left|q^{\rm CI}_k - q^{\rm IC}_k\right|}{\sum_k \sigma_k \left|q^{\rm IC}_k\right|}, \quad (98)$$

where $q^{\rm CI}_k$ is the coarsened-integrated solution which can be obtained by coarsening it at time $n-1$ from a Δx_i grid to a $2\Delta x_i$ grid, then time integrating it with time-step $2\Delta t^{n-1}_l$; while $q^{\rm IC}_k$ is the integrated-coarsened solution which can be obtained by time integrating $(q_k)^n_l$ with time-step Δt^{n-1}_l. To be specific, $q^{\rm CI}_k$ and $q^{\rm IC}_k$ can be obtained by

$$(q_k)^{n-1}_l \xrightarrow{\text{coarsening}} (q_k)^{n-1}_{2\Delta x_i} \xrightarrow[2\Delta t^{n-1}_l]{\text{advance}} q^{\rm CI}_k,$$
$$(q_k)^{n}_l \xrightarrow[\Delta t^{n-1}_l]{\text{advance}} q^{\rm IC}_k. \quad (99)$$

The integrator used in this Richardson-based estimator can be low-order (e.g. first-order), dimensionally unsplit, incorporating unsplit source terms.

A simpler and computationally cheaper variant of historical estimator is the local comparison of the solution between t^{n-1} and

t^n. For instance, the local error of variable q_k can be written as

$$E^{\rm rel}_{\boldsymbol{x},k} = \frac{|q_k^{n-1} - q_k^n|}{|q_k^{n-1}|}. \tag{100}$$

It is worth to point out that although these kinds of historical estimators work successfully in variety of test problems, relying on previous solution only may be insufficient for rapidly moving, strong shock cases. For more details, see discussions in Keppens et al. (2012).

2.8.5 *Löhner type estimator*

The Löhner's error estimator (Löhner 1987) was originally developed for finite-element simulations. This is computationally efficient since it requests the 'current' (t^n) solution only. The multidimensional generalization is given by

$$E^{\rm rel}_{\boldsymbol{x},k} = \sqrt{\frac{\sum_p \sum_q \left(\frac{\partial^2 q_k}{\partial x_p \partial x_q} \Delta x_p \Delta x_q \right)^2}{\sum_p \sum_q \left[\left(|\frac{\partial q_k}{\partial x_p}|_{\boldsymbol{x}+\Delta x_p} + |\frac{\partial q_k}{\partial x_p}|_{\boldsymbol{x}-\Delta x_p} \right) \Delta x_p + f_{{\rm wave},l} |\bar{q}_k|_{p,q} \right]^2}}, \tag{101}$$

where the indices p and q run over all dimensions $p, q = 1, \cdots, N_D$. The last term in the denominator $f_{{\rm wave},l}|\bar{q}_k|_{p,q}$ prevent refinement of small ripples, where $f_{{\rm wave},l}$ is the level dependent 'wave-filter' parameter which is typically chosen of the order 10^{-2}. $|\bar{q}_k|_{p,q}$ here is an average value computed from surrounding corners,

$$|\bar{q}_k|_{p,q} := |q_k|_{\boldsymbol{x}+\Delta x_p+\Delta x_q} + 2|q_k|_{\boldsymbol{x}} + |q_k|_{\boldsymbol{x}+\Delta x_p+\Delta x_q}. \tag{102}$$

Unless explicitly stated, the error is estimated based on the density ρ only by using the Löhner's error estimator when adaptive mesh refinement module is activated.

2.9 Time stepping

To evolve the hyperbolic partial differential equations stably, the size of the time-step must satisfy the Courant–Friedrichs–Lewy (CFL) conditions (Courant, Friedrichs & Lewy 1928). Physically, this conditions ensure that the propagation speed of any travelling wave is always smaller than the 'numerical speed' $\sim\Delta x/\Delta t$.

Before we present how we determine the time-step Δt, we briefly outline the block-tree adaptive mesh refinement data structure as well as the notations we use here. As discussed in Section 2.8.1, in Gmunu, the whole computational domain is decomposed into $N_{\rm leaf}$ active grid leafs, each of them contains $N_{\rm block}$ blocks, each block contains $(N_{\rm grid})^{N_{\rm dim}}$ grid cells, where $N_{\rm grid}$ is the number of grids per dimension and $N_{\rm dim}$ is the number of dimensions. In addition, to simplify the implementation significantly, the same time-step Δt is assigned for all levels. In the following, $1 \le i \le N_{\rm dim}$ denotes the dimension, $1 \le j \le N_{\rm block}$ denotes the block index and finally $1 \le k \le N_{\rm leaf}$ denotes the leaf index.

In Gmunu, the time-step Δt is given by

$$\Delta t = \min_{1\le k\le N_{\rm leaf}} (c_{\rm CFL}\tau_k), \tag{103}$$

where $c_{\rm CFL}$ is the Courant–Friedrichs–Lewy factor with range (0, 1] and typically below 0.9; τ_k is the unrestricted time step at the k-th leaf. In Gmunu, there are three possible ways to evaluate the unrestricted time-step τ_k, they are 'minimun', 'summax', and 'maxsum':

$$\tau_k^{-1} = \begin{cases} \max\limits_{1\le i\le N_{\rm dim}} \left(\max\limits_{1\le j\le N_{\rm block}} \left(\frac{c_i^{\max}}{\Delta_i} \right) \right), & \text{minimum;} \\ \sum\limits_{i=1}^{N_{\rm dim}} \left(\max\limits_{1\le j\le N_{\rm block}} \left(\frac{c_i^{\max}}{\Delta_i} \right) \right), & \text{summax;} \\ \max\limits_{1\le j\le N_{\rm block}} \left(\sum\limits_{i=1}^{N_{\rm dim}} \frac{c_i^{\max}}{\Delta_i} \right), & \text{maxsum (default case),} \end{cases} \tag{104}$$

where $c_i^{\max}$ is the maximal signal propagation speed (at cell-centres) in the i direction, usually the maximum of the absolute value of eigenvalue $\max\limits_i(|\lambda_i|)$ (see Section 2.3) is used. In addition, Δ_i here is the spatial step size in direction i. For example, in Cartesian coordinates (x, y, z), Δ_i are simply $(\Delta x, \Delta y, \Delta z)$. It is worth to point out that, in spherical-polar coordinates (r, θ, ϕ), the corresponding Δ_i are $(\Delta r, r\Delta\theta, r\sin\theta\Delta\phi)$, the Courant–Friedrichs–Lewy become rigorous in multidimensional cases at the centre ($r \to 0$) or at the pole ($\theta \to 0, \pi$) in 3D cases. Many approaches proposed to deal with the rigorous time-step constraint in spherical-polar coordinates (see Müller 2020 and references therein). In Gmunu, we made use of adaptive mesh refinement (see Section 2.8), the grids are enforced to be coarsened to keep $r\Delta\theta \sim \Delta r$ when r is small and similarly, we require $r\sin\theta\Delta \sim \Delta r$ as $\theta \to 0$ or $\theta \to \pi$.

3 NUMERICAL TESTS

In the remainder of this paper, we present a selection of representative test problems with our code. The tests range from special relativistic (magneto-)hydrodynamics to general relativistic (magneto-) hydrodynamics, from one to multiple dimensions and in Cartesian, cylindrical, and spherical coordinates. Unless otherwise specified, all simulations reported in this paper were performed with TVDLF approximate Riemann solver, 5th-order reconstruction method MP5 and SSPRK3 for the time integration.

3.1 Special relativistic hydrodynamics

3.1.1 *Two-dimensional smooth problem*

A smooth test for relativistic hydrodynamics code proposed in He & Tang (2012), which describes a wave propagating in a 2D space, is well suited for checking the order of accuracy of a numerical code at the smooth part. In particular, we perform the simulations with Cartesian coordinates (x, y) on a flat space–time, the computational domain covers the region $0 \le x \le 3/\sqrt{2}$ and $0 \le y \le 2$. The initial condition is given as

$$\rho = 1 + A\sin[2\pi(x\cos\theta + y\sin\theta)], \tag{105}$$

$$p = 1, \tag{106}$$

$$v_x = v_0, \qquad v_y = 0, \tag{107}$$

where the wave is propagating at an angle $\theta = \pi/6$ relative to the horizontal axis, $A = 0.2$ and $v_0 = 0.2$. We consider an ideal-gas equation of state $p = (\Gamma - 1)\rho\epsilon$ with $\Gamma = 5/3$. This problem has the exact solution

$$\rho = 1 + A\sin[2\pi((x\cos\theta + y\sin\theta) - (v_x\cos\theta + v_y\sin\theta)t)], \tag{108}$$

$$p = 1, \tag{109}$$

$$v_x = v_0, \qquad v_y = 0. \tag{110}$$

Table 1. Numerical errors and convergence rates of the 2D relativistic hydrodynamics smooth problem at $t = 2$. In this test, 2nd order accurate strong-stability preserving Runge–Kutta (SSPRK2) time integrator, Harten, Lax and van Leer (HLL) Riemann solver (Harten et al. 1983) with 2nd order Montonized central (MC) limiter (van Leer 1974) are used. As expected, the second-order convergence is achieved with this setting.

N	δ_N	R_N
32	1.859E−3	–
64	4.839E−4	1.94
128	1.246E−4	1.96
256	3.120E−5	2.00
512	7.692E−6	2.02
1024	1.896E−6	2.02
2048	4.690E−7	2.02

The discretization of the computational domain is set to be [N, $2N$] with an integer N which controls the resolution. In this test, we use 2nd order accurate strong-stability preserving Runge–Kutta (SSPRK2) time integrator, Harten, Lax and van Leer (HLL) Riemann solver Harten, Lax & Leer (1983) with 2nd order Montonized central (MC) limiter (van Leer 1974).

To quantify the convergence rate at $t = 2$, we define the relative numerical errors δ_N as

$$\delta_N = \frac{\sum_i^N \sum_j^N |\rho^N_{i,j} - \rho^{\text{exact}}_{i,j}|}{\sum_i^N \sum_j^N |\rho^{\text{exact}}_{i,j}|}, \tag{111}$$

and the convergence rate R_N can be obtained by

$$R_N = \log_2\left(\frac{\delta_{N/2}}{\delta_N}\right). \tag{112}$$

Table 1 shows the numerical errors and convergence rates of this problem at $t = 2$ at different resolution N. The convergence rate can be virtually present with a numerical-errors-versus-resolution plot, as shown in the Fig. 2. As expected, the second-order convergence is achieved with this setting.

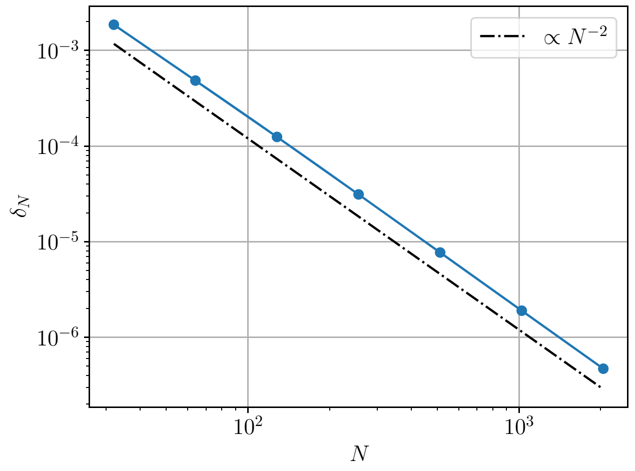


Figure 2. Numerical errors versus resolution (blue line) for the 2D relativistic hydrodynamics smooth problem. Second-order ideal scaling is given by the dashed black line. In this test, 2nd order accurate strong-stability preserving Runge–Kutta (SSPRK2) time integrator, Harten, Lax and van Leer (HLL) Riemann solver (Harten et al. 1983) with 2nd order Montonized central (MC) limiter (van Leer 1974) are used. As expected, the second-order convergence is achieved with this setting.

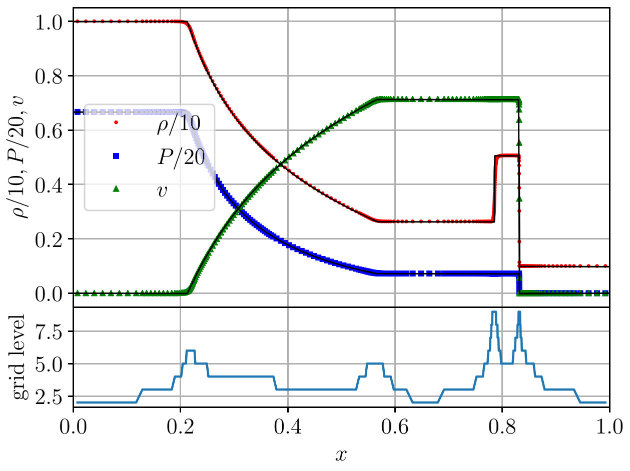


Figure 3. The upper panel shows the density (red dots), pressure (blue squares), and velocity (green triangles) profile at $t = 0.4$ for the relativistic shocktube test problem. The solid lines are the analytical solutions. The numerical results obtained by Gmunu agree with the analytical solutions. The lower panel shows the grid-level at different location of the computational domain. The grid-level is higher to provide finer resolution when the density is sharper.

3.1.2 *Relativistic shock tubes*

We follow Martí & Müller (2003) in this 1D shock tube problem. In particular, we perform the simulation with Cartesian coordinates on a flat space–time. Instead of simulating this problem with a uniform grid, we activate the block-based AMR module in this case. For instance, the computational domain covers the region $0 \le x \le 1$ with 16 base grid points and allows for 10 AMR levels (i.e. an effective resolution of 8192). The initial condition is given as

$$(\rho, p, v^x) = \begin{cases} (10, 40/3, 0) & \text{if } x < 0.5, \\ (1, 0, 0) & \text{if } x > 0.5. \end{cases} \tag{113}$$

We consider an ideal-gas equation of state $p = (\Gamma - 1)\rho\epsilon$ with $\Gamma = 5/3$. The upper panel of the Fig. 3 shows the comparison between the numerical results and the analytical solutions for the density, pressure, and velocity profiles at $t = 0.4$. The figure shows that our numerical results agree with the analytical solutions. The lower panel shows the grid-level at different location of the computational domain. The grid-level is higher to provide finer resolution when the density is sharper.

3.1.3 *2D Riemann problem*

To test how Gmunu works in 2D Cartesian coordinates, we picked a demanding highly relativistic 2D Riemann problem (Del Zanna & Bucciantini 2002). Here, we follow the modified version of this test presented in Mignone, Plewa & Bodo (2005), in which elementary waves are introduced at every interface. The initial condition is given as

$$(\rho, p, v^x, v^y) = \begin{cases} (\rho_1, p_1, 0, 0) & \text{if } x > 0, y > 0, \\ (0.1, 1, 0.99, 0) & \text{if } x < 0, y > 0, \\ (0.5, 1, 0, 0) & \text{if } x < 0, y < 0, \\ (0.1, 1, 0, 0.99) & \text{if } x > 0, y < 0, \end{cases} \tag{114}$$

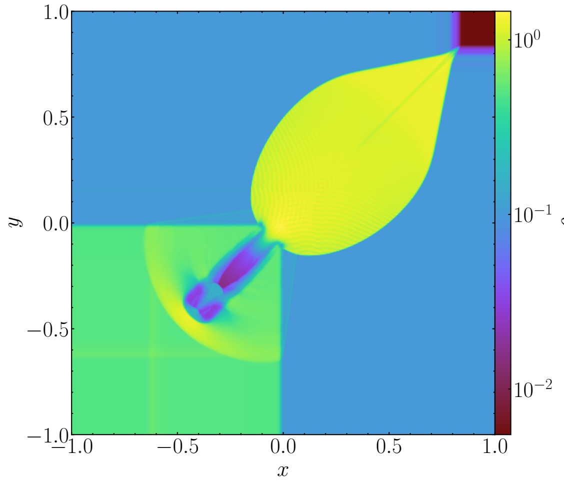


Figure 4. The density profile of the 2D relativistic Riemann problem (Mignone et al. 2005) at time $t = 0.8$. The result agrees qualitatively with Mignone et al. (2005).

where $\rho_1 = 5.477\,875 \times 10^{-3}$, $p_1 = 2.762\,987 \times 10^{-3}$. Here, we consider an ideal-gas equation of state $p = (\Gamma - 1)\rho\epsilon$ with $\Gamma = 5/3$. This test is run with a uniform grid 512×512 which covers the region $[-1, 1]$ for both x and y. Fig. 4 shows the density profile at $t = 0.8$. Gmunu is able to evolve this demanding test without crashing the code.

3.1.4 2D axisymmetric jet in cylindrical geometry

We study the propagation of a 2D axisymmetric relativistic jet in cylindrical coordinates. Not only would we like to test if Gmunu works properly in cylindrical geometry, to test the code's robustness, we simulated the model C2 in Martí et al. (1997), which contains strong relativistic shocks, instabilities, and shear flows and is highly supersonic. The computational domain covers $0 \le r \le 15$ and $0 \le z \le 45$ with resolution 512×1536. Initially, the jet is configured in the region $r \le 1$ and $z \le 1$ with density $\rho_{\rm b} = 1 \times 10^{-2}$, pressure $p_{\rm b} = 1.70305 \times 10^{-4}$, the velocity along z-axis $v_z = v_{\rm b} = 0.99c$ (which corresponds to a Lorentz factor ~ 7). Here, we consider the ideal-gas equation of state with $\Gamma = 5/3$. The rest of the computational domain is filled with an ambient medium with density $\rho_{\rm m} = 1$, pressure $p_{\rm m} = p_{\rm b}$, and zero velocity. We apply reflecting boundary conditions at the symmetric axis while the out-going boundary conditions were applied at all outer boundaries except that we keep the value unchanged inside the jet inlet $z = 0$, $r < 1$. In this test, we use 3rd order reconstruction method PPM.

Fig. 5 shows the density distribution of the axisymmetric jet at $t = 100$. As shown in Fig. 5, an expanding bow shock is formed and the Kelvin–Helmholtz instability is developed. The key structures of the jet, e.g. the head location, the shape of the bow shock and the development of the Kelvin–Helmholtz instability all agree with Martí et al. (1997).

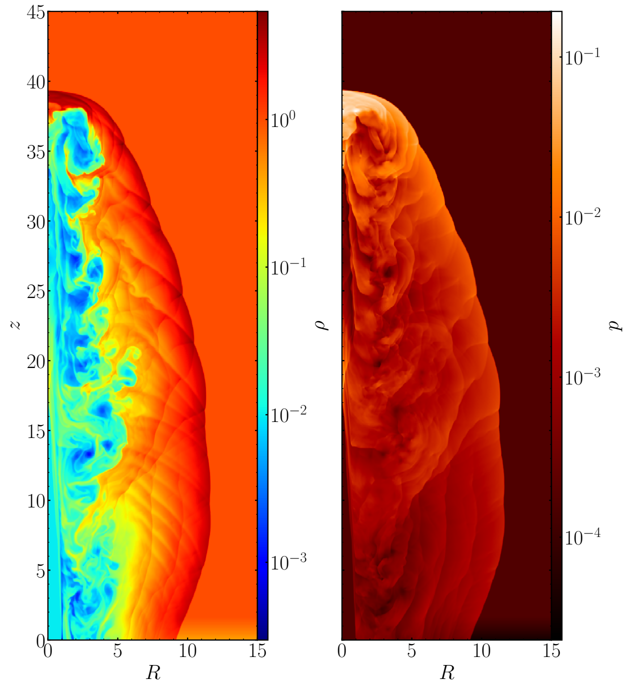


Figure 5. Density distribution (*left-hand panel*) and the pressure (*right-hand panel*) of the axisymmetric jet model C2 in Martí et al. (1997) at $t = 100$. The jet material interacts with the ambient medium and forms an expanding bow shock and develops the Kelvin–Helmholtz instability. The key structures of the jet, e.g., the head location, the shape of the bow shock and the development of the Kelvin–Helmholtz instability agrees with Martí et al. (1997).

3.2 Special relativistic magnetohydrodynamics

3.2.1 Large-amplitude circularly polarized Alfvén waves

Large-amplitude circularly polarized Alfvén waves test was first proposed in Del Zanna et al. (2007). This test describes the propagation of circularly polarized Alfvén waves with large-amplitude along a uniform background magnetic field $\boldsymbol{B}_0$. Here, by following (Mösta et al. 2014), we perform the simulations with 1D Cartesian coordinates x on a flat space–time, the computational domain covers the region $0 \le x \le 1$ with periodic boundary condition. The initial condition is given as

$$\rho = 1.0,\ p = 0.5 \tag{115}$$

$$v^x = 0, \quad v^y = -v_A A_0 \cos(kx), \quad v^z = -v_A A_0 \sin(kx), \tag{116}$$

$$B^x = B_0, \quad B^y = A_0 B_0 \cos(kx), \quad B^z = A_0 B_0 \sin(kx), \tag{117}$$

where $k = 2\pi/L_x$ is the wave vector, $B_0 = 1$ is the constant magnetic field for B^x, $A_0 = 1$ is the amplitude parameter and finally the square of the Alfvén speed v_A^2 can be expressed as

$$v_A^2 = \frac{2B_0^2}{\rho h + B_0^2\left(1 + A_0^2\right)} \times \left[1 + \sqrt{1 - \left(A_0^2 \frac{2B_0^2}{\rho h + B_0^2\left(1 + A_0^2\right)}\right)^2}\right]^{-1}. \tag{118}$$

We consider an ideal-gas equation of state $p = (\Gamma - 1)\rho\epsilon$ with $\Gamma = 5/3$. The simulation is allowed up to $t = T = 2$ (one period). The discretization of the computational domain is set to be N with an integer N which controls the resolution. In this test, we use 2nd order accurate strong-stability preserving Runge–Kutta (SSPRK2) time integrator, Harten, Lax and van Leer (HLL) Riemann solver Harten et al. (1983) with 2nd order Montonized central (MC) limiter (van Leer 1974).

Table 2. Numerical errors and convergence rates of the large-amplitude circularly polarized Alfvén waves test at $t = 2$. In this test, 2nd order accurate strong-stability preserving Runge–Kutta (SSPRK2) time integrator, Harten, Lax and van Leer (HLL) Riemann solver Harten et al. (1983) with 2nd order Montonized central (MC) limiter (van Leer 1974) are used. As expected, the second-order convergence is achieved with this setting.

N	$\lvert\lvert B^z(t=2) - B^z(t=0)\rvert\rvert_1$	R_N
32	1.859E−3	–
64	4.839E−4	2.32
128	1.246E−4	2.11
256	3.120E−5	2.04
512	7.692E−6	2.02
1024	1.896E−6	2.01
2048	4.690E−7	2.01

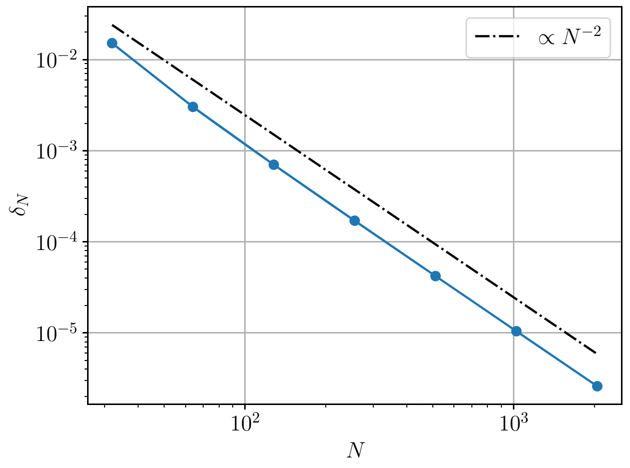


Figure 6. Numerical errors versus resolution (blue line) for the 1D ideal relativistic-magnetohydrodynamics Alfvén wave problem. Second-order ideal scaling is given by the dashed black line. In this test, 2nd order accurate strong-stability preserving Runge–Kutta (SSPRK2) time integrator, Harten, Lax and van Leer (HLL) Riemann solver (Harten et al. 1983) with 2nd order Montonized central (MC) limiter (van Leer 1974) are used. As expected, the second-order convergence is achieved with this setting.

To quantify the convergence rate at $t = 2$, we compute the L_1-norm of the difference of the difference between the initial and final values of the z-component of the magnetic field B^z as

$$||B^z(t=2) - B^z(t=0)||_1 := \frac{\sum_i |B^z(t=2) - B^z(t=0)|\Delta V_i}{\sum_i \Delta V_i}, \tag{119}$$

and the convergence rate R_N follows equation (112).

Table 2 shows the L_1-norm of the difference of the difference between the initial and final values of the z-component of the magnetic field B^z $||B^z(t=2) - B^z(t=0)||_1$ and convergence rates of this problem at $t = 2$ at different resolution N. The convergence rate can be virtually present with a numerical-errors-versus-resolution plot, as shown in the Fig. 6. As expected, the second-order convergence is achieved with this setting.

3.2.2 *Relativistic shock tubes*

Similar to relativistic hydrodynamics, there are shock tube tests in MHD. We follow Balsara (2001) in this 1D shock tube problem. In

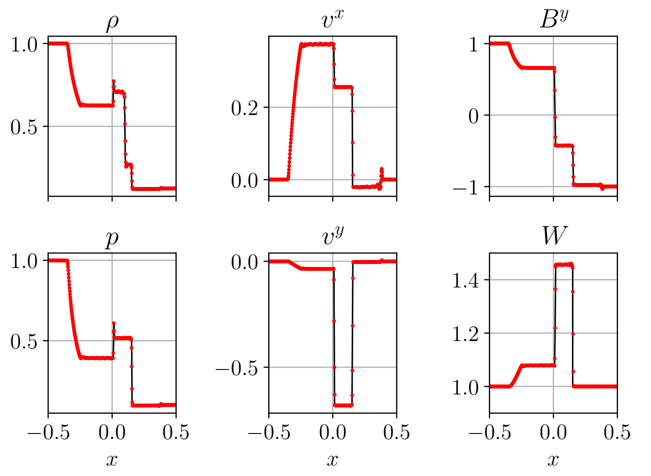


Figure 7. The density ρ (upper left), pressure p (lower left), velocity components v^x (upper middle), and v^y (lower middle), the y-component of the magnetic field B^y (upper right) and the Lorentz factor W (lower right) for the shock tube test at $t = 0.4$. The red dots show the numerical results obtained by Gmunu, which agree with the reference solutions (black solid lines) (Balsara 2001).

particular, we perform the simulation with Cartesian coordinates on a flat space–time. The initial condition is given as

$$(\rho, p, B^x, B^y) = \begin{cases} (1, 1, 0.5, 1) & \text{if } x < 0, \\ (0.125, 0.1, 0.5, -1) & \text{if } x > 0. \end{cases} \tag{120}$$

We consider an ideal-gas equation of state $p = (\Gamma - 1)\rho\epsilon$ with $\Gamma = 2$.

Fig. 7 compares the numerical results obtained by Gmunu (red dots) with the reference solutions (black solid lines) (Balsara 2001) at $t = 0.4$. It illustrates the shock-capturing ability of Gmunu and the results agree with the reference results.

3.2.3 *Cylindrical blast wave*

The cylindrical blast wave is a well-known difficult multidimensional SRMHD test problem. This problem describes an expanding blast wave in a plasma with an initially uniform magnetic field. Here, we follow the parameters presented in Komissarov (1999). The initial condition of this test problem is determined with radial parameters r_{in} and r_{out}. The density (and also the pressure, in the same form) profile is given by

$$\rho(r) = \begin{cases} \rho_{\text{in}} & \text{if } r \le r_{\text{in}}, \\ \exp\left[\frac{(r_{\text{out}}-r)\ln\rho_{\text{in}} + (r-r_{\text{in}})\ln\rho_{\text{out}}}{r_{\text{out}}-r_{\text{in}}}\right] & \text{if } r_{\text{in}} \le r \le r_{\text{out}}, \\ \rho_{\text{out}} & \text{if } r \ge r_{\text{out}}, \end{cases} \tag{121}$$

where the parameters are:

$$r_{\text{in}} = 0.8, \quad r_{\text{out}} = 1.0; \tag{122}$$

$$\rho_{\text{in}} = 10^{-2}, \quad \rho_{\text{out}} = 10^{-4}; \tag{123}$$

$$p_{\text{in}} = 1.0, \quad p_{\text{out}} = 3 \times 10^{-5}; \tag{124}$$

$$B^i = (0.1, 0, 0), \quad v^i = (0, 0, 0). \tag{125}$$

Here, we consider the ideal-gas equation of state with $\Gamma = 4/3$. The computational domain covers $[-6, 6]$ for both x and y directions with the resolution 128×128.

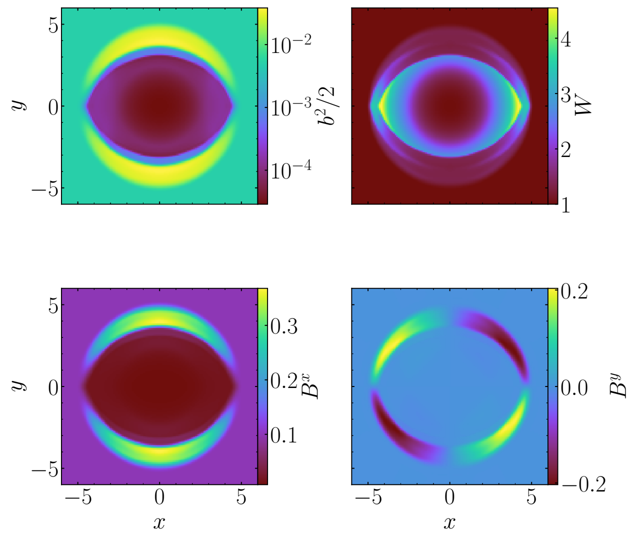


Figure 8. The 2D profile for the cylindrical blast wave of the magnetic field strength B^iB_i (upper left), Lorentz factor W(upper right), B^x (lower left), B^y (lower right) at $t = 4.0$.

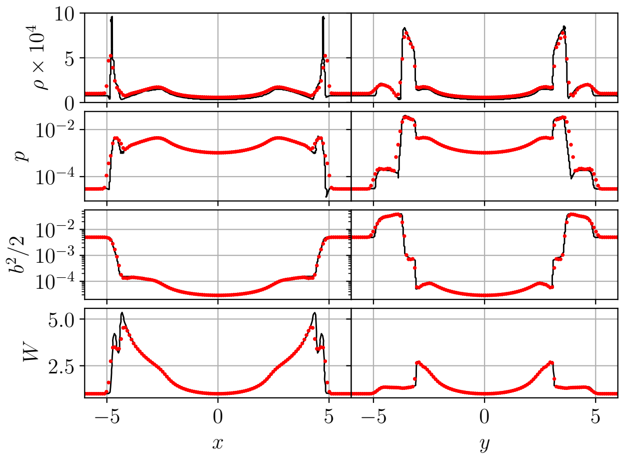


Figure 9. 1D slices along the x-axis (left column) and y-axis (right column) for the density ρ (top row), pressure p (second row), magnetic pressure $b^2/2$ (third row), and Lorentz factor W (fourth row) for the MHD cylindrical blast wave test at $t = 4.0$. The red dots show the numerical results obtained by `Gmunu`, which agree with the reference solutions (black solid lines) Mösta et al. (2014).

Fig. 8 shows the 2D profile of the magnetic field strength B^iB_i, B^x, B^y and the Lorentz factor W at $t = 4.0$. To compare our results with other groups (e.g. Mösta et al. 2014) in more detail, we also plot 1D slices along the x- and y-axes for the rest-mass density ρ, pressure p, magnetic pressure $b^2/2$, and the Lorentz factor W at $t = 4$, as shown in Fig. 9. In this test, the numerical results obtained by `Gmunu`, which agree with the reference solutions (Mösta et al. 2014).

3.2.4 *Loop advection*

The advection of a weakly magnetized loop is a well-known test to examine divergence-control technique in an MHD code. This test is performed on an uniform background with $\rho = 1$, $p = 1$, $v^x = 0.2$, and $v^y = 0.1$. The initial condition of the magnetic field B^i is given

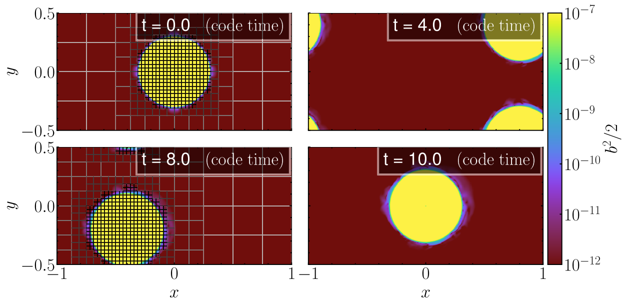


Figure 10. The evolution of the magnetic pressure $b^2/2$ for the loop advection test at various time slices. The AMR blocks of 8×8 cells are shown on the left-hand panels. The shape of the loop is preserved well at $t = 10$, where the magnetic field has translated with 1 cycle.

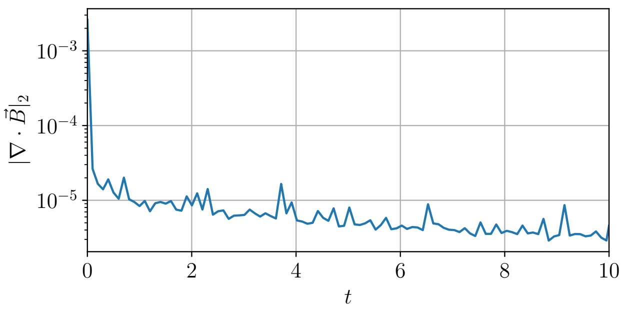


Figure 11. The L_2-norm of $\nabla \cdot \boldsymbol{B}$ versus time for the advected field loop test. The $|\nabla \cdot \boldsymbol{B}|_2$ is suppressed to lower than 10^{-5} immediately when the evolution started and is well controlled for the rest of the evolution.

as

$$B^i = \begin{cases} (-A_0 y/r, A_0 x/r, 0) & \text{if } r < R, \\ (0, 0, 0) & \text{if } r > R, \end{cases} \tag{126}$$

where $R = 3$ is the radius of the advecting magnetic loop, $r \equiv \sqrt{x^2 + y^2}$ and A_0 is chosen to be 10^{-3}. We consider an ideal-gas equation of state $p = (\Gamma - 1)\rho\epsilon$ with $\Gamma = 4/3$. The computational domain is set to be *periodic* at all boundaries and covers the region $-1 \leq x \leq 1$ and $-0.5 \leq y \leq 0.5$ with the base grid points $n_x \times n_y = 32 \times 16$ and allowing 5 AMR levels (i.e. an effective resolution of 512×256). Note that in this test, the refinement is determined based on the strength of the magnetic field. In particular, the grid is refined if the square of the magnetic field B^iB_i is larger than 10^{-10} while it is coarsen otherwise.

Fig. 10 gives an example of the evolution of the magnetic pressure $b^2/2$ for the loop advection test at different times. The shape of the loop is preserved well at $t = 10$, where the magnetic field has translated with 1 cycle.

Fig. 11 shows the evolution of the L_2-norm of $\nabla \cdot \boldsymbol{B}$, defined as

$$|\nabla \cdot \boldsymbol{B}|_2 \equiv \sqrt{\frac{1}{V}\int |\nabla \cdot \boldsymbol{B}|^2 \mathrm{d}V}, \tag{127}$$

which can be used to indicate the validity of the divergence-control. The L_2-norm of $\nabla \cdot \boldsymbol{B}$ is suppressed to lower than 10^{-5} immediately when the evolution started and is well controlled for the rest of the evolution. Overall, the elliptic divergence cleaning (see Section 2.2) works well to control monopole errors for this test case.

It is worth to point out that, although elliptic cleaning of the divergence of magnetic field is technically acausal, no artefacts from these tests are observed. The main reason is that, the magnetic monopole of the initial setup of the runs are below the machine precision, some are even identically equal to zero. Although the divergence of magnetic field is small, it is cleaned by using elliptic solver at each time-step. While this act is technically acausal, removing negligible and non-physical parts of the magnetic field at each time-step will not significantly affect the evolution of the systems. Artefacts may arises when the divergence of magnetic field is non-negligible before cleaning, however, this situation itself is non-physical. As mentioned, more detailed and systematic studies of different divergence handling approaches is planned in the future.

3.3 General relativistic (magneto-)hydrodynamics in dynamical space–time

3.3.1 Refinement criteria

In addition to the error estimators mentioned above, user-defined additional conditions are available in MPI-AMRVAC. Note that all the error estimators discussed are mostly local (block-based calculations), applying them only may not sufficient to have a optimized mesh refinement. For example, numerical studies suggest that the ratio of the radius of the compact objects R and the grid size Δx should be larger than 50 for neutron stars (Shibata 2015). In our experience, it is hard to simultaneously resolve the interior of the star and the star surface properly by applying a local estimator only. This is because the density gradients at the star surface could be extremely large. As a result, the interior of the star seems sufficiently smooth (refining is not necessary) compare with the star surface. Moreover, low density matters often eject from the star surface into the vacuum, which will trigger local error estimator to refine the mesh in these regions. The computational cost are wasted if those ejecta is not the main focus of the studies. Thus, it is usually beneficial to include addition conditions on top of the local error estimators if prior knowledge of the system are available.

The lapse function α can be used as an indicator for the grid refinement (Shibata & Shapiro 2002; Shibata & Sekiguchi 2004). We defined a relativistic gravitational potential $\Phi := 1 - \alpha$. Since Φ is approximately proportional to M/R, Φ^{-1} can be used as a measure of the characteristic length-scale. The grid resolution can then be optimized by assigning the valid range of the potential for all available levels, e.g. Φ_l. This approach is adopted in all our general relativistic simulations to be discussed below. The grid refinement used in this work is the following: For any Φ larger than the maximum potential $\Phi_{\max}$ (which is set as 0.2 in this work), the block is set to be finest. While for the second finest level, the same check is performed with a new maximum potential which is half of the previous one, so on and so forth. The grid is updated every 500 time-steps.

3.3.2 Stability of a rapidly rotating neutron star

Here, we study the evolution of a stable rapidly rotating neutron star with a dynamical background metric. In this test, we consider a uniformly rotating model which is constructed with the polytropic equation of state with $\Gamma = 2$ and $K = 100$ with central rest-mass density $\rho_c = 1.28 \times 10^{-3}$ and the angular velocity $\Omega = 2.633 \times 10^{-2}$ (in $c = G = M_\odot = 1$ unit), which is also know as 'BU8' in the literature (Dimmelmeier et al. 2006; Cordero-Carrión et al. 2009). The initial neutron star model is generated with the open-source code XNS (Bucciantini & Del Zanna 2011; Pili, Bucciantini & Del Zanna 2014, 2015, 2017). The computational domain covers $0 \le r \le 30$, $0 \le \theta \le \pi/2$ with the resolution $n_r \times n_\theta = 640 \times 64$. This test problem is simulated with the ideal-gas equation of state $P = (\Gamma - 1)\rho\epsilon$ with

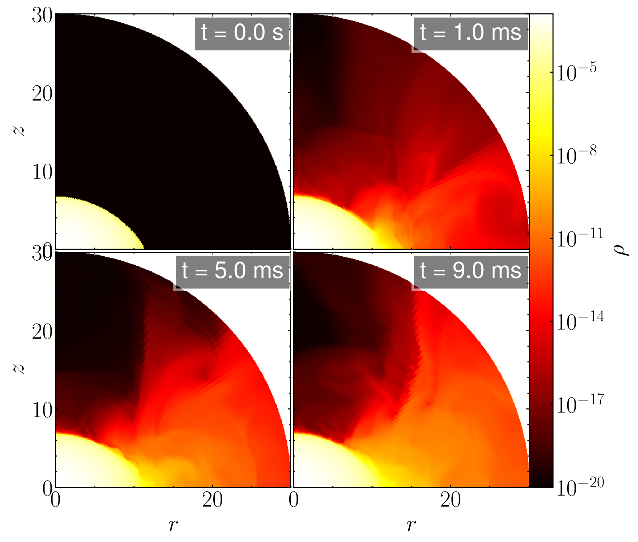


Figure 12. Example of the evolution of the density ρ of the rapidly rotating neutron star BU8 at various time slices. As shown in the density map, since the 'atmosphere' density ρ_{atmo} is set to be 10^{-20}, the low-density fluid (e.g. $\rho \sim 10^{-9}$ to 10^{-13}, which are the typical values of the 'atmosphere' in the literature) is free to be evolved without crashing the code. This can be achieved with the positivity preserving limiter (see Section 2.4) and could significantly avoid violations of the conservation properties at the neutron star surface.

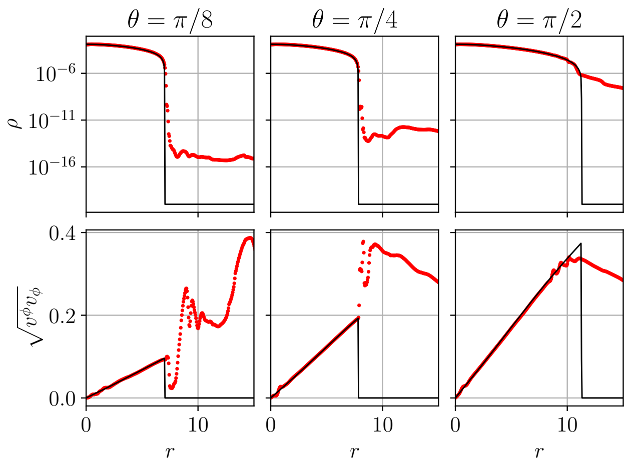


Figure 13. 1D slices of the rapidly rotating neutron star BU8 along the $\theta = \pi/8$ (left), $\theta = \pi/4$ (middle), and $\theta = \pi/2$ (right) for the density ρ (upper) and the rotational velocity $\sqrt{v_\phi v^\phi}$ (lower). The black solid lines show the initial profiles while the red dots show the profiles $t = 9$ ms.

$\Gamma = 2$. Long-time evolution of this model is demanding since the rotational rate is close to the mass shedding limit. While maintaining this model stably is formidable, we challenge the robustness of our code with the use of the positivity preserving limiter by setting an extremely low 'atmosphere' density $\rho_{\text{atmo}} = 10^{-20}$ (which is below *machine precision*) and simulate the system with a 5th order reconstruction method MP5. As in Cheong et al. (2020), in order to increase the size of the time-steps in our simulations, we treat $0 < r < 0.4$ as a spherically symmetric core (i.e. only radial motions are allowed).

With the positivity preserving limiter and the robust recovery of primitive variables scheme, Gmunu evolve such demanding systems stably even with an extremely low density of 'atmosphere' up to at

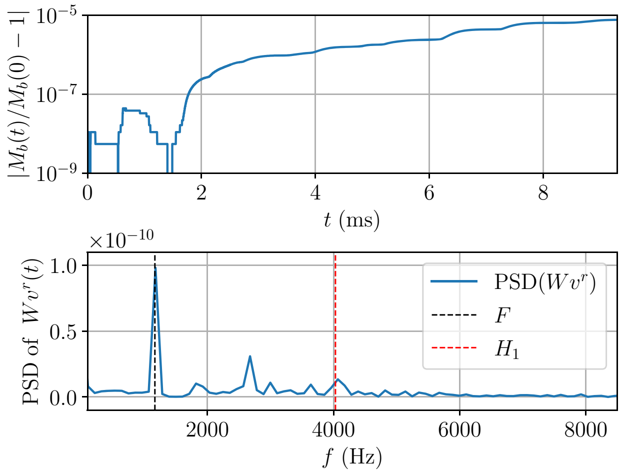


Figure 14. Upper panel: The relative variation of the rest mass M_b of the rapidly rotating neutron star BU8 in time. Lower panel: The power spectral density of the radial velocity $Wv^r(t)$ at $r = 5$, $\theta = \pi/4$ (inside the neutron star). The vertical lines represent the known and well-tested eigenmode frequencies (Dimmelmeier et al. 2006). Even for such rapidly rotating neutron star BU8 with extreme simulations settings (i.e. $\rho_{\rm atmo} = 10^{-20}$ with MP5 reconstruction), Gmunu is able to maintain the profile up to 9 ms and the relative variation of the rest mass of the order 10^{-5}.

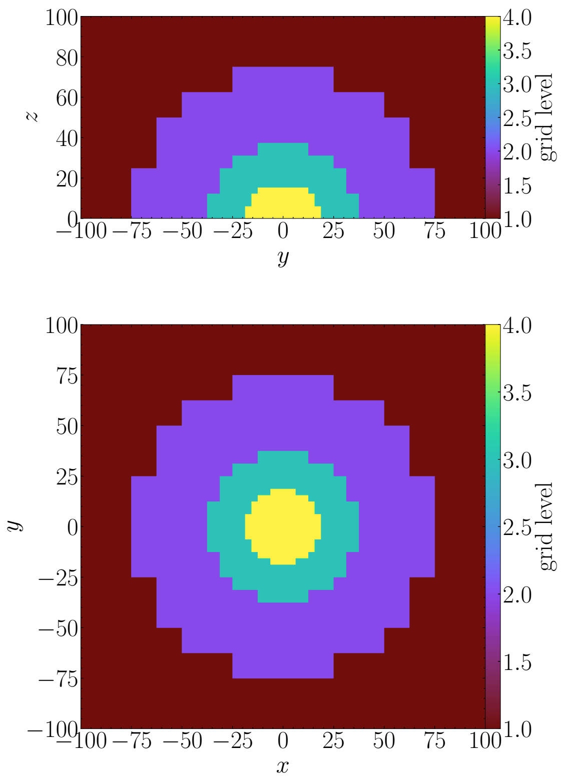


Figure 15. The projection of grid levels along x-axis (upper panel) and z-axis (lower panel) in Cartesian coordinates. The computational domain covers $[-100, 100]$ for both x, y while $z \in [0, 100]$, with the resolution $N_x \times N_y \times N_z = 64 \times 64 \times 32$ and allowing 5 AMR level (an effective resolution of $1024 \times 1024 \times 512$).

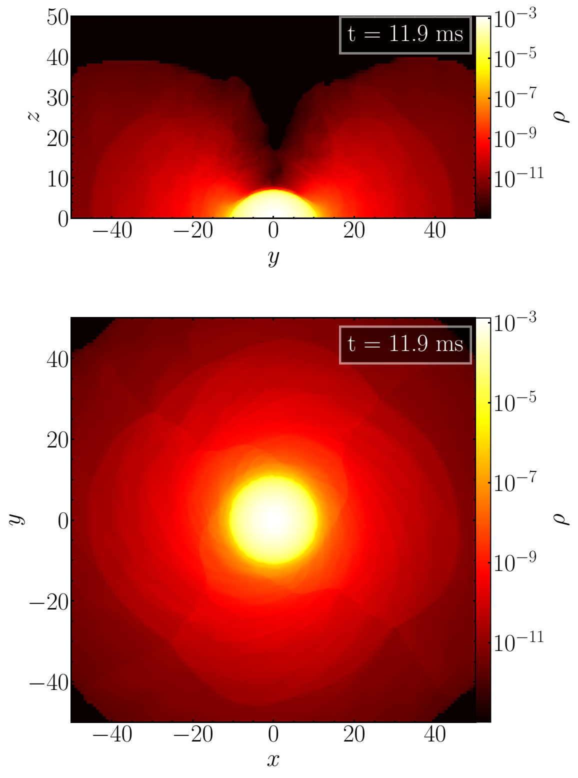


Figure 16. The projection of density profile along x-axis (upper panel) and z-axis (lower panel) of a rapidly rotating neutron star in Cartesian coordinates with the annotated mesh lines at $t = 11.9$ ms.

least $t = 9$ ms without crashing the code. Fig. 12 gives an example of the evolution of this rapidly rotating neutron star model BU8 at different time. Fig. 13 shows 1D slices of the rapidly rotating neutron star BU8 along the $\theta = \pi/8$, $\theta = \pi/4$, and $\theta = \pi/2$ for the density ρ and the rotational velocity $\sqrt{v_\phi v^\phi}$, respectively. The density and the velocity profiles are maintained well except that some low density 'atmosphere' $\rho \sim 10^{-9}$ to 10^{-17} is surrounding the neutron star.

To illustrate the conservation properties, we monitor the total rest mass M_b of the whole system, where the rest mass M_b is given by

$$M_b = \int_{\Sigma_t} \psi^6 \rho W \sqrt{\hat{\gamma}} \mathrm{d}^3 x. \tag{128}$$

The upper panel of Fig. 14 shows the relative variation of the rest mass M_b in time. Even for such rapidly rotating neutron star BU8 with extreme configurations, Gmunu is able to maintain the profile up to 9 ms and the relative variation of the rest mass of the order 10^{-5}. As an another indicator for the validity of the code, the lower panel of Fig. 14 shows the power spectral density of the radial velocity $Wv^r(t)$ at $r = 5$, $\theta = \pi/4$ (inside the neutron star), which agrees with the well-tested eigenmode frequencies (Dimmelmeier et al. 2006).

We simulated the same model in Cartesian coordinates (x, y, z). The computational domain covers $[-100, 100]$ for both x, y while $z \in [0,100]$, with the resolution $N_x \times N_y \times N_z = 64 \times 64 \times 32$ and allowing 5 AMR level (an effective resolution of $1024 \times 1024 \times 512$). Fig. 15 shows the grid level at different locations in the computational domain while Fig. 16 shows the projection of density profile of a rapidly rotating neutron star.

Fig. 17 shows the evolution of the Rapidly rotating neutron star BU8 in Cartesian coordinate. Gmunu is able to maintain the profile

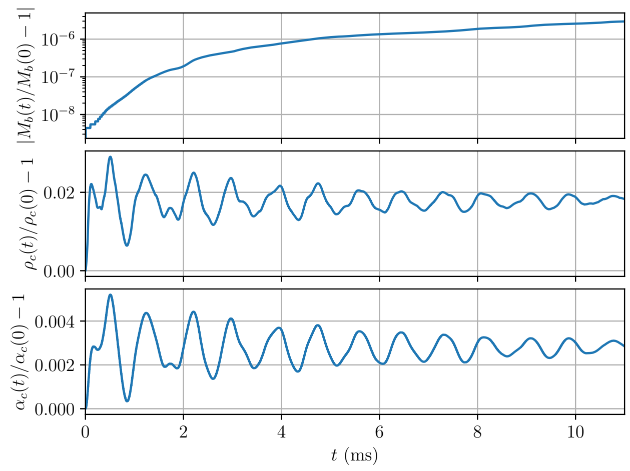


Figure 17. Upper panel: The relative variation of the rest mass M_b in time. The conservation of the rest mass M_b is preserved remarkably well from $t = 0$ to $t = 12$ ms where the relative variation is of the order 10^{-5}. Middle panel: The relative variation of the density ρ_c in time. Lower panel: The relative variation of the lapse function α_c in time.

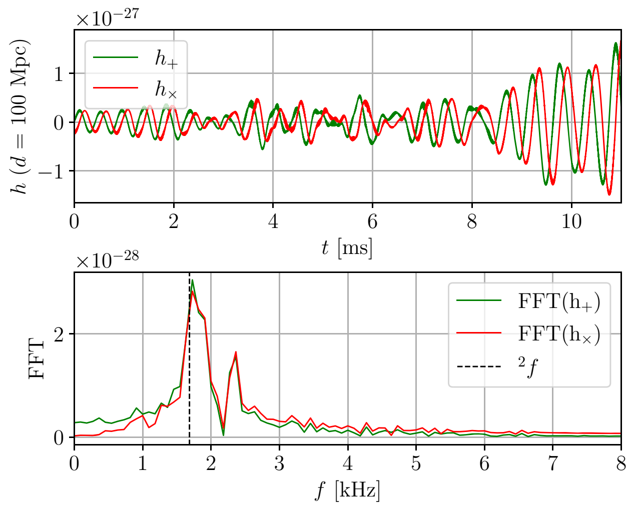


Figure 18. Upper panel: Gravitational waves extracted from the rapidly rotating neutron star BU8 in time at distance $d = 100$ Mpc at the equator. Lower panel: The Fast Fourier Transform the gravitational waves. The vertical lines represent the known and well-tested eigenmode (^2f mode) frequencies (Dimmelmeier et al. 2006).

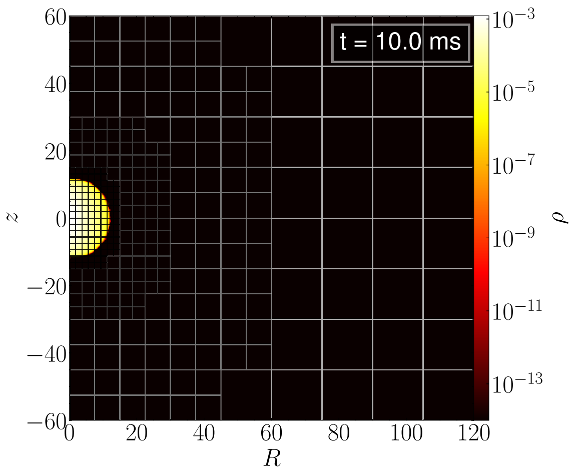


Figure 19. The density profile of a differentially rotating strongly magnetized equilibrium neutron star in cylindrical coordinates with the annotated mesh lines at $t = 10$ ms. The computational domain covers $0 \le R \le 120$ and $-120 \le z \le 120$, with the resolution $n_R \times n_z = 32 \times 64$ and allowing 5 AMR levels. At the outer region ($R \sim 100$), the size of a block (containing 8×8 cells) is almost the size of the neutron star.

up to 100 ms and the relative variation of the rest mass of the order of 10^{-5}.

Although conformally flat approximation is a gravitational-waveless approximation to general relativity, gravitational waves can still be extracted by using quadrupole formula. Fig. 18 shows the gravitational waves extracted at distance $d = 100$ Mpc at the equator in time domain and also in frequency domain, the dominating non-radial ^2f mode which agrees with the well-tested eigenmode frequencies (Dimmelmeier et al. 2006).

3.3.3 *Differentially rotating strongly magnetized neutron star*

Here, we study the evolution of a differentially rotating strongly magnetized equilibrium neutron star. As there are no similar studies in the literature except (Bucciantini & Del Zanna 2011), we use the same equilibrium model as in Bucciantini & Del Zanna (2011) here. In this test, we construct an equilibrium model with a polytropic equation of state with $\Gamma = 2$ and $K = 100$ with central rest-mass density $\rho_c = 1.28 \times 10^{-3}$. The neutron star is differentially rotating with $\Omega_c = 2.575 \times 10^{-2}$, $A^2 = 70$ and is magnetized with magnetic polytropic index $m = 1$ and magnetic coefficient $K_m = 3$. Here, we note that this is a strong toroidal magnetic field, $\sim 5 \times 10^{17}$ G inside the neutron star, which is roughly 10 per cent of the total internal energy. This test problem is simulated with the polytrope equation of state with $\Gamma = 2$ and $K = 100$.

We simulate this initial model in 2D cylindrical coordinates (R, z, φ), where the computational domain covers $0 \le R \le 120$ and $-120 \le z \le 120$, with the resolution $n_R \times n_z = 32 \times 64$ and allowing 5 AMR levels (i.e. an effective resolution of 512×1024). As an example, Fig. 19 shows the density profile with the annotated mesh lines at $t = 10$ ms.

Fig. 20 shows the evolution of this differentially rotating strongly magnetized equilibrium neutron star in cylindrical coordinate. The rest mass M_b is unchanged during the whole simulation ($t = 0$ to $t = 10$ ms). Fig. 21 compares the initial ($t = 0$) density profile, rotational velocity and the magnetic field (black solid lines) with the same quantities (red dots) at $t = 10$ ms. The profiles are maintained well except some slight distortions.

3.3.4 *Stability of a non-rotating neutron star*

We present a full 3D simulation of a spherically symmetric neutron star here. In this test, we consider a non-rotating model which is constructed with the polytropic equation of state with $\Gamma = 2$ and $K = 100$ with central rest-mass density $\rho_c = 1.28 \times 10^{-3}$(in $c = G = M_\odot = 1$ unit), which is also know as ‘BU0’ in the literature (Dimmelmeier et al. 2006; Cordero-Carrión et al. 2009). Actually, such a spherically symmetric model can be simulated in a 1D or 2D spherical coordinate. Nevertheless, as a demonstration, we simulate this system in 3D Cartesian coordinates without imposing any symmetries, i.e. this problem is simulated in the full 3D configuration. The computational domain covers [−100, 100] for both x, y, and z, with the resolution $n_x \times n_y \times n_z = 64 \times 64 \times$

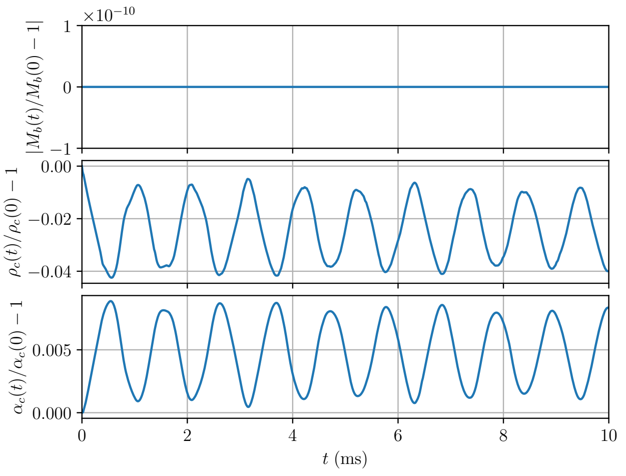


Figure 20. Upper panel: The relative variation of the rest mass M_b in time. The conservation of the rest mass M_b is preserved remarkably well from $t = 0$ to $t = 10$ ms where the relative variation is zero (so cannot be plotted in log scale). Middle panel: The relative variation of the density ρ_c in time. Lower panel: The relative variation of the lapse function α_c in time.

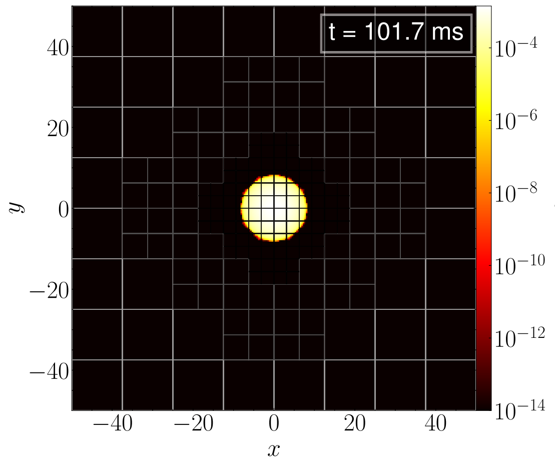


Figure 22. The projection of density profile along z-axis of a spherical neutron star in Cartesian coordinates with the annotated mesh lines at $t = 101.7$ ms. The computational domain covers $[-100, 100]$ for both x, y, and z, with the resolution $n_x \times n_y \times n_z = 64 \times 64 \times 64$ and allowing 4 AMR level (an effective resolution of 512^3).

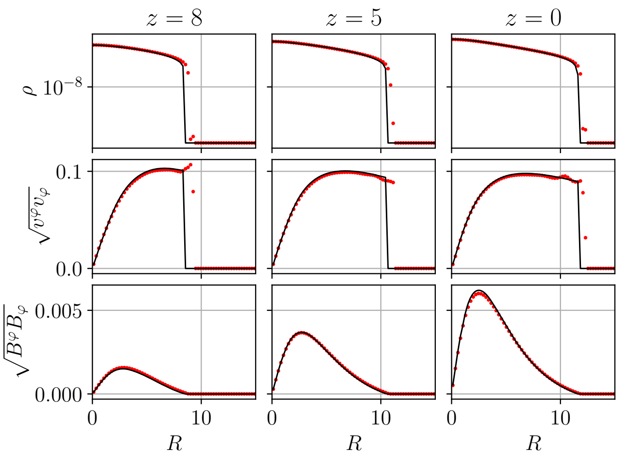


Figure 21. 1D slices of a differentially rotating strongly magnetized equilibrium neutron star in spherical coordinates along the $z = 8$ (left column), $z = 5$ (middle column), and $z = 0$ (right column) for the density ρ (upper row), the rotational velocity $\sqrt{v_\varphi v^\varphi}$ (middle row), and the magnetic field $\sqrt{B_\varphi B^\varphi}$. The black solid lines show the initial profiles while the red dots show the profiles $t = 10$ ms.

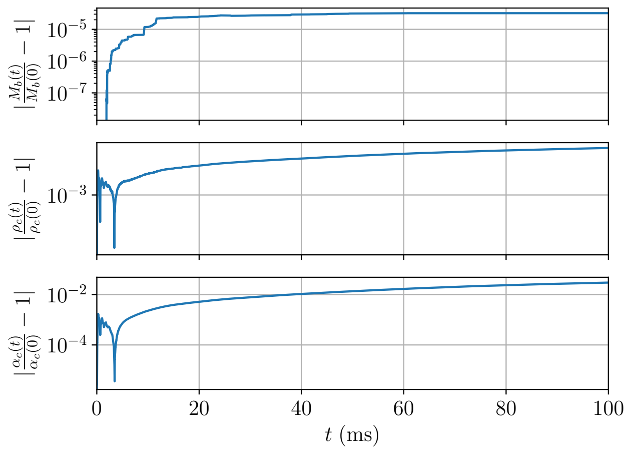


Figure 23. Upper panel: The relative variation of the rest mass M_b in time. The conservation of the rest mass M_b is preserved remarkably well from $t = 0$ to $t = 100$ ms where the relative variation is of the order of 10^{-4}. Middle panel: The relative variation of the density ρ_c in time. Lower panel: The relative variation of the lapse function α_c in time.

64 and allowing 4 AMR level (an effective resolution of 512^3). The refinement setting is identical to Section 3.3.3. As an example, Fig. 22 shows the density profile with the annotated mesh lines at $t = 101.7$ ms.

Fig. 23 shows the evolution of the spherically symmetric neutron star BU0 in Cartesian coordinate. Gmunu is able to maintain the profile up to 100 ms and the relative variation of the rest mass of the order of 10^{-4}. Fig. 24 compares the initial density profile (black solid lines) with the same quantities (red dots) at $t = 101.7$ ms. The density profile is maintained well.

3.3.5 Migration of an unstable neutron star

To see how Gmunu preform in the fully non-linear regime with significant changes and coupling in the metric and fluid variables, here we present a simulation of the migration of an unstable neutron star, which is one of the standard tests for hydrodynamical evolution coupled with dynamical space–time in the fully non-linear regime (Font et al. 2002; Cordero-Carrión et al. 2009; Bernuzzi & Hilditch 2010; Bucciantini & Del Zanna 2011). In this test, we consider an unstable neutron star, which lies on the unstable branch of the mass–radius curve. The neutron star is constructed with the polytropic equation of state with $\Gamma = 2$ and $K = 100$ with central rest-mass density $\rho_\mathrm{c} = 8.00 \times 10^{-3}$ (in $c = G = M_\odot = 1$ unit), which is also known as 'SU' in Cordero-Carrión et al. (2009). We simulate this initial model in 2D cylindrical coordinates (R, z), where the computational domain covers $0 \le R \le 60$ and $-60 \le z \le 60$, with the resolution $n_R \times n_z = 32 \times 64$ and allowing 4 AMR level (i.e. an effective resolution of 256×512). The refinement setting is identical to Section 3.3.3. We adopt the ideal-gas (gamma-law) equation of state $P = (\Gamma - 1)\rho\epsilon$ with $\Gamma = 2$ for the fluid so that we can also capture the shock heating effect.

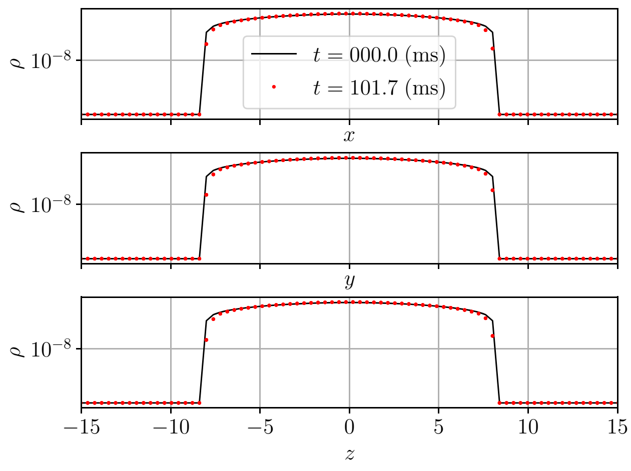


Figure 24. 1D slices of the non-rotating equilibrium neutron star BU0 along the x-axis (upper panel), y-axis (middle panel), and z-axis (lower panel) for the density ρ. The black solid lines show the initial profiles while the red dots show the profiles $t = 101.7$ ms.

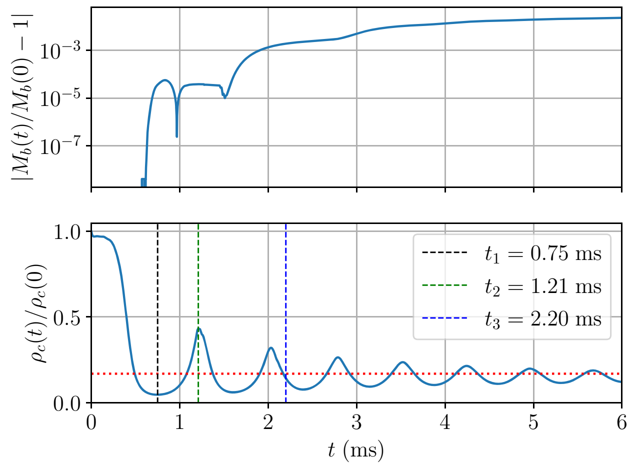


Figure 25. Evolution of an unstable spherically symmetric neutron star. Upper panel: The relative variation of the rest mass M_b in time. Lower panel: The central density $\rho_c/\rho_c(t=0)$ in time. The dotted line represents the central density ρ_c of the neutron star on the stable branch.

As the star evolves and migrates to the corresponding stable configuration $\rho_c = 1.346 \times 10^{-3}$ with the same mass, the radius of the star expands to a large value. Fig. 25 shows the evolution of the baryon mass M_b and the central density ρ_c as a function of time. The oscillations of the central density ρ_c are damped since shock waves are formed at every pulsation and some kinetic energy is dissipated into thermal energy. A small amount of mass is ejected outwards from the surface of the star to the surrounding artificial low-density ($\rho_{\rm atmo} = 10^{-14}$) 'atmosphere' whenever these shock waves hit the surface of the star, and thus the total baryon mass M_b decays once the shock waves hit the outer numerical boundaries. With weaker oscillation, the decay rate of the baryon mass is smaller. This dissipation effect can also be seen in the density profile, as shown in Fig. 26. As a result, the baryon mass and the central density of the final equilibrium stable configuration is slightly lower than the expected value.

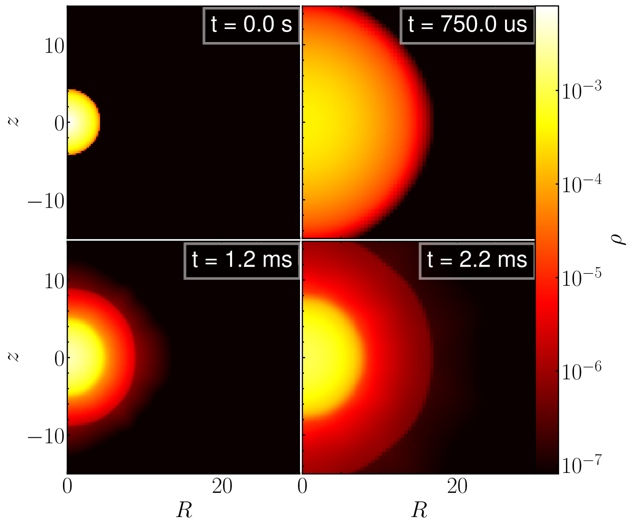


Figure 26. The evolution of the density ρ an unstable neutron star SU at various time slices. Initially (upper left), the star has the central density $\rho_c = 8.00 \times 10^{-3}$ with a radius $r = 4.267$. As the star evolves, the central density reduces and the radius of the star expands to a larger value. At $t = 0.75$ ms (upper right), the central density of the system reach the lowest point and start to increase. At $t = 1.21$ ms (lower left), the central density of the system hits the local maxima. A small amount of mass is ejected outside the surface of the star due to the shock heating effect. After some time, at $t = 2.20$ ms (lower right), the ejected mass surrounding the star covers most of the computational domain.

4 PERFORMANCE AND SCALING

In this section, we present two tests to assess the strong scaling of Gmunu. The scaling tests to be presented below were obtained on the Central Research Computing Cluster in The Chinese University of Hong Kong. In particular, in all tests, we used computing nodes on the Central Cluster with dual Intel Xeon Gold 6130 processors, for a total of 32 cores per node, and one MPI process per core was used.

4.1 Scaling of special-relativistic (magneto-)hydrodynamics

In this subsection, we focus on the strong scaling of the hydrodynamics solver (hyperbolic sector) of Gmunu. The test here is the 2D Riemann problem, as discussed in Section 2.8.1, with a slightly different setting. In particular, in this test, PPM reconstruction is used, and the simulation box consist of a set of uniform grid 1024^2 decomposed into 128^2 blocks with each block of 8^2 cells. Fig. 27 shows the cells updated per second at different number of cores.

4.2 Performance of the metric solver

In this section, we demonstrate the convergence properties and performance of our multigrid metric solver. We solve the metric of model BU8 (see 3.3.2), which represents a rapidly rotating neutron star and far from spherically symmetric, in a *full* 3D setting. For instance, the computational domain covers [−200, 200] for both x, y and z, with the resolution $N_x \times N_y \times N_z = 256 \times 256 \times 256$, is decomposed into $16 \times 16 \times 16$ blocks with each block of $16 \times 16 \times 16$ cells. In addition to this uniform grid setting, we also consider the same case with allowing 4 AMR level (an effective resolution of $2048 \times 2048 \times 2048$). To make it a fair and general comparision,

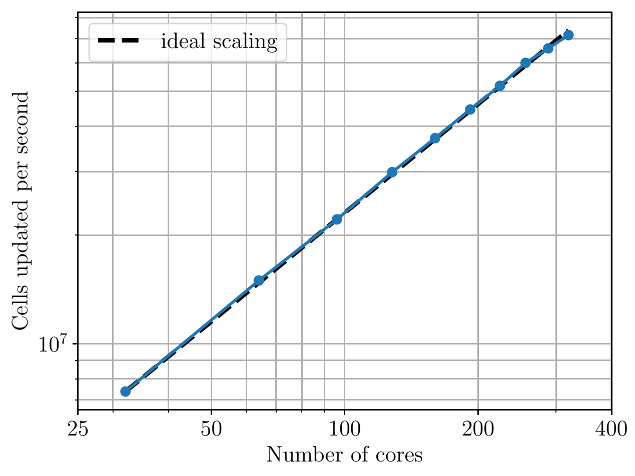


Figure 27. Strong scaling of a relativistic hydrodynamics application with a set of uniform grid 1024^2 decomposed into 128^2 blocks with each block of 8^2 cells. The figure shows the cells updated per second for increasing number of cores. The blue line shows the results obtained by `Gmunu` while the black dashed line is the ideal scaling.

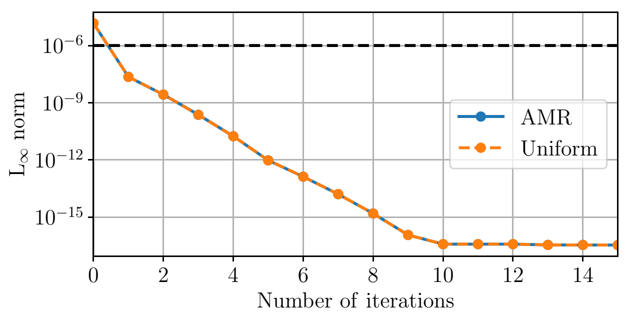


Figure 28. L_∞ norm of residual of equation (74) of an highly non-spherically symmetric model BU8 as a function of the number of full multigrid (FMG) iterations. The convergence properties with or without AMR activated are almost identical in this test case. Even if the multigrid solver starts from the flat space initial guess, *one* iteration is sufficient to converge to the prescribed tolerance (horizational black dashed line), and the residual is reduced up to machine precision (i.e. $L_\infty \lesssim 10^{-16}$) after about 10 iterations.

instead of using the pre-solved initial data as the initial guess, we focus on solving the lapse function α with the flat space initial guess $\alpha = 1$. In the following test, two upward and downward red-black Gauss–Seidel smoothing steps are used.

4.2.1 *Convergence properties*

Fig. 28 shows the L_∞ norm of the residual of equation (74) as a function of the number of full multigrid (FMG) iterations. The convergence properties with or without AMR activated are almost identical in this test case. Even if the multigrid solver starts from the flat space initial guess, *one* iteration is sufficient to converge to the prescribed tolerance (horizational black dashed line), and the residual is reduced up to machine precision (i.e. $L_\infty \lesssim 10^{-16}$) after about 10 iterations. The L_∞ norm of residual of equation (74) in both cases are identical, which implies the maximum residual in both cases are the same. Indeed, in this test, the maximum residual is located at the outer boundary of the computational domain, where the resolution with AMR is set to be the lowest in purpose (see the discussion of Section 3.3.1), which is identical to the uniform case.

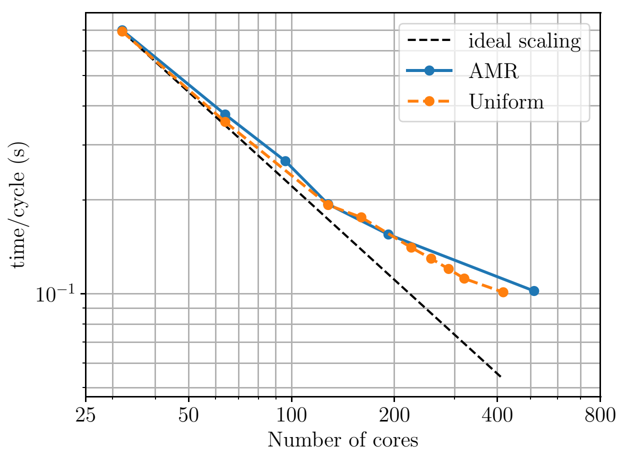


Figure 29. Computational time per full multigrid (FMG) cycles when solving equation (74) of an highly non-spherically symmetric model BU8 as a function of the number of cores. The computational time per cycle reduces with increasing number of cores. The scaling close to the ideal scaling for the number of cores $\lesssim 100$, however, it is not ideal with larger amount of cores. This is because the problem size is not larger enough to have good scaling.

In practice, at the beginning of the simulation, we use the initial data provided by `XNS` as initial guess. During the evolution, we use the previous solution as initial guess for the next iteration. This makes the solver converge much faster as the solutions on previous time step are usually good approximation to the solution.

4.2.2 *Strong scaling*

Here, we assess the performance and scaling of our metric solver with the same setup mentioned above. We measure the time per full multigrid (FMG) cycle by averaging over 500 cycles. Fig. 29 shows the computational time per full multigrid (FMG) cycles when solving equation (74) of an highly non-spherically symmetric model BU8 as a function of the number of cores. The computational time per cycle reduces with increasing number of cores. The scaling close to the ideal scaling for the number of cores $\lesssim 100$, however, it is not ideal with larger amount of cores. This is because the problem size is not larger enough to have good scaling. For instance, in the uniform case ($N_x \times N_y \times N_z = 256 \times 256 \times 256$) with 416 cores, only about 32^3 unknowns are involved.

4.3 Scaling of general-relativistic hydrodynamics

Finally, we assess the performance of `Gmunu` in general-relativistic hydrodynamics simulations. In this test, we again evolve the rapidly rotating neutron star BU8 (see Section 3.3.2) but this time in a *full* 3D setting. For instance, the computational domain covers [−200, 200] for both *x*, *y*, and *z*, with the resolution $N_x \times N_y \times N_z = 64 \times 64 \times 64$ and allowing 5 AMR level (an effective resolution of $1024 \times 1024 \times 1024$). The simulation box is decomposed into $8 \times 8 \times 8$ blocks with each block of $8 \times 8 \times 8$ cells. Also, TVDLF Riemann solver and third-order accurate PPM limiter is used. This system is evolved to $T_{\rm final} = 10$ ms, and output data at every 1 ms. Note that, this test includes not only the hydrodynamics part (hyperbolic sector), but also the metric equations (elliptic sector). Unlike in the case of divergence cleaning of magnetic field, where the elliptic equation needed to be solve is the trivial and well behave Poisson equation, the metric equations in extended CFC scheme are highly non-linear and

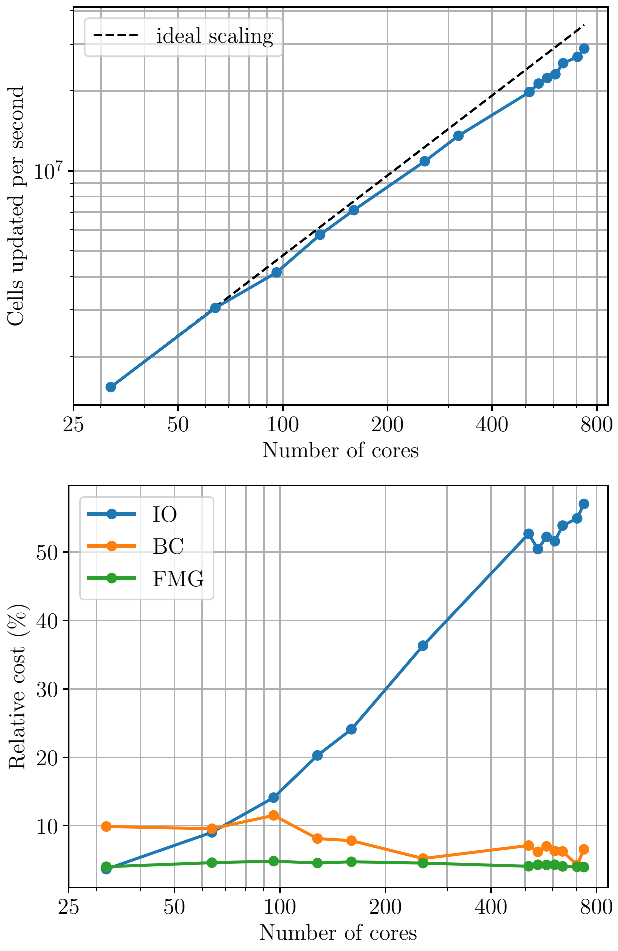


Figure 30. Strong scaling of the evolution of full 3D rapidly rotating neutron star BU8 in Cartesian coordinates. The computational domain covers [−200, 200] for both x, y, and z, with the resolution $N_x \times N_y \times N_z = 64 \times 64 \times 64$ and allowing 5 AMR level (an effective resolution of 1024 × 1024 × 1024). The simulation box is decomposed into 8 × 8 × 8 blocks with each block of 8 × 8 × 8 cells. The upper panel shows the cells updated per second for increasing number of cores. The blue line shows the results obtained by Gmunu while the black dashed line is the ideal scaling. Note that even with the elliptic metric solver included, the scaling is closed to ideal scaling. The lower panel shows the relative cost measure by time of different parts of the code. The cost of the metric solver (with full multigrid (FMG), green line) is slightly below 5 per cent in all cases we have tested, which is even below the cost required for updating boundary conditions (BC, orange line). The relative cost of output data (IO, blue line) gradually increase with increasing number of cores, the computational time of which is limited by the speed of writing/reading data to/from the hard disc drive (disc I/O speed).

include vectorial elliptic equations, the computational cost required by the metric solve may be different from time to time during the dynamical simulations.

Fig. 30 shows the strong scaling and the relative computational cost (measure by time) of this test problem. As shown on the left-hand panel, the scaling is closed to ideal scaling, even with the elliptic metric solver included. Besides, as shown on the right-hand panel, the relative cost of metric solver is slightly below 5 per cent in all cases we have tested, which is relatively low in the sense that even the cost of updating the boundary conditions (including ghost cells) requires more than the metric solver. The relative cost of output data (IO) gradually increase with increasing number of cores, the computational time of which is limited by the speed of writing/reading data to/from the hard disk drive (disc I/O speed).

5 CONCLUSIONS

We present the new methodology and implementation of Gmunu, a parallelized multiscale multidimensional curvilinear general-relativistic magnetohydrodynamics code with a cell-centred non-linear multigrid solver which is fully coupled with an adaptive mesh refinement modules. The code has been designed to perform generic general relativistic (magneto-)hydrodynamical simulations in dynamical space–time. With the flexibility of choosing coordinates and the efficient block-based adaptive mesh refinement module, depending on the nature of the problems and the study interests, users can balance the computational cost and the accuracy of the results easily without changing to other codes. For the divergence-less handling for the magnetic field, in this work, we present, to our knowledge, the first example of using elliptic divergence cleaning dynamically during the relativistic magnetohydrodynamics simulations. Currently, Gmunu is able to solve the elliptic-type metric equations in the extended conformally flat condition (xCFC) approximation to general relativity.

We have tested Gmunu with several benchmarking tests, from special-relativistic to general-relativistic (magneto-)hydrodynamics in 1D, 2D, and 3D Cartesian, cylindrical, and spherical coordinates. These tests include (i) SR(M)HD shock tubes, SRHD Riemann test, axisymmetric jet in SRHD, cylindrical blast wave and magnetic field loop advection in SRMHD, the evolution of rapidly/ differentially rotating, strongly magnetized neutron stars in GR(M)HD. In the GRMHD tests, we demonstrate that the multigrid algorithm in Gmunu is able to solve CFC metric equations in multiple dimensions and in different coordinates with or without coupling with the AMR module. In addition, the robust positivity preserving limiter and conserved-to-primitive variables conversions enable us to set the density of the ‘atmosphere’ $\rho_{\rm atmo}$ to the order of $\mathcal{O}(10^{-20})$ (below machine precision) even in the evolution of a rapidly rotating or strongly magnetized neutron star with good rest mass conservation and accurate results.

In the future, we will present the implementations and comparisons of various divergence-free treatments, i.e. elliptic cleaning, generalized Lagrange multiplier (GLM), constrained transport (CT) and the vector potential schemes. Furthermore, we will implement radiation hydrodynamics for MHD also for neutrino physics. We shall also extend Gmunu to a fully constrained scheme in exact general relativity such as the formulation of Bonazzola et al. (2004).

ACKNOWLEDGEMENTS

PCKC thanks David Yat-Tung Pong for setting up and providing technical support for The Chinese University of Hong Kong – Gravitational Wave group workstations. We acknowledge the support of The Chinese University of Hong Kong Central High Performance Computing Cluster, on which the scaling tests in this work have been performed. This work was partially supported by grants from the Research Grants Council of Hong Kong (Project No. CUHK24304317 and CUHK 14306419), the Croucher Innovation Award from the Croucher Fundation Hong Kong and by the Direct Grant for Research from the Research Committee of The Chinese University of Hong Kong.

DATA AVAILABILITY

The data underlying this article are available in the article.

REFERENCES

Abramowicz M. A., Fragile P. C., 2013, Liv. Rev. Relativ., 16, 1
Alcubierre M., 2008, Introduction to 3 + 1 Numerical Relativity. Oxford Univ. Press, Oxford
Anile A. M., 1990, Relativistic Fluids and Magneto-fluids: With Applications in Astrophysics and Plasma Physics (Cambridge Monographs on Mathematical Physics). Cambridge Univ. Press, Cambridge
Baiotti L., Rezzolla L., 2017, Rep. Prog. Phys., 80, 096901
Balsara D., 2001, ApJS, 132, 83
Berger M. J., Colella P., 1989, J. Comput. Phys., 82, 64
Berger M. J., Oliger J., 1984, J. Comput. Phys., 53, 484
Bernuzzi S., Hilditch D., 2010, Phys. Rev. D, 81, 084003
Bonazzola S., Gourgoulhon E., Grand clément P., Novak J., 2004, Phys. Rev. D, 70, 104007
Briggs W. L., Henson V. E., McCormick S. F., 2000, A Multigrid Tutorial, 2nd Edition. SIAM
Bryan G. L. et al., 2014, ApJS, 211, 19
Bucciantini N., Del Zanna L., 2011, A&A, 528, A101
Burrows A., 2013, Rev. Mod. Phys., 85, 245
Cheong P. C.-K., Lin L.-M., Li T. G. F., 2020, Class. Quantum Gravity, 37, 145015
Cipolletta F., Kalinani J. V., Giacomazzo B., Ciolfi R., 2020, Class. Quantum Gravity, 37, 135010
Cordero-Carrión I., Cerdá-Durán P., Dimmelmeier H., Jaramillo J. L., Novak J., Gourgoulhon E., 2009, Phys. Rev. D, 79, 024017
Courant R., Friedrichs K., Lewy H., 1928, Math. Ann., 100, 32
Del Zanna L., Bucciantini N., 2002, A&A, 390, 1177
Del Zanna L., Zanotti O., Bucciantini N., Londrillo P., 2007, A&A, 473, 11
Dimmelmeier H., Font J. A., Müller E., 2002, A&A, 388, 917
Dimmelmeier H., Stergioulas N., Font J. A., 2006, MNRAS, 368, 1609
Dowell M., Jarratt P., 1971, BIT Num. Math., 11, 168
Duez M. D., Zlochower Y., 2019, Rep. Prog. Phys., 82, 016902
Faber J. A., Rasio F. A., 2012, Liv. Rev. Relat., 15, 8
Font J. A. et al., 2002, Phys. Rev. D, 65, 084024
Fryxell B. et al., 2010, Astrophysics Source Code Library, record ascl:1010.082
Galeazzi F., Kastaun W., Rezzolla L., Font J. A., 2013, Phys. Rev. D, 88, 064009
Gammie C. F., McKinney J. C., Tóth G., 2003, ApJ, 589, 444
Gourgoulhon E., 2007, preprint (arXiv:gr–qc/0703035)
Harten A., Lax P., Leer B., 1983, SIAM Rev., 25, 35
He P., Tang H., 2012, Commun. Comput. Phys., 11, 114
Hu X. Y., Adams N. A., Shu C.-W., 2013, J. Comput. Phys., 242, 169
Janka H. T., Langanke K., Marek A., Martínez-Pinedo G., Müller B., 2007, Phys. Rep., 442, 38
Kaspi V. M., Beloborodov A. M., 2017, ARA&A, 55, 261
Kastaun W., Kalinani J. V., Ciolfi R., 2021, Phys. Rev. D, 103, 023018
Keppens R., Meliani Z., van Marle A. J., Delmont P., Vlasis A., van der Holst B., 2012, J. Comput. Phys., 231, 718
Keppens R., Teunissen J., Xia C., Porth O., 2020, preprint (arXiv:2004.03275)
Komissarov S. S., 1999, MNRAS, 303, 343
Kwak D. Y., 1999, SIAM J. Sci. Comput., 21, 552
Liska M. et al., 2019, preprint (arXiv:1912.10192)
Löhner R., 1987, Comput. Methods in Appl. Mech. Eng., 61, 323
Lorimer D. R., 2005, Liv. Rev. Relativ., 8, 7
MacNeice P., Olson K. M., Mobarry C., de Fainchtein R., Packer C., 2000, Comput. Phys. Commun., 126, 330
Martí J. M., Müller E., 2003, Liv. Rev. Relativ., 6, 7
Martí J. M., Müller E., Font J. A., Ibáñez J. M. Z., Marquina A., 1997, ApJ, 479, 151
Mereghetti S., Pons J. A., Melatos A., 2015, Space Sci. Rev., 191, 315
Metzger B. D., 2017, Liv. Rev. Relativ., 20, 3
Mewes V., Zlochower Y., Campanelli M., Baumgarte T. W., Etienne Z. B., Armengol F. G. L., Cipolletta F., 2020, Phys. Rev. D, 101, 104007
Mignone A., Plewa T., Bodo G., 2005, ApJS, 160, 199
Mignone A., Bodo G., Massaglia S., Matsakos T., Tesileanu O., Zanni C., Ferrari A., 2007, ApJS, 170, 228
Mignone A., Zanni C., Tzeferacos P., van Straalen B., Colella P., Bodo G., 2012a, ApJS, 198, 7
Mignone A., Flock M., Stute M., Kolb S. M., Muscianisi G., 2012b, A&A, 545, A152
Mohr M., Wienands R., 2004, Comput. Vis. Sci., 7, 129
Montero P. J., Baumgarte T. W., Müller E., 2014, Phys. Rev. D, 89, 084043
Mösta P. et al., 2014, Class. Quantum Gravity, 31, 015005
Müller B., 2020, Liv. Rev. Comput. Astrophys., 6, 3
Olivares H., Porth O., Davelaar J., Most E. R., Fromm C. M., Mizuno Y., Younsi Z., Rezzolla L., 2019, A&A, 629, A61
Ott C. D., 2009, Class. Quantum Gravity, 26, 063001
Pili A. G., Bucciantini N., Del Zanna L., 2014, MNRAS, 439, 3541
Pili A. G., Bucciantini N., Del Zanna L., 2015, MNRAS, 447, 2821
Pili A. G., Bucciantini N., Del Zanna L., 2017, MNRAS, 470, 2469
Porth O., Olivares H., Mizuno Y., Younsi Z., Rezzolla L., Moscibrodzka M., Falcke H., Kramer M., 2017, Comput. Astrophys. Cosmol., 4, 1
Press W. H., Teukolsky S. A., Vetterling W. T., Flannery B. P., 1992, Numerical Recipes in FORTRAN (2nd ed.): The Art of Scientific Computing. Cambridge Univ. Press, New York, NY, USA
Radice D., Rezzolla L., Galeazzi F., 2014, Class. Quantum Gravity, 31, 075012
Radice D., Bernuzzi S., Perego A., 2020, Ann. Rev. Nucl. Part. Sci., 70, 95
Ripperda B. et al., 2019, ApJS, 244, 10
Rosswog S., 2015, Int. J. Mod. Phys. D, 24, 1530012
Schive H.-Y., Tsai Y.-C., Chiueh T., 2010, ApJS, 186, 457
Schive H.-Y., ZuHone J. A., Goldbaum N. J., Turk M. J., Gaspari M., Cheng C.-Y., 2018, MNRAS, 481, 4815
Shibata M., 2015, in Numerical Relativity. World Scientific Publishing Co. Pte. Ltd
Shibata M., Sekiguchi Y.-I., 2004, Phys. Rev. D, 69, 084024
Shibata M., Shapiro S. L., 2002, ApJ, 572, L39
Shibata M., Taniguchi K., 2011, Liv. Rev. Relat., 14, 6
Skinner M. A., Dolence J. C., Burrows A., Radice D., Vartanyan D., 2019, ApJS, 241, 7
Stone J. M., Gardiner T. A., Teuben P., Hawley J. F., Simon J. B., 2008, ApJS, 178, 137
Stone J. M., Tomida K., White C. J., Felker K. G., 2020, ApJS, 249, 4
Stout Q. F., De Zeeuw D. L., Gombosi T. I., Groth C. P. T., Marshall H. G., Powell K. G., 1997, Proceedings of the 1997 ACM/IEEE Conference on Supercomputing. SC '97. Association for Computing Machinery, New York, NY, p. 1
Teunissen J., Keppens R., 2019, Comput. Phys. Commun., 245, 106866
Turolla R., Zane S., Watts A. L., 2015, Rep. Prog. Phys., 78, 116901
van der Holst B., Keppens R., 2007, J. Comput. Phys., 226, 925
van Leer B., 1974, J. Comput. Phys., 14, 361
Xia C., Teunissen J., El Mellah I., Chané E., Keppens R., 2018, ApJS, 234, 30
Zanotti O., Dumbser M., 2015, Comput. Phys. Commun., 188, 110
Zhang W., MacFadyen A. I., 2006, ApJS, 164, 255

APPENDIX A: FLAT METRIC IN 3D

The cell volume ΔV, cell surface ΔA, and the volume-average of the 3-Christoffel symbols $\left\langle \hat{\Gamma}^{l}_{ik} \right\rangle$ which is contained in the geometrical are non-trivial when the reference metric $\hat{\gamma}_{ij}$ is chosen to be cylindrical or spherical. Here, we list out the relation we implemented in `Gmunu`.

A1 cylindrical coordinate

The line element can be expressed as: $ds^2 = dR^2 + dz^2 + R^2 d\varphi^2$, with the reference metric $\hat{\gamma}_{ij}$:

$$\hat{\gamma}_{ij} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & R^2 \end{bmatrix}. \tag{A1}$$

The associated 3-Christoffel symbols $\hat{\Gamma}^l_{ik}$ are:

$$\hat{\Gamma}^R_{ij} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -R \end{bmatrix}, \quad \hat{\Gamma}^z_{ij} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

$$\hat{\Gamma}^\varphi_{ij} = \begin{bmatrix} 0 & 0 & \frac{1}{R} \\ 0 & 0 & 0 \\ \frac{1}{R} & 0 & 0 \end{bmatrix}. \tag{A2}$$

The geometrical source terms for the momentum equations are:

$$\hat{\Gamma}^l_{Rk}(f_{S_l})^k = \hat{\Gamma}^\varphi_{R\varphi}(f_{S_\varphi})^\varphi, \tag{A3}$$

$$\hat{\Gamma}^l_{zk}(f_{S_l})^k = 0, \tag{A4}$$

$$\hat{\Gamma}^l_{\varphi k}(f_{S_l})^k = 0. \tag{A5}$$

Here, we note that since z and φ do not explicitly enter into the reference metric $\hat{\gamma}_{ij}$, the corresponding geometrical source terms for the momentum equation q_{S_j} are vanishing. In this formulations, the linear momentum q_{S_z} and the angular momentum q_{S_φ} are conserved to machine precision.

To work out the cell volume ΔV, cell surface ΔA, and the volume-average of the 3-Christoffel symbols $\langle \hat{\Gamma}^l_{ik} \rangle$, we define the following notations:

$$R_\pm \equiv R \pm \frac{1}{2}\Delta R, \quad z_\pm \equiv z \pm \frac{1}{2}\Delta z, \quad \varphi_\pm \equiv \varphi \pm \frac{1}{2}\Delta\varphi, \tag{A6}$$

where (R, z, φ) are the location at the cell centre at some particular point while $(\Delta R, \Delta z, \Delta\varphi)$ are the corresponding grid sizes. The cell surface ΔA and the cell volume ΔV can then be expressed as

$$\Delta A_R\big|_{R_\pm} = \left(R \pm \frac{\Delta R}{2}\right)(\Delta z)(\Delta\varphi), \tag{A7}$$

$$\Delta A_z\big|_{z_\pm} = R(\Delta R)(\Delta\varphi), \tag{A8}$$

$$\Delta A_\varphi\big|_{\varphi_\pm} = R(\Delta R)(\Delta z), \tag{A9}$$

$$\Delta V = R(\Delta R)(\Delta z)(\Delta\varphi). \tag{A10}$$

Finally, the non-vanishing volume-averaged 3-Christoffel symbols $\langle \hat{\Gamma}^l_{ik} \rangle$ are:

$$\langle \hat{\Gamma}^R_{\varphi\varphi} \rangle = -\frac{1}{R}\left(R^2 + \frac{1}{12}(\Delta r)^2\right), \tag{A11}$$

$$\langle \hat{\Gamma}^\varphi_{\varphi R} \rangle = \langle \hat{\Gamma}^\varphi_{R\varphi} \rangle = \frac{1}{R}. \tag{A12}$$

A2 Spherical coordinates

The line element can be expressed as: $ds^2 = dr^2 + r^2 d\theta^2 + r^2 \sin^2\theta d\phi^2$, with the reference metric $\hat{\gamma}_{ij}$:

$$\hat{\gamma}_{ij} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & r^2 & 0 \\ 0 & 0 & r^2\sin^2\theta \end{bmatrix}. \tag{A13}$$

The associated 3-Christoffel symbols $\langle \hat{\Gamma}^l_{ik} \rangle$ are:

$$\hat{\Gamma}^r_{ij} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -r & 0 \\ 0 & 0 & -r\sin^2\theta \end{bmatrix}, \tag{A14}$$

$$\hat{\Gamma}^\theta_{ij} = \begin{bmatrix} 0 & \frac{1}{r} & 0 \\ \frac{1}{r} & 0 & 0 \\ 0 & 0 & -\sin\theta\cos\theta \end{bmatrix}, \tag{A15}$$

$$\hat{\Gamma}^\phi_{ij} = \begin{bmatrix} 0 & 0 & \frac{1}{r} \\ 0 & 0 & \cot\theta \\ \frac{1}{r} & \cot\theta & 0 \end{bmatrix}. \tag{A16}$$

The geometrical source terms for the momentum equations are:

$$\hat{\Gamma}^l_{rk}(f_{S_l})^k = \hat{\Gamma}^\theta_{r\theta}(f_{S_\theta})^\theta + \hat{\Gamma}^\phi_{r\phi}(f_{S_\phi})^\phi, \tag{A17}$$

$$\hat{\Gamma}^l_{\theta k}(f_{S_l})^k = \hat{\Gamma}^r_{\theta\theta}(f_{S_r})^\theta + \hat{\Gamma}^\theta_{\theta r}(f_{S_\theta})^r + \hat{\Gamma}^\phi_{\theta\phi}(f_{S_\phi})^\phi, \tag{A18}$$

$$\hat{\Gamma}^l_{\phi k}(f_{S_l})^k = 0. \tag{A19}$$

Similarity, as in the cylindrical case, the angular momentum q_{S_ϕ} are conserved to machine precision.

To work out the cell volume ΔV, cell surface ΔA, and the volume-average of the 3-Christoffel symbols $\langle \hat{\Gamma}^l_{ik} \rangle$, we define the following notations:

$$r_\pm = r \pm \frac{1}{2}\Delta r, \quad \theta_\pm = \theta \pm \frac{1}{2}\Delta\theta, \quad \phi_\pm = \phi \pm \frac{1}{2}\Delta\phi, \tag{A20}$$

where (r, θ, ϕ) are the location at the cell centre at some particular point while $(\Delta r, \Delta\theta, \Delta\phi)$ are the corresponding grid sizes. The cell surface ΔA and the cell volume ΔV can then be expressed as

$$\Delta A_r\big|_{r_\pm} = \left(r \pm \frac{\Delta r}{2}\right)^2 \left(2\sin\theta\sin\left(\frac{\Delta\theta}{2}\right)\right)(\Delta\phi) \tag{A21}$$

$$\Delta A_\theta\big|_{\theta_\pm} = \left(\left(r^2 + \frac{1}{12}(\Delta r)^2\right)\Delta r\right)\left(\sin\left(\theta \pm \frac{\Delta\theta}{2}\right)\right)(\Delta\phi) \tag{A22}$$

$$\Delta A_\phi\big|_{\phi_\pm} = \left(\left(r^2 + \frac{1}{12}(\Delta r)^2\right)\Delta r\right)\left(2\sin\theta\sin\left(\frac{\Delta\theta}{2}\right)\right) \tag{A23}$$

$$\Delta V = \left(\left(r^2 + \frac{1}{12}(\Delta r)^2\right)\Delta r\right)\left(2\sin\theta\sin\left(\frac{\Delta\theta}{2}\right)\right)(\Delta\phi) \tag{A24}$$

Finally, the non-vanishing volume-averaged 3-Christoffel symbols $\langle \hat{\Gamma}^l_{ik} \rangle$ are

$$\langle \hat{\Gamma}^\theta_{r\theta} \rangle = \langle \hat{\Gamma}^\theta_{\theta r} \rangle = \langle \hat{\Gamma}^\phi_{r\phi} \rangle = \langle \hat{\Gamma}^\phi_{\phi r} \rangle = \frac{1}{\Delta V}\frac{1}{2}\left(\Delta A_r\big|_{r_+} - \Delta A_r\big|_{r_-}\right) \tag{A25}$$

$$\langle \hat{\Gamma}^r_{\theta\theta} \rangle = -\frac{1}{\Delta V}\frac{1}{4}\left(r_+^2 \Delta A_r\big|_{r_+} - r_-^2 \Delta A_r\big|_{r_-}\right) \tag{A26}$$

$$\langle \hat{\Gamma}^r_{\phi\phi} \rangle = \frac{1}{\Delta V}\left(-\frac{1}{4}r^4\Big|_{r_-}^{r_+}\right)\left(\frac{1}{3}\cos^3\theta - \cos\theta\right)\Big|_{\theta_-}^{\theta_+}\Delta\phi \tag{A27}$$

$$\langle \hat{\Gamma}^\theta_{\phi\phi} \rangle = -\frac{1}{3}\frac{1}{\Delta V}(\sin^2(\theta_+)\Delta A_\theta|_{\theta_+} - \sin^2(\theta_-)\Delta A_\theta|_{\theta_-}) \tag{A28}$$

$$\langle \hat{\Gamma}^\phi_{\theta\phi} \rangle = \langle \hat{\Gamma}^\phi_{\phi\theta} \rangle = \cot\theta \tag{A29}$$

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